Vrije Universiteit Brussel
Faculty of Sciences and Bioengineering Sciences
Department of Computer Science

MindXpres

Conceptual and Technical Foundations for Next Generation Presentation Solutions

PhD Candidate:
Reinout Roels

Promoter:
Prof. Dr. Beat Signer

Dissertation submitted in fulfilment of the
requirements for the degree of Doctor of Sciences

Members of the Jury

Prof. Dr. Olga De Troyer	Vrije Universiteit Brussel	(Chair)
Prof. Dr. Johan Loeckx	Vrije Universiteit Brussel	(Secretary)
Prof. Dr. Beat Signer	Vrije Universiteit Brussel	(Promoter)
Prof. Dr. Jo Tondeur	Vrije Universiteit Brussel	
Prof. Dr. Kris Luyten	Universiteit Hasselt	
Prof. Dr. Katrien Verbert	KU Leuven	

May 2019

Abstract

Presentation software such as PowerPoint, also known as *slideware*, has become a de facto standard as support for oral presentations. However, these tools were initially created as simulators for physical slides and have adopted many of their characteristics, including their limitations. These limitations include a strictly linear navigation of presentations, relatively static content and a tedious and time-consuming authoring process. Presentation software was introduced over 30 years ago but these tools have changed very little since their inception. This is especially remarkable given the other technological advances that were made in the same timespan. Even though slideware may not be the optimal medium for all scenarios, the fact remains that it is used everywhere. Therefore, addressing existing shortcomings and unmet user needs can have a major impact. In this dissertation we start by investigating the existing shortcomings of presentation tools. By means of a literature study, observations, a survey and the programmatic analysis of more than 12 000 PowerPoint documents, we derive user needs and pinpoint issues that should be addressed. Our experiments resulted in a large dataset and a database of extracted metadata that will further help other researchers in the domain.

As we show, existing presentation tools enforce specific (arguably outdated) paradigms which makes it difficult to address the observed shortcomings directly in existing tools. We motivate the need to revisit the concept of a presentation and to rebuild the conceptual and technical foundations that are required for improved presentation tools. This resulted in the MindXpres platform, a reusable presentation prototyping platform that allows researchers and developers to easily implement and evaluate novel presentation functionality. We present a new conceptual framework, content model and presentation engine with the aim of enabling novel presentation solutions. Our underlying information system is based on advanced hypermedia concepts and provides core support for content reuse, collaborative authoring, user management, versioning, semantics and context awareness. The MindXpres presentation engine's plug-in architecture is more flexible than existing presentation software, enabling a wide range of presentation styles, data visualisations, audience interactions and solu-

tions that help presenters to deliver their content more effectively. Any component can easily be extended or replaced, and we support both existing presentation styles such as Prezi-like canvas presentations or PowerPoint's classic slideshows, as well as some new presentation styles. The extensible architecture further provides support for content types that are not easily visualised in other presentation software, such as dynamic, interactive or audience-driven content.

After providing the necessary conceptual and technical foundation we demonstrate the flexibility of the MindXpres platform by developing a number of proof-of-concept solutions for some of the identified shortcomings and unmet user needs. We discuss various plug-in solutions related to the management of presentation content, a better involvement of the audience in presentations, interactive and dynamic data visualisation, providing feedback on the presenter's performance and content, novel presenter views and some solutions that allow the presenter to create better content with less effort. As a presentation platform, MindXpres offers the conceptual and technical foundations that allow developers and researchers to create well integrated and interoperable presentation solutions, and to realise currently unsupported next generation presentation concepts. Ultimately, our contributions and proof-of-concept solutions might pave the way for more efficient presentation authoring and more effective knowledge transfer via oral presentations.

Acknowledgements

First of all, I would like to thank the **promoter** of this thesis, Prof. Dr. Beat Signer. Beat has taught me a lot and I am very grateful for the countless hours he has spent on providing feedback. Every year I look back, and every year I see how Beat's guidance has continued to improve my skills.

Next I would like to thank the **members of the jury**: Prof. Dr. Olga De Troyer, Prof. Dr. Johan Loeckx, Prof. Dr. Jo Tondeur, Prof. Dr. Katrien Verbert and Prof. Dr. Kris Luyten. The effort that they put into reviewing my work resulted in fruitful discussions and their feedback has definitely improved this final version of the dissertation.

During my PhD trajectory I supervised 2 Bachelor's and 16 Master's theses. Many of these were on the topic of MindXpres and I thank all these **students** for the contributions that they have made to this research project. A complete list of students can be found in Appendix A, and their contributions will be referenced in the relevant sections of this dissertation.

I could not have finished my PhD without also having some fun, and for that I thank all my **friends** and **colleagues**. The list is too long to thank everyone separately, but you know who you are. Whether it was help, discussions, game nights, drinking sessions, LAN parties, trying random new restaurants or painstakingly walking 100km, these moments have kept me sane and will never be forgotten.

Last, but definitely not least, I would like to express my gratitude to my **family**. I want to thank my parents, as their continuous support has gotten me where I am today. I thank my parents-in-law and sister-in-law for their help and for the immeasurable amounts of delicious food and orange juice that kept me fuelled. Most importantly, I thank **Shirley**, my life partner. Life with a newborn baby is not easy, and life with a newborn baby *and* a person finishing their PhD must be impossible. Yet, you always understood and you were always there for me. Your advice, patience and sacrifices have allowed us to get through this rough period, and I look forward to the other adventures we will have together.

Contents

I Introduction, Background and Related Work 1

1 Introduction 3
1.1 Context . . . 3
1.2 Terminology . . . 4
1.3 Problem Statement . . . 5
1.4 Research Questions and Objectives . . . 6
1.5 Research Approach and Methodology . . . 8
1.6 Contributions . . . 12
1.7 Publications . . . 12

2 Background and Related Work 17
2.1 History of Presentation Software . . . 17
2.2 Commonly Used Presentation Software . . . 18
2.2.1 Commercial Presentation Software . . . 19
2.2.2 Markup Languages and Software Frameworks . . . 20
2.3 The Pros and Cons of Slideware . . . 20
2.3.1 Effect of Slideware on Learning Performance . . . 21
2.3.2 Effect of Slideware on Attitude and Perception . . . 22
2.3.3 Effect of Slideware on Attendance . . . 22
2.3.4 Effect of Slideware on Presentation Flow . . . 22
2.3.5 Effect of Slideware on Presentation Content . . . 23
2.3.6 Effect of Slideware on Presenter Skills . . . 25
2.4 State of the Art of Presentation Solutions . . . 25
2.4.1 Zoomable User Interfaces (ZUIs) . . . 26
2.4.2 Non-linear Navigation . . . 27
2.4.3 Content Reuse and Collaboration . . . 28
2.4.4 Semantics . . . 29
2.4.5 Dynamic and Interactive Content . . . 29
2.4.6 Audience Involvement . . . 30
2.4.7 Presentation Skills . . . 31
2.5 Conclusions Drawn from Related Work . . . 32

II Identifying Shortcomings and Unmet User Needs 35

3 User Needs, Preferences and Requirements 37
3.1 Literature Review . . . 37
3.1.1 Need for Skilled Presenters and Good Presentations . . . 38
3.1.2 Need for Content Sharing, Reuse and Collaboration . . . 38
3.1.3 Need for Improved Content Support . . . 40
3.1.4 Need for Improved Slide Design Support . . . 40
3.2 Observations . . . 41
3.3 Programmatic Analysis of PowerPoint Documents . . . 43
3.3.1 A Representative Sample of Presentations . . . 44
3.3.2 A Software Framework for Analysing the Sample . . . 45
3.3.3 Analysis Findings . . . 47
3.4 Survey . . . 55
3.4.1 Survey Findings . . . 56
3.4.2 Survey Conclusions . . . 68
3.5 Summary of User Needs, Preferences and Requirements . . . 69

III MindXpres: Conceptual and Technical Foundations 75

4 A Conceptual Framework for Next Generation Presentations 77
4.1 Next Generation Presentation Concepts and Functionality . . . 78
4.2 A Unifying Conceptual Framework . . . 81

5 A Hypermedia-based Content Model and Information System 87
5.1 Existing Document Formats and Content Models . . . 88
5.1.1 OOXML, ODF and Similar Presentation Document Formats . . . 88
5.1.2 Learning Material Content Models . . . 89
5.2 Towards a Unifying "Future-Proof" Content Model for Presentations . . . 90
5.2.1 Model Requirements . . . 91
5.2.2 The Original Hypermedia Vision . . . 94
5.3 The Resource-Selector-Link (RSL) Hypermedia Metamodel . . . 95
5.4 Extensions to the RSL Hypermedia Metamodel . . . 97
5.5 The MindXpres Content Model . . . 100
5.6 Model Functionality and Discussion . . . 103
5.7 Implementation . . . 107

6 The MindXpres Presentation Engine 117
6.1 Technical Platform Requirements . . . 118
6.2 Architecture . . . 119
6.2.1 Compiling MindXpres Documents into Presentations . . . 120
6.2.2 MindXpres Runtime Engine . . . 121
6.3 Implementation . . . 123
6.3.1 Document Format and Authoring Language . . . 123
6.3.2 Compiler . . . 124
6.3.3 Runtime Engine . . . 124
6.4 Example Plug-ins and Presentation . . . 129
6.5 Discussion . . . 130

IV Addressing Shortcomings and Unmet User Needs 133

7 Solutions for Content Visualisation 137
7.1 Dynamic Diagram Visualisations . . . 137
7.2 Mathematical Expressions via MathML or LaTeX . . . 139
7.3 Advanced Video Player . . . 140
7.4 Source Code Visualisation . . . 141
7.5 Interactive Source Code Visualisation for Teachers . . . 143
7.6 Interactive and Narrative Data Visualisation . . . 147
7.7 Embedded Linux Environment . . . 158
7.8 Doom . . . 159

8 Solutions for Involving the Audience 161
8.1 Polls and Quizzes . . . 161
8.2 Navigation Mirroring and Takeover . . . 163
8.3 Digital Questions . . . 165
8.4 Shareable Highlights and Notes . . . 166

9 Solutions for Supporting the Presenter During the Presentation 167
9.1 Multimodal Presentation Interaction . . . 167
9.2 TabletPoint: A Pen and Tablet-based Presenter Tool . . . 171
9.3 Presentation Feedback System . . . 175
9.4 Augmented Reality Presenter View . . . 179
9.5 Virtual Presentation Assistant . . . 183

10 Solutions for Authoring, Reuse and Collaboration 187
10.1 Guided Data Visualisation . . . 187
10.2 PowerPoint Import and Automatic Content Layout . . . 190
10.3 Graphical Authoring Tools . . . 193
10.3.1 Graphical Template Authoring Tool . . . 193
10.3.2 Graphical Presentation Authoring Tool . . . 195

V Discussion and Future Work 199

11 Conclusions and Future Work 201
11.1 Summary . . . 201
11.2 Discussion and Future Work . . . 207
11.3 Final Conclusions . . . 215

VI Appendices 217

Appendix A Related Student Contributions 219

Appendix B Database Schema for Programmatic Analysis Results 221

Appendix C Questionnaires 225
C.1 System Usability Scale (SUS) . . . 225
C.2 Computer System Usability Questionnaire (CSUQ) . . . 227
C.3 Custom Survey . . . 229

Appendix D RSL Implementation: Java Classes 239

Appendix E Used Tools and Technologies 241

VII Bibliography 243

Part I

Introduction, Background and Related Work

Chapter 1

Introduction

1.1 Context

Presentation software such as Microsoft PowerPoint was introduced in 1987 and has since become the de facto standard as a tool for oral presentations in education, academia and business. Slide-based presentation software was initially created so that users could digitally design their slides before printing them on physical media such as photographic slides, saving users a lot of time and money. We see that even now, 30 years later, presentation software still mimics the characteristics of their physical counterparts. For example, most presentations still consist of a linear sequence of spatially restricted slides, and the content of a presentation is often relatively static. In contrast, the computers that are used to deliver these presentations underwent major improvements over the last 30 years and have become capable of much more. Turing Award recipient Alan Kay went as far as stating that "*PowerPoint is just simulated acetate overhead slides, and to me, that is a kind of a moral crime*". In the more general context of digital documents, Ted Nelson is known for comparing digital documents to "*paper under glass*" [203], criticising that the tools for digital documents merely simulate physical paper on a display, with all its limitations. In the context of learning technologies, Bush remarks that learning technologies fall short of their aims and that this "*stems from a penchant to implement technology in ways that automate that past*" [43]. In a way, this also applies to presentation software and related tools. One might even state that some functionality was lost during the evolution towards digital slides. For instance, blackboards offer more space, a better overview and make it easier to relate different content. Transparencies for overhead projectors made it easier to present content out of order and to make annotations during the presentation. Nevertheless, the use of slideshows has embedded itself in

many aspects of modern society and remains popular. It is difficult to find exact user and market share statistics, but already in 2001 Microsoft estimated that 30 million PowerPoint presentations were created *every day* [211, 74], whereas in 2004 it was estimated that 400 million copies of PowerPoint were in use [253] and in 2010 Microsoft announced that Microsoft Office (which includes PowerPoint) reached one billion users[1]. We later show that presentation content also serves a purpose after the presentation session, for instance as learning material, reference material or as reports. However, tools such as PowerPoint were created to support a presenter's oral narrative and not for scientific reporting or for creating educational content. Slideware might not be the ideal medium for every scenario but the fact remains that it is used everywhere anyway, even with its shortcomings. It is too late to stop the "misuse" of slideware, but we can make an impact by enabling solutions for some existing shortcomings and unmet user needs. By providing proper support for currently unsupported presentation concepts, new solutions can be built that might allow presenters to create better presentations with less effort, help mitigate the negative effects that result from "misusing" presentation software, improve knowledge transfer or which make presentation software more applicable for a wider range of use cases.

1.2 Terminology

We briefly explain some of the recurring terminology that is used throughout this dissertation. Some explanations may seem trivial but many terms have ambiguous meanings and are used in different ways even in an academic context. We choose to clarify our interpretation of these terms early on, in order to avoid any potential confusion later in this chapter or in the rest of this dissertation.

Presentation: In the context of this dissertation we always see presentations as *oral* presentations. More specifically, a presentation is a situation where knowledge is orally transferred from a presenter to an audience, but it might involve the use of supporting visuals. During a typical presentation the presenter is in charge of the narrative. The audience is expected to respect this and presentations often lean towards a unidirectional flow of communication, which makes them different from a regular conversation between people.

Presentation tools: A presentation tool is any tool, software or hardware, that supports presenters in creating or delivering presentations. This includes the obvious presentation software such as PowerPoint but it also covers overhead projectors, blackboards, laser pointers or even wooden pointer sticks for pointing out content.

[1] `https://twitter.com/Office/status/16222613591`

Presentation tools also include tools for creating presentation content as well as tools that help presenters to practise their presentation before delivering it to an audience.

Presentation software: Presentation software is a subset of presentation tools which consists of purely digital solutions. Typically this term is used to refer to tools such as PowerPoint or Prezi[1]; complete software packages supporting both the authoring and delivery of digital presentations.

Slideware: Presentation software such as PowerPoint is based on the slide paradigm which enables the "sliding" through a sequence of rectangular slides, much like the older slide projectors based on photographic film. We refer to presentation software that follows the slide paradigm as slideware. Presentation software such as Prezi does not belong to this category, as it is based on a zoomable canvas and not on the concept of slides.

1.3 Problem Statement

The fact that slideware is used everywhere does not necessarily mean that presentation software is a perfect solution for oral presentations. PowerPoint's market share indicates that there are not many alternatives, and most of them are based on the same principles as PowerPoint. For this reason users often assume that slideware is the only way to deliver a presentation. Many users have learned to work around some of the limitations and might not even consider that alternative approaches can make their lives easier and remove the need for tedious workarounds. However, in the academic world, the use of digital slides has received its fair share of critique. Tufte's essay 'The Cognitive Style of PowerPoint' [278] is a popular example, pointing out various issues and shortcomings of slideware. As we show later, the added value of slideware is still heavily debated and there are compelling arguments in favour, as well as against slideware tools. Furthermore, academic work has identified more concrete shortcomings related to content visualisation, navigation, interactivity, presentation authoring, content reuse or content sharing. The related work does not only identify shortcomings but also some potential solutions are investigated. However, we will show that existing academic solutions are often isolated stand-alone proof-of-concepts. These solutions have to work around the restrictions enforced by existing tools such as PowerPoint and as a result they are often offered as workarounds that are not interoperable, not extensible, not well integrated or not reusable. As another example of the inflexibility of existing tools, research in pedagogy and information visualisation has shown irrefutable evidence that concepts such as active learning or interactive visualisations can bring many benefits. However, various technical as well

[1] `https://prezi.com`

as ideological restrictions make it difficult, if not impossible, for presenters to apply these findings in the existing presentation software.

We see that user needs have changed and the concept of a presentation is slowly evolving beyond what is supported by existing tools. Especially in related work we see a lot of upcoming new trends that go far beyond what is currently offered by PowerPoint. Yet, PowerPoint has barely changed since its inception and still enforces the idea that a presentation should always be a linear sequence of static content. We argue that in order to provide proper solutions for the existing shortcomings and unmet user needs, we need to rethink the concept of a presentation and we need to create a common foundation that does not make assumptions about what a presentation can or cannot be. One of the major challenges is that such a conceptual framework has to be generic enough to unify existing state-of-the-art presentation concepts but it should also be flexible enough to represent presentation concepts that have yet to come. In addition to a *conceptual foundation*, we also need to make changes to our presentation software and the underlying technologies. By creating a new *technical foundation* we can offer the technical support that helps researchers and developers to bring currently unsupported next generation presentation concepts to end users. Ultimately, we aim to provide the foundations that enable a new generation of presentation solutions that can address existing shortcomings and unmet user needs. However, in order to properly support solutions for existing shortcomings it is also important to gain insight into the existing shortcomings and user needs, and to design our foundations accordingly. The resulting conceptual and technical foundations can then enable a new generation of presentation solutions that are interoperable, well-integrated and more suitable for a variety of use cases. These new solutions might for instance allow presenters to create better presentations with less effort, they might help us to implement beneficial techniques from pedagogy or information visualisation, or they might make presentation software more effective for common use cases that are not taken into account by existing tools.

1.4 Research Questions and Objectives

From literature, informal discussions with peers at international conferences and other events we observed strong statements in favour as well as against presentation software. However, given the many different opinions and stances we felt it was necessary to first find answers to common questions such as "*Is the use of presentation software even beneficial?*", "*Does slideware really play an important role in education?*" or "*Can we not just create minimal slides like Steve Jobs to avoid any issues?*". This resulted in the formulation of the first research question:

- **RQ 1**: How is presentation software used and what are the benefits or downsides?

Answering this question would give us insight into existing presentation tools and allow us to shape the rest of our research. This research question is later answered in this dissertation by basing ourselves on the large corpus of existing studies on that matter. Even though we did not directly contribute to the answers ourselves, it is not a trivial question and the outcome might have had a huge impact on our research course, making it a very important question. For example, if shortcomings could really be avoided by simply using minimal slides without text, a lot of the contributions we made later would lose their value. This is however not the case and this finding helped us to justify our later contributions.

Given that the first objective was to obtain a better understanding on the effect of presentation software, the next research question was formulated as:

- **RQ 2**: What are the shortcomings and unmet user needs in existing presentation tools?

In other words, a next research objective was to show that there are some concrete shortcomings and unmet user needs that should be addressed in presentation tools and their underlying technologies. After identifying various shortcomings and unmet user needs we also intended to provide the necessary conceptual and technical support for enabling concrete solutions, resulting in the following research question:

- **RQ 3**: How do we need to extend, modify or rebuild presentation software and the underlying technologies to allow researchers and developers to address existing shortcomings and user needs?

However, later in this dissertation we show that existing tools and technologies cannot easily be extended or adapted for next generation presentation concepts. As a result we rebuilt the conceptual and technical foundations and enabled new improvements in presentation software and the related tools. In order to demonstrate the flexibility of the platform we designed and built various proof-of-concept solutions for some of the shortcomings and user needs. This resulted in the forth and last research question:

- **RQ 4**: Given our new conceptual and technical foundation that is free of some of the restrictions seen in other presentation tools, what are the innovative solutions that can be provided for the documented shortcomings and user needs?

Answering this research question allowed us to fulfil the objective of investigating and providing some concrete solutions for proven shortcomings. Each shortcoming usually resulted in a new contribution that was properly designed, implemented and some of them also evaluated, forming smaller research artefacts on their own.

1.5 Research Approach and Methodology

In this section we detail the research methodology we used for providing answers to the research questions and for achieving the objectives mentioned above. As a framework for the described research we used the Design Science Research Methodology (DSRM) for information systems research, defined by Peffers et al. [212]. The proposed DSRM was designed for the creation of theoretical artefacts such as models and algorithms as well as more concrete outcomes such as software frameworks and user interfaces. As shown in this section, the steps of the chosen DSRM align well with the research objectives defined in Section 1.4. As our research revolves around the storage, management and visualisation of information, the DSRM's focus on information systems research makes it a particularly suitable methodology.

The utilised DSRM process model builds on prior research in design science and defines the following six steps:

1. Problem identification and motivation
2. Define the objectives for a solution
3. Design and development
4. Demonstration
5. Evaluation
6. Communication

For each of these steps we summarise how they were approached for the research described in this dissertation.

Step 1: Problem Identification and Motivation

The main problem can be summarised by stating that existing presentation tools have some shortcomings, as discussed earlier in Section 1.3, but due to technical as well as ideological restrictions it is difficult to address these shortcomings in existing tools. By investigating related work we gather previously documented shortcomings and by means of literature study, observations, a survey and a programmatic analysis of PowerPoint documents we identify a set of problems that are not fully addressed by existing tools. This allowed us to provide answers to RQ 1 and RQ 2. Concretely, the identified limitations are related to how presentation content is reused and shared, how content is visualised and interacted with by presenters or audience members, and how presentation tools can better support the presenter before or during the presentation. Although there is some related work, academic solutions are often isolated stand-alone proof-of-concept solutions that work around the restrictions of

existing tools, and they are often not interoperable, extensible or reusable. In order to enable interoperable and well-integrated solutions for existing shortcomings we motivate the need for new conceptual and technical foundations that are generic, extensible and without restrictions.

Step 2: Define the Objectives for a Solution

With the following objectives we aim to create a new presentation platform that *enables* new functionality which cannot be offered based on existing approaches or tools. Later in this dissertation we motivate that it is necessary to revisit the concept of a presentation and to rebuild the conceptual and technical foundations for an improved presentation platform so that researchers and developers can address the identified shortcomings and unmet user needs. Concretely, this results in the following objectives.

- Unlike existing tools, our platform should not make restrictive assumptions on what a presentation is and how it is visualised or navigated. As a first solution objective, we should conceptually unify and abstract common as well as state-of-the-art academic presentation concepts in order to conceptually frame all existing, and to some degree also future functionality.
- As a result of the first solution objective we argue the need to design and implement a new content model and information system that can represent and store next generation presentations. In addition to simply storing content we also aim to enable improved content sharing, reuse and collaboration.
- In order to visualise next generation presentations the next objective is a new presentation engine that is flexible and extensible enough to support a wide range of presentation styles, interactive and dynamic visualisations as well as other next generation presentation concepts. As will be shown, other requirements such as portability and a presentation's "afterlife" also come into play.

We would like to point out that the formulation of some solution objectives was influenced by the outcome of other solution objectives, and only the summary is provided above. Any choices and claims made above are of course fully motivated later in this dissertation as we discuss the execution of these objectives.

Step 3: Design and Development

In line with the objectives defined in Step 2, we created the conceptual and technical foundations for the new MindXpres presentation platform. First, we developed a new conceptual framework that unifies existing presentation concepts and we present this framework in Chapter 4. We then focus on bringing the extensibility, flexibility and

advanced presentation concepts from the conceptual framework to the technical level. Driven by the concepts defined in the conceptual framework we introduce our new content model and information system for next generation presentation solutions in Chapter 5. In Chapter 6 we detail our new flexible and extensible presentation engine, bringing the characteristics and functionality of the conceptual framework and information system to the visualisation layer.

Step 4: Demonstration

As required by the chosen DSRM we demonstrate the effectiveness of our artefacts in various ways. As a result, all of our contributions have been implemented as functional software applications that help us to demonstrate the benefits of our solutions. By using our new information system to represent and store a variety of complex presentations, we demonstrate that it can handle next generation presentation concepts not supported by other systems. We simulated reuse and collaboration directly in the data layer, and provide a basic graphical authoring tool, further demonstrating that it meets the previously mentioned objectives.

In order to demonstrate how the MindXpres platform enables solutions that address some of the existing shortcomings, we designed and implemented a wide range proof-of-concept solutions on top of the MindXpres platform. Concretely, we based ourselves on the most common shortcomings and user needs, and we focussed on demonstrating solutions for the following themes:

- Our presentation solutions should provide support for specific content types that are difficult to present with existing tools.
- Our presentation solutions should allow presenters to involve the audience more during presentations.
- Our presentation solutions should better support the presenter during presentations.
- Our presentation solutions should help the presenter when creating content and should offer better functionality related to reuse, sharing and collaboration.

Solutions related to content visualisation are presented in Chapter 7, solutions for involving the audience are presented in Chapter 8, solutions for supporting the presenter during the presentation are presented in Chapter 9 and solutions for authoring, reuse and collaboration are presented in Chapter 10. As described by the used DSRM process model, each of these artefacts involved gathering artefact-specific requirements, a (architectural) design phase, and in some cases a software implementation. By implementing a wide range of plug-in solutions on top of the MindXpres platform

we demonstrate its flexibility and extensibility. These individual plug-in solutions further allowed us to demonstrate how the implemented plug-in solutions might address some of the shortcomings of existing presentation tools.

Step 5: Evaluation

Depending on the type of research artefact, we selected different forms of evaluation that are endorsed by the DSRM. In the case of the conceptual framework, we compare the modelled concepts to those of state-of-the-art presentation solutions to show that it can unify all next generation presentation concepts and that it is future-proof up to a certain degree. For the content model we compare the model's functionality with the solution's objectives and show that it enables the creation of new solutions that address existing shortcomings. By implementing the data model as a functional information system, we evaluate the model's feasibility and the system allowed us to simulate various scenarios that involved content reuse, sharing, collaboration and context adaptation. We evaluated the presentation engine by comparing the solution's objectives with the engine's functionality and have extensively extended the platform through the plug-in mechanism to evaluate the flexibility and extensibility. We have supervised 1 Bachelor's and 12 Master's theses on topics related to MindXpres, and having these students work with the MindXpres artefacts can be seen as another form of evaluation that resulted in feedback and design revisions. Some of the individual plug-in solutions have been evaluated for usability via SUS [42] or CSUQ [169] questionnaires, while others were evaluated based on how well they meet the requirements, or based on statements made in related work that show that our approach has proven benefits.

Step 6: Communication

Our most important contributions have been published in peer-reviewed conference proceedings and journals. This has resulted in a total of 8 related papers that describe the conceptual framework, content model, information system, presentation engine as well as various plug-in solutions built on top of the platform. Two of these papers have received a best paper award. We refer to Section 1.7 for a complete list of the related publications. By presenting these papers at conferences, we were able to get feedback and exchange ideas with international peers and domain experts. Some of the research outcomes have also been incorporated in a few of our courses to gain further feedback and discussion. For instance, the technical details of the presentation engine were discussed as part of a course on web technologies to demonstrate the capabilities of HTML5 and related technologies. The more theoretical research outcomes were discussed as part of a seminar-based course on advanced topics in information systems. This allowed further discussion on the problem and its impor-

tance, and the novelty and effectiveness of the resulting artefacts, as recommended by the DSRM. The previously mentioned Bachelor's and Master's theses further provided a new influx of opinions, ideas, feedback and requirements. Last but not least, as part of the communication step we have set up a simple website[1] in order to gain some visibility in the research community.

1.6 Contributions

- We investigated the current **shortcomings of presentation tools** and **user requirements** through a literature review, observations, a programmatic analysis of more than 12 000 PowerPoint presentations and a survey. The programmatic analysis of exiting presentation documents resulted in a large **new dataset**, a small **software framework for presentation analysis** as well as a **database with extracted information** which might help other researchers in gaining new insights. See Chapters 2 and 3.
- We revisited the concept of a presentation and defined a new **conceptual framework** that unifies existing state-of-the-art functionality and to some degree also supports future presentation functionality. See Chapter 4.
- We designed an extensible **content model** for presentation content that supports reuse, collaboration, user access management, semantics, non-linearity and context resolving. We based ourselves on the RSL hypermedia metamodel and implemented a new **model-driven hypermedia link service** providing an easily reusable foundation for any hypermedia-driven application. See Chapter 5.
- We designed and implemented a **reusable presentation prototyping engine** that allows us, as well as other researchers and developers to easily implement and evaluate currently unsupported presentation functionality. See Chapter 6.
- We used the resulting MindXpres platform to provide proof-of-concept **solutions for some existing problems** related to audience participation, presenter support, content visualisation and presentation feedback. See Chapters 7, 8, 9 and 10.

1.7 Publications

As part of this dissertation we have published some of our work as peer-reviewed conference and journal papers.

[1] `http://mindxpres.com`

Related Publications

The following list specifies papers related to this dissertation. We will reference these papers individually in later sections when we discuss the related contributions.

- R. Roels and B. Signer, 2013. "*An Extensible Presentation Tool for Flexible Human-Information Interaction*". Demo Proceedings of the 27th International BCS Human Computer Interaction Conference (BCS-HCI 2013), pages 59–60, London, UK. British Computer Society.

- R. Roels and B. Signer, 2014. "*MindXpres: An Extensible Content-Driven Cross-Media Presentation Platform*". Proceedings of the 15th International Conference on Web Information Systems Engineering (WISE 2014), pages 215–230, Thessaloniki, Greece. Springer. https://doi.org/10.1007/978-3-319-11746-1_16.

- R. Roels, C. Vermeylen, and B. Signer, 2014. "*A Unified Communication Platform for Enriching and Enhancing Presentations with Active Learning Components*". Proceedings of the 14th IEEE International Conference on Advanced Learning Technologies (ICALT 2014), pages 131–135, Athens, Greece. IEEE. https://doi.org/10.1109/ICALT.2014.46. **Best Paper Award**.

- R. Roels, P. Meștereagă and B. Signer, 2015. "*Towards Enhanced Presentation-based Teaching of Programming: An Interactive Source Code Visualisation Approach*". Proceedings of the 7th International Conference on Computer Supported Education (CSEDU 2015), pages 98–107, Lisbon, Portugal. SciTePress. https://doi.org/10.5220/0005445300980107. **Student Best Paper Award**.

- R. Roels, P. Meștereagă and B. Signer, 2015. "*An Interactive Source Code Visualisation Plug-in for the MindXpres Presentation Platform*". Communications in Computer and Information Science (CCIS), 583, pages 169–188, Springer. https://doi.org/10.1007/978-3-319-29585-5_10.

- R. Roels, Y. Baeten, and B. Signer, 2016. "*An Interactive Data Visualisation Approach for Next Generation Presentation Tools*". Proceedings of the 8th International Conference on Computer Supported Education (CSEDU 2016), pages 123–133, Rome, Italy. SciTePress. https://doi.org/10.5220/0005806401230133.

- R. Roels, Y. Baeten, and B. Signer, 2016. "*Interactive and Narrative Data Visualisation for Presentation-based Knowledge Transfer*". Communications in Computer and Information Science (CCIS), 739, pages 237–258, Springer. https://doi.org/10.1007/978-3-319-63184-4_13.

- R. Roels and B. Signer. 2019. "*A Conceptual Framework and Content Model for Next Generation Presentation Solutions*". Proceedings of the ACM on Human-Computer Interaction, 3, (EICS), 2019. https://doi.org/10.1145/3331149

Unrelated Publications

Given that this document is presented in pursuit of a PhD degree, we would also like to mention the following unrelated publication.

- R. Roels, A. De Witte, and B. Signer, 2018. "*INFEX: A Unifying Framework for Cross-Device Information Exploration and Exchange*". Proceedings of the ACM on Human-Computer Interaction, 2(EICS):2, 2018. https://doi.org/10.1145/3179427

Thesis Structure

As explained in Section 1.4, the research is composed of different phases that build on top of each other. We illustrate the different phases in Figure 1.1 and we show how they map to the different chapters of this dissertation.

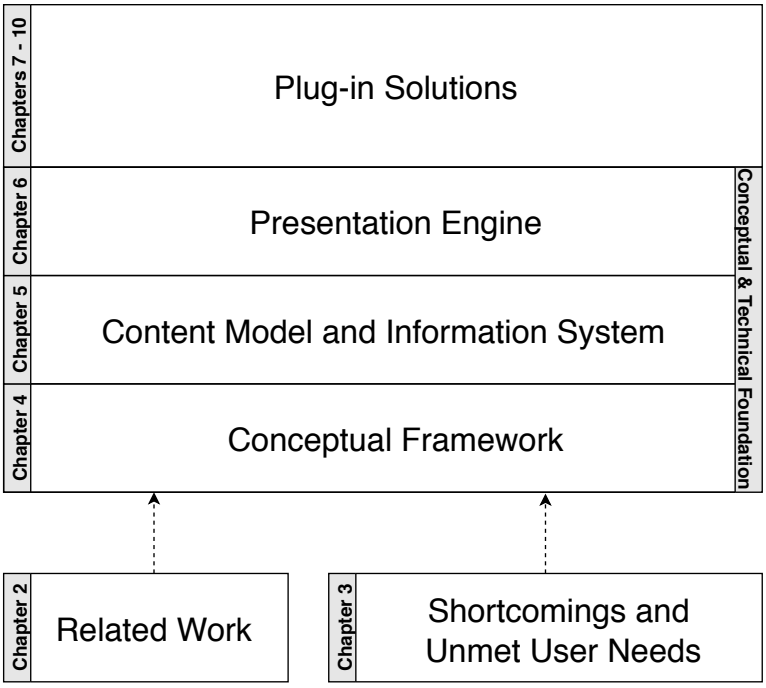


Figure 1.1: A mapping of the research goals and artefacts to the chapters of this dissertation

In Chapter 2 we start by introducing existing presentation software and we discuss their benefits and shortcomings. Later in the same chapter we move to the state-of-the-art in academic related work and offer a glimpse of the next generation presentation concepts that will be important for this dissertation later on. In Chapter 3 we discuss our literature study, observations, survey and programmatic analysis of presentation files and we end with a set of concrete shortcomings and unmet user needs. We motivate that existing shortcomings cannot easily be addressed in the existing presentation software and their related technologies. We state that we have to rebuild the conceptual and technical foundations before we can provide improved solutions, and we call this the MindXpres presentation platform. We revisit the concept of a presentation and in Chapter 4 we present a new conceptual framework for presentations, unifying existing next generation presentation concepts. The conceptual framework then leads to a new content model for presentations which is presented in Chapter 5. We discuss the design of the hypermedia-based content model as well as the implementation of a model-driven hypermedia link service that turns the model into a functional information system for presentation content. In Chapter 6 we present a new presentation engine that is capable of visualising the complex presentations supported by our underlying information system. After providing a new conceptual and technical foundation in Chapters 4, 5 and 6 we demonstrate the flexibility of the platform by introducing some of our own proof-of-concept solutions for some of the shortcomings and unmet user needs, based on the MindXpres platform. In Chapter 7 we focus on solutions for content that is difficult or time-consuming to present with existing presentation software. In Chapter 8 we present some solutions for better involving the audience during presentations. In Chapter 9 we discuss solutions for supporting presenters during presentations. The last category of solutions, solutions for helping presenters create good content, is presented in Chapter 10. We end the dissertation by answering the research questions presented earlier in Chapter 11, which serves as a summary of our work. This final chapter also includes a critical discussion, covering limitations and future work, as well as some final conclusions.

Chapter 2

Background and Related Work

In this chapter we introduce some of the general concepts and related work of presentation tools. In order to highlight some of the current limitations of slide-based presentation tools, we briefly detail the history of digital presentation tools in Section 2.1. In Section 2.2 we discuss the commercial presentation tools that currently dominate the market. We then present some benefits and shortcomings of these presentations tools in Section 2.3 and mention the relevant studies to support these claims. Last but not least, we discuss the state-of-the-art of presentation solutions in Section 2.4 and introduce various next generation presentation concepts. Note that additional related work is also introduced in other chapters, as many of our research artefacts have some additional topic-specific related work.

2.1 History of Presentation Software

PowerPoint was originally developed as a digital tool for designing transparencies for overhead projectors and later for designing physical slides for slide projectors. Before computer-supported slide authoring, creating these physical slides was a tedious and costly process, especially if multiple iterations had to be made. Software such as PowerPoint allowed presenters to create their slides faster, and only once they were fully satisfied with their content in the software they would get their slides developed for overhead or photographic slide projectors. The first product proposal for PowerPoint was submitted by Robert Gaskins in 1984, at a company called Forethought, Inc (not yet Microsoft) [99]. PowerPoint was first commercially released in 1987 and in the same year Forethought, Inc was acquired by Microsoft. In 1990 the first version of Microsoft PowerPoint was released.

Over time PowerPoint evolved. PowerPoint 1.0 only supported black and white overhead slides, PowerPoint 2.0 added support for 35mm colour slides and PowerPoint 3.0 introduced video effects and allowed presentations to be shown directly from a computer, for instance via video projection [100, 211]. PowerPoint was born as an emulator for physical slides and the software has changed a lot. However, the characteristics of physical slides are still the foundation of PowerPoint and even now PowerPoint can still be called an emulator of physical slides. For instance, a presentation is still a linear sequence of spatially-bounded slides. Content is relatively static and almost nothing has chanced to the way we interact with, and give our presentations when compared to the old 35mm slide projectors. This is a remarkable observation considering that the adoption and capabilities of digital devices have increased massively over the same time period. Some additional observations can be made when we look at the evolution towards digital slideware, as illustrated in Figure 2.1. One could argue that some functionality was even lost when teachers moved from simple blackboards to overhead projectors, slide projectors and eventually digital slideware. For instance, a blackboard allows presenters to dynamically make changes to their content and the board provides a larger overview making it easier to relate concepts. When the overhead projector was introduced content became spatially restricted but it still allowed presenters to make annotations with markers and the transparencies could be traversed in any order. Although slide projectors introduced the ability to display colour and photos, the ability to annotate content was lost and slides had to be traversed as a linear sequence. As digital slideware mimics the characteristics of physical slides, including their limitations, it becomes clear that some benefits of other media have been lost over time.

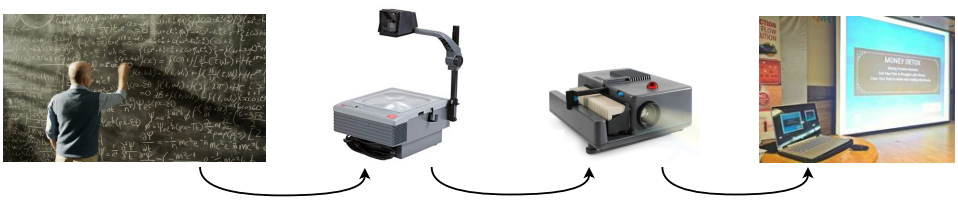


Figure 2.1: The evolution towards digital slideware

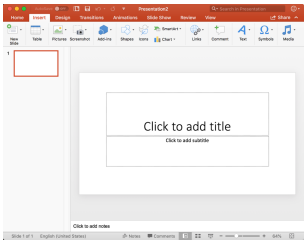
2.2 Commonly Used Presentation Software

Before introducing the academic related work in Chapter 2, we briefly discuss the more widely used presentation software in this section.

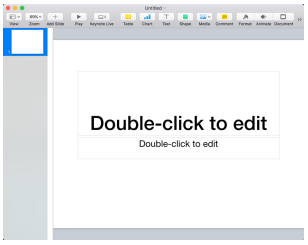
2.2.1 Commercial Presentation Software

Microsoft PowerPoint's market share is estimated to be around 95% [311, 270], last verified in 2012. At the time of writing this number might have diminished slightly but there is no doubt that PowerPoint is still the de facto standard for presentation tools. In terms of functionality, users can create a linear sequence of digital slides. The content that can be added to slides includes text, images, video, audio, mathematical formulas, diagrams and charts. Animations and transition effects can be added and different visual themes can be applied to the content. During the presentation, PowerPoint's presenter view gives presenters some additional information not seen by the audience, for instance the time, the presenter notes and the next slide.

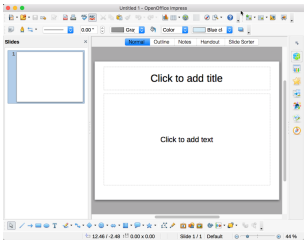
Although PowerPoint is the most popular tool, there are some commonly known alternatives with similar functionality. Apple's office suite iWork includes the Keynote presentation tool, and the LibreOffice and Apache OpenOffice suites come with the Impress presentation tool. Although each of these tools has some unique characteristics, the core functionality and features are comparable to those of PowerPoint. Each of these tools is based around the concept of slides, and is therefore often referred to as *slideware*. As shown in Figure 2.2, the user interfaces look relatively similar. It is also interesting to point out that although these interfaces offer a lot of buttons and options, almost all of them are related to visual details (e.g. fonts, colours or animations) and little to no space is dedicated to functionality related to creating good content.



(a) Microsoft PowerPoint



(b) Apple Keynote



(c) OpenOffice Impress

Figure 2.2: Some of the most popular slide-based presentation tools

There are also web-based slideware tools. The most well-known is Google Slides[1] which differentiates itself from other slideware tools with its collaboration and sharing functionality. Note that Microsoft now also offers a web-based version of its office suite. We would further like to mention Slides[2] as a web-based slideware tool, which has been growing steadily in popularity over the last few years. Slides mentions

[1] `https://www.google.com/slides/about`
[2] `https://slides.com`

collaboration as a main selling point, but further offers facilities for developers to use CSS or JavaScript in presentations. Last but not least, Prezi[1] is one of the few commercial presentation tools that break with the classic slide-based paradigms and instead focusses on canvas-based presentations. By using panning and zooming, presenters focus on relevant parts of the canvas which results in a more dynamic way of jumping between content. Zoomable user interfaces such as those utilised by Prezi are discussed later in more detail. There are of course other commercial tools and platforms for creating digital presentations, but the ones mentioned above cover at least 99% of the market share according to some market studies [270].

2.2.2 Markup Languages and Software Frameworks

Next to the presentation tools with graphical editors, there are also some software frameworks and markup languages that are used to create slide-based presentations. For instance, there is a non-negligible group of presenters that prefers to create their slides in LaTeX. More specifically, the Beamer[2] package offers document classes and macros specifically for creating slide-based presentations. These presentations are most often compiled to PDF documents and presented with PDF viewer software.

Many of the other frameworks are based on web technologies such as (X)HTML and JavaScript. Examples include W3C's Slidy [217], S5[3], deck.js[4], bespoke.js[5], reveal.js[6] and impress.js[7]. Most of these frameworks allow users to build web-based presentations with features similar to those of PowerPoint presentations. However, it is worth noting that impress.js can also use a zoomable canvas layout. Furthermore, reveal.js sticks more closely to the classic slide paradigm, but slides can be laid out in a 2-dimensional grid allowing navigation in four directions instead of simply forwards and backwards.

2.3 The Pros and Cons of Slideware

From our personal experience, we noticed that a lot of people have strong opinions in favour as well as against PowerPoint and related tools. As an example of the polarity in opinions, Edward Tufte's well-known essay against PowerPoint [278] is publicly opposed by the equally well-known Jean-luc Doumont [71]. Both are considered as authorities in the domains of information visualisation and oral presentations, and both

[1] `https://prezi.com`
[2] `https://ctan.org/pkg/beamer`
[3] `https://meyerweb.com/eric/tools/s5`
[4] `http://imakewebthings.com/deck.js`
[5] `http://markdalgleish.com/projects/bespoke.js`
[6] `https://revealjs.com`
[7] `https://impress.js.org`

parties present strong arguments to support their claims. For this reason we believe it is important to introduce some related academic studies and to discuss scientific findings in order to better understand slideware and to avoid any misconceptions. Although most of the following experiments were conducted in an academic setting, many of the findings apply to other contexts as well.

2.3.1 Effect of Slideware on Learning Performance

One of the most surprising findings is that most studies show that the use of digital presentations in education has no effect on the learning performance of students [267, 13, 264, 265, 62, 218, 15, 29, 138, 167]. In other words, in general there is no increase but also no decrease in performance when comparing classes with digital presentations with classes that are taught using only traditional methods such as whiteboards. Typically these experiments use exam grades as a metric and show that there is no significant difference in grades between the two teaching methods. We did encounter two studies where a significant improvement of grades was measured [183, 179] as well as a study where a significant decrease in information retention was observed [238]. However, the grade improvements in [179] can be attributed to the fact that digital presentations were a completely new experience with a high novelty factor for students in 1995, and [183] tested the combined introduction of internet-based class notes and PowerPoint presentations, making it unclear which of the two caused an increase in grades.

In general, academic work seems to agree that there is no benefit in terms of learning performance, but there is evidence that digital presentations do *facilitate learning*. In other words, some studies suggest that digital presentations can be processed more effectively by students and in some cases less time and effort is required to obtain the same level of understanding compared to students who were taught the same content via other media. For instance, Szabo and Hastings [267] claim that PowerPoint lectures can benefit recall from memory. Another factor that facilitates learning is that presentations support more learning styles compared to whiteboard teaching [62, 138, 167]. The benefits of appealing to different learning styles in classrooms had already been demonstrated before the existence of digital presentation tools [53] and digital slides can help achieve similar benefits. Animations, charts and diagrams appeal to visual learners and similarly the abstractions, symbols, short sentences and structure enforced by PowerPoint may help logical learners to process content faster. According to Dils [69], the verbal, kinaesthetic and visual learning styles are the styles that can be most effectively addressed by digital slides.

There are of course other characteristics of digital presentations that facilitate learning. In the following sections we discuss additional factors that influence the effec-

tiveness of digital presentations and additional characteristics that facilitate learning will be mentioned.

2.3.2 Effect of Slideware on Attitude and Perception

There are also studies that investigate how students perceive digital presentations and how they feel about them compared to more traditional lectures. In general, the responses are overwhelmingly positive and studies show that digital presentation tools improve attitude, attractiveness and entertainment levels of lectures [267, 13, 265, 264, 62, 179, 238, 138]. In fact, several of the studies mention that students outright preferred the lectures with digital presentations [179, 238, 264, 138]. One of the most commonly mentioned reasons is that students perceive the course content as more organised, structured and summarised. As such, students *believe* the course material is easier to recall and understand (although this is not necessarily the case). In some cases students state they learned more from digital slides [167], even though the previously cited studies on learning performance state otherwise. Some studies also report an increase in teacher likeability as a direct result of digital presentation tools [267, 138, 13]. Other studies claim that digital presentation tools give students the impression that the teachers are well-prepared which enhances their credibility [15].

2.3.3 Effect of Slideware on Attendance

It is sometimes thought that if PowerPoint presentations are used and made available afterwards, students are less inclined to come to the lectures to take notes. However, there seems to be a lack of consensus on this topic. Young [303] reported a significant drop in attendance when he made the PowerPoint material available. Two other studies report no difference in attendance when comparing digital presentations with traditional lectures [265, 29]. We also encountered two studies that reported an increase in attendance [267, 138, 13]. Based on the literature we found, there is no clear answer to this question and the same conclusion is also made by El Khoury [82].

2.3.4 Effect of Slideware on Presentation Flow

We previously mentioned how digital presentation tools can be used to bring structure and organisation to content. These benefits also extend to when the presentation is delivered and studies report more efficient time management [62, 183]. For instance, it is clearer for the audience and presenter how much content is left and to manage the available time. The fact that the presentation content is prepared beforehand also makes sure that less time is wasted on drawing and writing by the presenter during the presentation. Complex visualisations can be prepared beforehand so that the limited lecture time can be spent more effectively (e.g. on the oral explanation).

If the slides are made available afterwards, the audience also has to take fewer notes. Studies have shown that separating note taking from listening can lead to better retention [6]. Due to more efficient time management, more material can be covered or it can be covered in greater detail.

One of the biggest critiques that come forward in studies is how digital presentation tools can change the relation between the audience and the presenter [167]. Digital presentations predefine the content that will be discussed and encourage a uni-directional flow of information from the presenter to the audience. Simons states "*it imposes an authoritarian presenter/audience relationship rather than facilitating a give-and-take exchange of ideas and information*" [253] and Tufte describes it as "*the pushy PP style imposes itself on the audience and, at times, seeks to set up a dominance relationship between speaker and audience*" [278]. As Hlynka et al. describe it, spontaneity is minimised and PowerPoint "*entrenches our sense of 'presentation' within a one-way delivery metaphor*" [122]. In other words, students are "*passively engaged instead of actively engaged*" [56] and students might sit passively through the session as they are never challenged to think. Farkas states that the stop-and-go rhythm of a linear slide sequence might reduce dynamism and reduce the audience's engagement [86]. Of course this depends on the context and the presenter, and as we show later, there are solutions for increasing audience involvement during presentations. However, literature agrees that this uni-directional form of communication is almost encouraged by digital presentations and presenters should take special care to avoid this common pitfall.

2.3.5 Effect of Slideware on Presentation Content

Earlier we have mentioned that digital presentation tools originated as emulators for their physical counterparts and digital slides inherit a lot of the characteristics of physical slides. First of all, slides are limited in space and to make matters worse, projectors often have a limited resolution. It is argued that for this reason content is often over-simplified. As a consequence, details and context are lost, leading to issues such as over-generalisation, unfounded claims and misleading conclusions [278, 253, 122, 86]. Because of the limited space, larger pieces of content have to be spread out across multiple slides and related information becomes scattered and out of view. Next, modern presentation tools still cling to the idea that slides have to be traversed as a linear sequence, just like physical dia slides. This forces presenters to flatten a potentially complex idea into a one-dimensional structure of slides with hierarchical bullets, but as famously quoted by Vannevar Bush, "*The human mind does not work that way. It operates by association.*" [44]. As a consequence, a complex graph-like data structure of information has to be flattened and context and relationships are inevitably lost [278]. The issue is also brought up by Norvig, who states "*PowerPoint*

makes it harder to have an open exchange between presenter and audience, to convey ideas that do not neatly fit into outline format" [208].

In Figure 2.1 we have shown some presentation tool interfaces and we pointed out how most of the offered functionality is related to visual details (e.g. fonts, colours and layout). Tufte argues that PowerPoint values form over content [278] and that most features just introduce additional "noise" that takes attention away from the message that the presenter wants to convey. Users often feel that they should use all the features related to aesthetics, but Blokzijl and Naeff claim that students dislike sound effects and animations, and that students prefer slides with large fonts, consistent design, solid quiet backgrounds, no busy colours and colour combinations with significant contrast [35]. This implies that in some cases "less is more". Additionally, it has been shown that irrelevant content such as irrelevant sounds [195], irrelevant pictures such as clipart [24] or interesting but extraneous text harms performance [243]. However, these irrelevant pictures can enhance entertainment and interest levels [13]. The use of relevant graphics however, can help with difficult, complex or abstract concepts [24, 142].

As another characteristic inherited from physical slides, we would like to point out that presentation content is still relatively static. Presentation tools started with support for basic content such as text and images, but even now almost 30 years later the most dynamic additions we have seen are audio, video, and animations. Arguably, audio, video and animations were also used before digital presentation tools so it was not even a new feature. Earlier, when we discussed the evolution towards slideware we also pointed out that slideware does not have all the affordances of earlier media. One of these affordances is the ability to draw on slides, as was possible with transparencies. Considering that computers have come a long way in the same time frame and are capable of so much more, it is surprising how presentation tools are still relatively limited when it comes to dynamic content. It has also been shown that dynamic content can be immensely beneficial for data-driven storytelling [244] and for strengthening students' mental model of complex concepts [125]. However, current tools do not offer the necessary support for creating dynamic and interactive content, so it is often underutilised. More related work and background on dynamic content is provided in Section 7.6, where a possible solution is discussed.

Finally, we would like to discuss the consequences of how PowerPoint content is stored. A PowerPoint presentation is stored as a single monolithic file. In order to reuse content it has to be copied and pasted into a new presentation. Next to the redundant storage, this causes the issue that the same content is now part of multiple presentations, and if it has to be changed, it has to be changed in multiple places. Microsoft confirms many of these issues in a related publication [74]. To make matters worse, presenters often have to adapt a presentation to a specific audience or

context which makes it difficult to avoid duplicate content. For this reason, knowledge workers often have to maintain collections of hundreds of presentations [74]. The issues related to content reuse, sharing and collaboration will be further discussed in more detail in later chapters.

2.3.6 Effect of Slideware on Presenter Skills

The digital presentation content is of course not the only factor that influences the learning effectiveness of a presentation. The way the content is presented and explained plays an important role. Effective delivery of the message depends on factors such as body language [247, 205, 156], the tone of the voice [156, 165, 161, 205] and interaction with the audience [247, 294]. The use of PowerPoint has no effect on body language or vocal abilities, but as mentioned before, PowerPoint may decrease audience interaction due to the uni-directional flow of information.

One recurring concern that we encountered is that some people believe digital presentation tools make presenters lazy and less skilled. The reasons that are given are related to what we discussed earlier; content has to be summarised and cut up to fit on a slide. Therefore it is not always expected by the public that a presentation goes into a lot of detail. Some believe that this, together with entertaining animations and images, can help to hide incompetence or shortcomings in the reasoning [278, 95, 182]. It would allow presenters to get away with being less knowledgeable on the presented topic, or at least to be less prepared. Another concern is that since presenters know they can always fall back on simply reading the slides aloud, less effort is spent on preparing the oral delivery. These concerns are plausible, but to the best of our knowledge there is no evidence showing that the use of digital presentation tools makes people lazy or less skilled. We also argue that the same concerns can be raised over other media and content delivery methods such as textbooks, videos or even whiteboard lectures. Additionally, Farkas counters these concerns by stating that these claims wrongfully assume that the majority of presenters are simply reading bullet points to the audience, and that they ignore the fact that many presenters can give a proper oral presentation based on bullet points [85].

2.4 State of the Art of Presentation Solutions

So far we have focussed on discussing the effects digital presentation tools have on knowledge learning and we pointed out various benefits and shortcomings that are backed by studies. In this section we focus on presenting academic work that addresses some of the previously mentioned issues.

2.4.1 Zoomable User Interfaces (ZUIs)

We have previously discussed how digital presentation tools emulate physical slides, which is why slides are still traversed in linear order. However, it has been shown that this approach may not be optimal for knowledge transfer. Humans have limited working memory, as illustrated by the well-known Miller's law claiming that humans can only hold 7 plus or minus 2 objects in their working memory [191]. The slide paradigm forces content to be split and spread across multiple slides causing a lack of overview. As workarounds we see for instance the use of multiple projection areas [164] or the use of tiling to fit as many pictures as possible on the same slide [50]. Even with these somewhat impractical workarounds, audience members are still required to remember and relate content across several slides. Depending on the complexity of the content this might require more than what is offered by the average working memory. Presenters might add reminders to slides or go back and forth between slides, but in the end these are only workarounds for a more fundamental problem.

Zoomable User Interfaces (ZUIs) can be used to avoid these issues. ZUIs allow content to be laid out in more than one dimension and they can use spatial cues to create implicit relationships between content. Gestalt theory [296, 152, 153] shows that humans can naturally interpret spatial cues. For instance, objects that are similar in shape or colour are automatically interpreted as having a similar meaning, or proximity can be used to show that specific objects belong together. Nesting of objects can also be used to encode hierarchical relationships. By zooming out, viewers can be given an overview of the content, and by zooming in more details can be made visible. Research indicates that if verbal information is combined with spatial information (such as an oral presentation based on a ZUI), the information is stored separately in human memory and as a result more information is retained [107]. There are also indications that presenters that score high on spatial ability prefer canvas-based presentations over slide-based presentations [172].

The most well-known application of a ZUI in the context of presentations is Prezi, but there has also been a lot of academic work on the subject. Related work goes back to 1993 (long before Prezi), with projects such as Pad [213] and Pad++ [28]. Later the subject was further explored in CounterPoint [107], Slithy [311], Fly [171], IdeaMaché [174] and The Organic Presentation Tool [282]. Although not strictly a presentation tool, the topic is also discussed in detail in iMapping [116]. Further, Microsoft has experimented with the concept of ZUIs with pptPlex[1], an Office Labs experiment. A more flexible navigation, improved overview and the use of spatial cues are most often cited as motivation for zoomable user interfaces. CounterPoint also points out that since content can be revisited more easily, ZUIs can also reduce

[1] `https://www.microsoft.com/en-us/download/details.aspx?id=28558`

the need for duplicate content within the same presentation [107]. SlideSpace [81] investigates the authoring of canvas-based presentation using hierarchical outlines. The effect of ZUIs on learning performance has been investigated in studies such as [171, 36, 107, 51]. Bederson, the author of Pad++, later also published a thorough review on the cognitive benefits of ZUIs in general [27]. The findings from these studies are more or less consistent. They found that there is no significant difference in fact retention or the understanding of relations between content. They also showed that participants were more engaged, had a better orientation of the talk progression, and the structure of the presentation was easier to understand. Recent work further shows that it is important for the learning process that content is well integrated in the greater whole, both structurally and visually [112], which is something a ZUI can help with. A more recent study that focusses on Prezi in particular, with over a thousand participants, further confirms that the use of a ZUI does not result in a better recall or comprehension of the content [197]. However, similar to the other studies, the authors found that participants evaluated Prezi as more organised, more engaging, more persuasive and more effective.

2.4.2 Non-linear Navigation

Traditional slideshows enforce linear navigation, meaning that slides have to be traversed in a predefined order, one by one. Farkas states that because of this linearity and the limited interaction possibilities for navigation, presenters might be reluctant to skip slides or to make jumps in the sequence if the need arises [86]. ZUIs allow the navigation to be more flexible as there is not necessarily an order in the content. A navigational order can still be predefined in a ZUI-based presentation, but ZUIs are more flexible when it comes to improvisation [107]. The ZUI-based Organic Presentation Tool [282] provides multi-gestures that make it easier to leave the predefined path and to return later. ZUIs are of course not the only way to achieve non-linear navigation. As a first example, reveal.js allows slides to be laid out in a 2D grid-like structure, allowing navigation in four directions as shown in Figure 2.3. Depending on the audience or the context, different side-paths may be followed.

Non-linear navigation can also be achieved with classic slide sequences. One way to do this is to provide the presenter with an interface that gives an overview of all content and a way to jump to a specific slide in the overview. For instance, Palette [201] provides presenters a deck of physical paper cards and by scanning the barcodes on them, the presentation jumps to the corresponding slide. PaperPoint [251] offers an overview of multiple slides on a single sheet of paper and a digital pen is used to jump to specific slides. Customizable Presentations [196] allows presenters to predefine nested sub-paths over content, and offers a graphical overview for the presenter to adapt the presentation on demand. MultiPresenter [164] leverages a second presen-

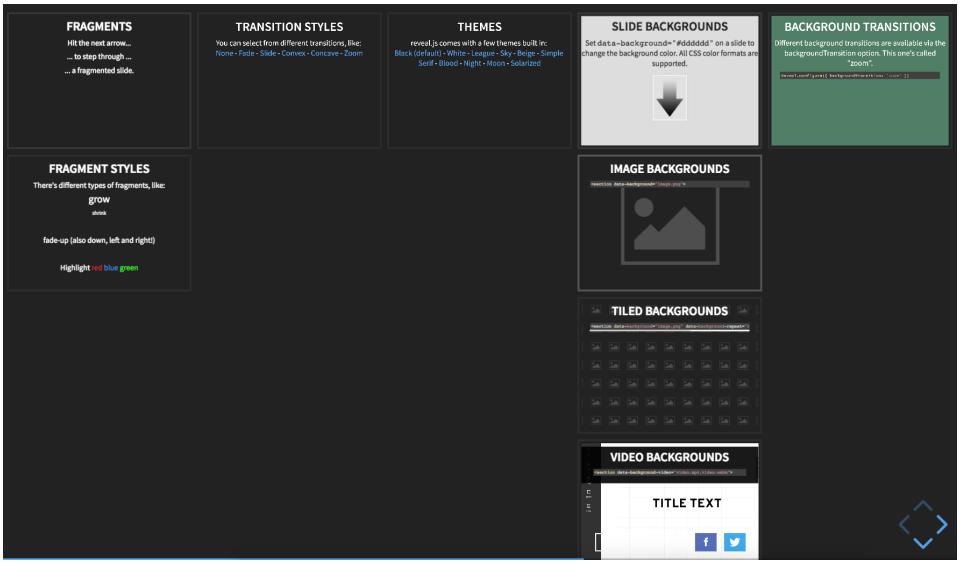


Figure 2.3: Overview of a reveal.js presentation
Source: `https://revealjs.com`

tation display for the audience and a graphical overview for the presenter to allow any content to be shown next to each other on demand. Other tools build on concepts from hypermedia to achieve non-linear navigation. NextSlidePlease [259] can automatically adapt navigational paths based on predefined weighted links between slides, or factors such as remaining presentation time. HyperSlides [80] offers a markup language for hierarchically structured content which is automatically transformed into hyperlinked slides with multiple navigational paths. As motivation for these tools, the authors mention reduced content duplication, showing relations between content and the ability to adapt the presentation to the audience or to other factors.

2.4.3 Content Reuse and Collaboration

As mentioned before, when users want to reuse content in PowerPoint, the content has to be duplicated which causes data synchronisation issues. Microsoft Research acknowledges this issue and investigated the use of a technical solution for helping users to merge slides and presentations [74]. Office SharePoint 2007, 2010 and 2013 used to offer Slide Libraries, but this feature is now discontinued. Slide Libraries allowed users to store slides in a shared repository. However, users were still required to manually upload, re-download and merge slides. SliDL [46] proposes a system where slides are stored in a central repository, where they are automatically processed to remove duplicates and to extract metadata such as keywords. This information,

together with thumbnail overviews would allow users to find relevant content more easily. Note that SliDL focusses on reuse at the level of entire slides, and not on a smaller granularity such as single images or pieces of text. SidePoint [176] extends the idea and offers a peripheral panel in PowerPoint that suggests related pieces of content during authoring, which can for instance be smaller pieces of text. The problem of fragmented and duplicate content is further detailed by Zhang et al. [308] and a method for detecting reused textual and visual content is described. The same authors also describe a system for querying existing presentations to (re)find specific text fragments, images or diagrams [309].

We would also like to mention the subject of collaborative presentation authoring, as it is related to sharing and reusing content and is also covered by our solutions which are presented later. Google Slides is the most well-known example of a collaborative presentation authoring tool, but the subject has also been touched by academic work such as Webstrates [149] and SlideWiki [145].

2.4.4 Semantics

Although not a feature seen in typical presentation solutions, it can be beneficial to give semantics to content to make it machine understandable, for instance by semantic annotation as investigated in SidePoint [176], iMapping [116] or CPoint [154]. For example, a specific text fragment might be tagged as a definition and associated with the concept it defines. This semantic data could then be used to offer additional functionality during authoring or during the presentation itself. For instance, semantic annotations can be used to find reusable content [119, 246, 140], or other related work explores the (semi-)automated assembly of presentations from existing semantically enhanced content [22, 184, 286, 175]. Other uses of the semantic metadata include content recommender systems [176] or semantic querying. Content reuse and semantic content is later discussed in more detail in Chapter 5 where we present a new content model that takes semantics into account.

2.4.5 Dynamic and Interactive Content

When we investigated the topic of dynamic and interactive presentations, we noticed that a lot of the popular writing seems to assume that a dynamic presentation implies the use of slide transitions, text animations, visual effects or ZUIs. Usually these enhancements are of purely aesthetic nature and have no measurable impact. However, in the academic context the term is usually interpreted as functionality that makes content less static for the purpose of enhancing knowledge transfer. This could, for instance, involve the use of embedded simulations or visualisations that can be influenced by the presenter or audience. In Section 2.3.5 we have already stressed

the importance of how dynamic content can aid storytelling or the development of a more complete mental model. However, it seems that current presentation tools offer little to no support to create such content. Presenters are required to use external tools or have to painstakingly set up a series of PowerPoint animations to give the appearance of dynamic content. There are various studies that detail the science behind the benefits of dynamic content, which we will present as part of our solutions in Chapter 7. However, in terms of related work that provides a technical solution for presentations, little could be found. Slithy [311] is one of the academic solutions we found and provides a script-based system for defining animations for presentations. Tiling Slideshow [50] automatically clusters related images and visualises them in a tile-based visualisation synchronised to music. For us, the lack of solutions in this category indicates that there is still room for improvement, particularly in the form of more general support for helping presenters to create dynamic content.

2.4.6 Audience Involvement

As discussed before, one recurring critique on digital presentation tools is that it encourages a uni-directional flow of information; the presenter talks and audience members just listen for long periods of time. Of course this is an avoidable issue which depends heavily on the presenter and the presentation style. Nevertheless, it has been shown that engaging the audience can increase the effectiveness of a presentation. In education the concept of actively engaging the students is coined "active learning" and in general, the many studies on the subject seem to agree on the benefits of this style of teaching. For instance, students are more focused, participate more and perform better, as shown by Kay's literature review of 67 studies [141]. Active learning can be encouraged with for instance, quizzes, discussions, exercises, simulations or games. With recent advances and improved access to technology, we have seen some technical solutions to help involve the audience during presentations. The most well-known example is the use of the clicker systems shown in Figure 2.4, where audience members can use remote-like devices to participate in quizzes and polls. However, there are also cheaper solutions that, for instance, use machine vision to detect and count paper cards held up by students in the audience [57]. Similarly, Zarraonandia et al. describe a system that uses augmented reality to visualise individual audience responses as an overlay on top of their paper-based markers [306].

As network connectivity is becoming more wide-spread, we also see solutions based on mobile devices and computers. Classroom technology has also been built around other hardware such as digital pens[1] or graphing calculators[2]. Microsoft Research also investigated the use of smartphones for providing binary feedback (e.g. "yes" or "no")

[1] `https://www.bic-education.com`
[2] `https://education.ti.com`

to questions during presentations [269]. These solutions help teachers and businesses to reap the benefits of active participation but we also see new challenges. For instance, we see a lack of integration in the presentation tools, activities are limited and tied to a specific vendor's hardware and the often expensive hardware is not interoperable. As we will discuss in more detail in Chapter 8, we see another opportunity to improve our presentation tools and to provide a more generic solution for these issues.

(a) Boxlight MimioVote Assessment System

(b) TurningPoint ResponseCard

Figure 2.4: Collecting responses from the audience with clicker systems Sources: https://mimio.boxlight.com and https://www.turningtechnologies.com

2.4.7 Presentation Skills

We have previously mentioned how a presenter's presentation skills might impact a presentation. It is said that no one is born as a bad or a good presenter, and presentation skills, like many other skills, improve with practice and experience [161]. There have been various academic tools that attempt to aid presenters with their presentation skills. One category of solutions comes in the form of automatic feedback systems that provide feedback during the presentation (online feedback) or after the presentation (offline feedback). Presentation Sensei [161] uses a microphone and camera to monitor factors such as speech rate, the use of fillers and the presenter's orientation and warns the presenter in real-time if any of the measurements go outside of predefined thresholds. Nguyen proposes a feedback system for non-verbal expressions such as gestures and posture using the Microsoft Kinect for body tracking. Kopf et al. [156] take a similar approach but also provide feedback on verbal and time management factors. Rhema [268] focusses on feedback on verbal skills and uses Google Glass for online feedback. Similar to Presentation Sensei, [96], [242] and [298] take a multimodal approach for gathering data and offer feedback on eye contact, posture, gestures, use of fillers or speech rate. One shortcoming of these systems is that thresholds are needed to provide feedback. For instance, at what

point is speech too fast, or what is the optimal amount of body movement? These thresholds are often hardcoded, or the users have to draw their own conclusions from the data that is provided. The main issue is that these thresholds depend on many factors such as the language that is used, the local culture or the context of the presentation. For instance, the speech rate for the Japanese language is much higher than the speech rate in English, or eye contact and excessive movement can be perceived differently in different cultures. We also observed that these systems mainly focus on the presenter but the presentation content itself it not really considered for feedback other than for time management purposes. It can be argued that proper structuring, layout, content density and use of aesthetics are also skills that can be classified as presentation skills, but to the best of our knowledge these are not considered by automatic feedback systems.

Next to automatic feedback systems there are also other approaches to aid the presenter to give a better presentation. DynamicDuo [275] aims to reduce anxiety by providing the presenter with a virtual co-presenter that can take over parts of the presentation. PitchPerfect [276] is a tool that allows presenters to plan the presentation flow and speech timings beforehand and rehearse them. Similarly, TurningPoint [216] helps presenters to plan the narrative of their presentation to create a more fluent story based on the presentation content. SketchStory [166] and Academic Presenter [17] provide authoring support for creating fluent animated visualisations to support data-driven storytelling. TalkZones [236] is a solution for mobile devices that focusses on setting, refining and following presentation time targets.

2.5 Conclusions Drawn from Related Work

From the presented studies we can draw a number of initial conclusions. First of all, it is clear that the use of digital presentation tools does not automatically result in improved learning performance. However, there is evidence that digital presentations can improve *learning effectiveness*. The most commonly mentioned reason for this is that digital presentations can appeal to more learning styles than for instance a textbook or a whiteboard lecture. In general, digital presentations are experienced very positively by students and digital presentations are often preferred over other teaching styles. For some findings we also presented studies that came to contradicting conclusions. When these contradicting results are discussed in literature we noticed that the same obvious but consequential explanation is given. Namely, digital slides *can* be used for more effective learning when used correctly, but blindly using them does not automatically provide the presented benefits. The effectiveness is determined by how the tool is used [267, 62], and more concretely by factors such as how the presentation is structured [122], the quality and relevance of content [24], the delivery

style [238], the topic and context [82, 238], and how the presenter is perceived [56]. Hlynka nicely summarises this as "*the power of PowerPoint lies in the hands of the presenter*" [122]. We would like to argue that current presentation tools offer users a lot of functionality, but the tools do not offer guidance in applying them in beneficial ways. The amount of freedom that is given can even be counter-productive, as users seem to spend "*disproportionate amounts of time agonizing over the choice of design template, colour scheme, page layout, and mode of slide transition*" [56]. Some academic solutions aim to provide presenters with feedback to improve their presentation skills. However, these tools focus mainly on improving public speaking skills and do not provide feedback on the content itself.

We have also shown how current presentation tools have changed very little since their birth as slide emulators. Content is still relatively static, space is limited and slideshows encourage a uni-directional flow of information. On the other hand, we have shown how storytelling, audience interaction and dynamic and interactive visualisation can help to improve students' mental models and aid knowledge transfer. Current computers are more than capable of supporting these features and learning styles could be addressed more effectively. However, the current presentation tools do not facilitate the creation of such complex content. Due to technical as well as ideological restrictions, attempted solutions cannot always reach their full potential and do not integrate well in the presentation process.

To summarise, academic work focusses on some of the individual shortcomings but solutions are often built based on the same assumptions and ideologies that are enforced by tools such as PowerPoint. Existing solutions accept the limitations caused by the enforced paradigms and provide separate stand-alone solutions that do not integrate well in the presentation process and that are not interoperable with other solutions. We claim that by letting go of some of the established paradigms, more effective solutions could be applied at the core of the presentation tools themselves. We see potential in eliminating some technical as well as some ideological restrictions. By addressing some of the issues at the root, we believe that interoperable and more effective solutions can be built that integrate seamlessly into a coherent software package for end-users.

Part II

Identifying Shortcomings and Unmet User Needs

Chapter 3

User Needs, Preferences and Requirements

In Chapter 2 we have highlighted the broader themes that are covered by modern state-of-the-art presentation solutions. As a next step, we investigate how tools such as PowerPoint are currently used and what users require from presentation tools. We discuss both existing features as well as features that may be needed but that are not supported in existing tools. In Section 3.1 we start by discussing existing academic work that focusses on user needs, preferences and requirements. We also performed a series of observations in order to determine how presentation tools are used, and the results of the observations are presented in Section 3.2. To further determine how presentation tools are used, we programmatically performed an analysis of presentation content on more than 12 000 randomly sampled PowerPoint presentations. The analysis and the results are discussed in Section 3.3. The findings from this chapter and Chapter 2 are used to identify potential shortcomings as well as new user needs. In order to verify some of the shortcomings and needs that have not been documented we performed a large-scale survey. Section 3.4 details the survey and the results. We end this chapter with a final summary on user needs, preferences and requirements in Section 3.5.

3.1 Literature Review

We start by listing some of needs and requirements that can be derived from existing literature. We focus mainly on some of the recurring themes that have not yet been addressed and that have helped us to define requirements for our later solutions.

3.1.1 Need for Skilled Presenters and Good Presentations

Given that presentations are commonly used for knowledge transfer and customer persuasion in business settings, it is unsurprising that companies value good presentation skills. Of course the same can be said about academia where knowledge transfer is the main goal. Surveys on the topic indicate not only the need for oral presentations in general but also reveal a demand for stronger presentation skills [52, 139, 260]. These skills include good verbal communication but also the ability to use tools such as PowerPoint and being able to author good presentations. One study demonstrates that many people in industry cannot even identify the characteristics of a good presentation, and participants were more than twice as likely to grade a poor presentation as "excellent" [109]. On the other hand, a survey commissioned by Prezi highlights that nearly 70% of the 2031 participants see presentations as critical to their success at work[1]. In a large-scale study by Thielsch et al. [270] one open feedback question was related to the audience experience and 48% of the responses mentioned the performance of the presenter as a point of improvement. Some participants further mention a need for improved presenter support such as better presenter views.

3.1.2 Need for Content Sharing, Reuse and Collaboration

We have already shown some solutions that involve sharing, reuse and collaboration for the authoring process in Section 2.4.3. In this section we further touch the subject but from the perspective of user needs and requirements, introducing some additional studies. This will help us to show that existing solutions may not be sufficient and that there is still much room for improvement.

In the context of learning in general, projects such as SCORM [132] motivate the need for reusable learning packages. Verbert et al. [286] provide references to show that reuse not only saves time and money [73, 221], but also enhances the quality of digital learning experiences, resulting in efficient, economic and effective learning [77]. As an alternative medium for learning we also believe that these arguments apply to presentations and presentation tools. The need for reuse is of course not limited to education and also applies to industry. It is said that knowledge workers often maintain hundreds of presentations [40] which comes with various challenges but little tool support. One major contribution on this subject is a 2011 study by Mejova et al. [188] which focusses on investigating the reuse of presentation content in a very large organisation (IBM). As one part of the study, the authors analysed the company's massive content archive and scanned all presentation content for reuse and duplication. Interesting findings include the fact that for all departments (e.g. research, sales, HR and products), 10% to 20% of all slides contain reused content.

[1] http://www.marketwired.com/press-release/presentations-are-critical-success-according-nearly-1-2-employed-americans-1949705.htm

Reusing complete text fragments was more common than reusing partial fragments. However, in many cases content is reused by the original author and from the 14.5% of all slides with reused content, only 20% to 35% of them contain content that is reused from another author. The product department is an exception where almost 60% of content is borrowed. Other than text, the most frequently reused components are graphics, screenshots and diagrams; content that is often time-consuming to reproduce. The study also followed up their content analysis with a survey. From the results we see that many employees start composing a new presentation by copying and pasting content from another presentation, or simply by taking a previous presentation as a starting point. Most respondents use their own material "often" and "very often" (79%) and 83% at least sometimes use materials created by their colleagues. The reason most frequently cited for reuse is "content is time consuming to re-create", followed by "to maintain consistency with content previously presented" and "content is difficult to recreate". Finally, the study also conducted some interviews to further determine the needs of the knowledge workers and to discover any shortcomings in their current tools. Some interesting conclusions are that presentations presentations are sometimes composed of slides from different sources (e.g. marketing and sales), further complicating the web of reused content. Important is also that 46% of participants stated that access to "strangers" and their materials is crucial. Currently, content is discovered by using mail, instant messaging, links to internal files or via project wikis. Most of the material reused comes from people's direct colleagues, and participants expressed a clear need for better tool support to find content outside of these circles. Another major issue stated by 67% of participants is the importance of being able to track the history and origin of content. However, this is not facilitated in current tools so authors often feel unsure about using material if for instance the origin or use rights are unclear. The study concludes by providing some explicit guidelines to improve the related tools. This includes the ability to track the history and origin of material, better ways of finding content made by others, better multimedia support and in general, tools should better support the workflow of presentation authoring and reuse. We limited ourselves to the most relevant findings and we refer to the original paper for many more interesting statements [188].

A 2012 study by Thielsch et al. [270] with 1014 participants mostly presents similar findings. For instance, also they found that presenters often take an existing presentation as a starting point when creating a new presentation. Other overlapping findings include the reuse of content and the import of content from external applications. The majority (92%) of participants imported data from other sources or applications, with images being the most frequent type of content. Approximately one-third of the participants frequently imported text or tables from other applications. The study also confirms that presentations are very often shared via email.

3.1.3 Need for Improved Content Support

The need for better content support can be interpreted from multiple angles. First of all, some content types are simply difficult to incorporate in a presentation. For instance, when source code is copied to a presentation all formatting and colouring is lost, requiring significant effort to make it presentable again. Similarly, teachers of mathematics courses sometimes state that preparing slides with mathematical expressions is too time consuming [104, 245]. As such, one way to interpret these statements is to say that there should be support for a wider range of content types. Additionally, tools should also reduce the effort required to properly present these content types. Furthermore, we see many interesting findings in the domain of information visualisation but existing tools offer little to no support to actually apply them. For instance, using storytelling [157, 244] or using interactive and dynamic visualisations [293, 70, 252, 302] have been shown to bring many benefits but presentation tools make it a tedious and time-consuming process to apply them. For this reason many presenters opt for easier, but less effective options. Farkas [86] also argues that employers might simply not want employees to spend all their time on optimising a presentation. Finally, the uni-directional flow of information encouraged by tools such as PowerPoint is further criticised, and as will be shown later, involving the audience and providing interactive content that involves the audience can further improve knowledge transfer.

3.1.4 Need for Improved Slide Design Support

Finally, we mention again that tools such as PowerPoint offer many options related to layout and style, but the user is responsible for properly applying them. This puts the user at risk of also applying them in negative ways [278]. So as another aspect of improved content support we point out that presentation tools can be improved if they also help users to use layouts and styles effectively. Better support would help reduce the time and effort needed to create a presentation, as a lot of the time is spent of design and animation [270]. Microsoft Research presented a tool for synchronising the alignment of multiple content element across different slides [79], but this is arguably a workaround for a deeper problem. There are many studies and guidelines on the design and layout of presentations. For instance, studies [267, 35] find that users prefer light backgrounds and the use of contrasting colours in content. Mackiewicz investigated which fonts are preferred [181]. Alley et al. argue that slide titles should always be full sentences [8]. Some sources will recommend reducing the amount of content [278, 103, 280, 34] on slides. Roth [234] provides guidelines for showing the progress of the presentation on every slide. Other authorities on academic communication such as Jean-luc Doumont [72] will also offer various guidelines. However, addressing this problem is not as simple as automatically applying existing guidelines. One may

have for instance noticed that guidelines contradict each other. The reason behind this, is that there is no optimal slide design and different use cases require different approaches. For instance, some use a presentation style with only images popularised by for instance Steve Jobs or TED Talks[1]. Similarly, Godin [103] states that a slide should not contain more than six words. However, this does not work for all scenarios and particularly not for education [86]. Farkas further argues that a lot of strongly voiced opinions against a particular presentation style originate from people who only have experience with a particular use case that has different requirements [86]. The differences in design requirements are further illustrated in a study by Gomez [106] where presentation content and lectures from different fields are analysed.

3.2 Observations

Later in this chapter we present a survey as well as a programmatic analysis of PowerPoint document that helped us to identify shortcomings and unmet user needs. However, we also felt that it is was necessary to perform observations and to attend some presentations with the goal of identifying any recurring patterns. There were two main motivations for performing these observations. First of all, the observations act as a pilot study that helped us to direct our literature study, survey and programmatic analysis. Second, we foresaw that some findings would not present themselves in the survey or programmatic analysis, as these forms of studies do not take context or situational information into account. To help us get an initial view on the use and potential shortcomings of presentation software we attended eight presentations at a university (Vrije Universiteit Brussel). We observed five lectures by professors, one research presentation by a PhD candidate and two seminar presentations by Master students. We intentionally varied the topics of the presentations and we attended presentations on computer science, mechanical engineering, biology, theatre, sociology and literature. All presentations were given by different presenters and lasted between one and two hours, with an average duration of 95 minutes. We obtained permission from the presenters to attend their presentations, to record audio and to make details notes of the events that occurred. After eight observation sessions the lack of new observations indicated that additional observations sessions would not contribute much. We analysed our observations by tagging and categorising the observed events and facts in MAXQDA[2], resulting in a code tree.

We summarise the most important findings. All but one presentation was delivered by projection and all presentations were driven by a laptop computer (five laptops running macOS, three laptops running Windows). Two presentations were created

[1] `https://www.ted.com/talks`
[2] `https://www.maxqda.com`

in PowerPoint, one in Keynote, two in LaTeX (with the Beamer package), and one presentation was essentially a website with relevant content created by the presenter. In terms of peripheral hardware, two presenters used a laser pointer, two presenters used a presentation remote, one presenter used a stick for pointing and four presenters used a blackboard or whiteboard during the presentation. As only two presenters used a presentation remote, most presenters used the keyboard of their laptop to navigate slides. One interesting observation is that these six presenters would be forced to remain within a few meters of their laptop and they remained in close proximity of their laptops for the whole presentation. Another interesting observation is that presenters try to keep track of time and frequently interrupt their presentation to look at a clock. Three presenters checked their wristwatch frequently and one presenter pulled a smartphone out of their pocket multiple times. We noticed major differences in the flow of the presentations and interactions with the audience. Three presenters encouraged the audience to interrupt with any questions or remarks. In these cases the audience members made use of this offer and would interrupt the presenter multiple times to ask for clarification or to share an opinion. On the other hand, during the other five presentations, audience members would listen or take notes but they did not interact with the presenter.

In terms of the actual presentation content, we saw that all but one presentation used images. Three presentations included charts and we observed box plots, graphs, bar charts, line charts, venn diagrams and bubble diagrams. Other observed content types include text and bullet points (all presentations), diagrams (two presentations), mathematical formulas (two presentations), interactive simulations (one presentation), tables (three presentations), source code (two presentations) and video (two presentations). One observation that we find interesting is that the use of content and slide design depended heavily on the use case. For instance, the lecture related to theatre was mainly driven by images and contained very little text. On the other hand, the lecture in computer science used almost no images and relied on slides filled with long bullet lists and mathematical formulas. Three presentations highlighted certain parts of their content with colour or by making text bold. Highlights were applied to text, diagrams and tables, and in all cases these highlights were animated to appear over time, drawing the attention to specific parts of the content. Five presentations contained elements that provide structure or overview in the presentation, such as labelled sections spanning multiple slides (e.g. “introduction” or “related work”), tables of content or progress bars.

We observed two major strategies for revisiting content from earlier slides. Three presenters navigated backwards in the slide sequence, one slide at a time, at least twice. Two other presenters placed identical content on multiple slides, foreseeing that they would need this content multiple times and avoided having to navigate

backwards. For the presenters that did navigate backwards and forwards by more than one slide, we observed situations where it went wrong for three presenters. Three presenters had multiple cases where they had trouble finding the relevant content when going back in the slide sequence. One presenter had issues to get back to the initial position after going back in the slide sequence. We also observed one scenario where the presenter moved to the next slide and then realised they had already discussed all the content on that slide by accident. As some final causes of issues, we observed two cases where the lighting in the room made content on the blackboard illegible, and in two cases the presentations contained large tables or large bodies of text that were too small to read.

Although these findings do not prove anything on their own, they did provide some points that were investigated in more detail as part of the literature study, survey and programmatic analysis. For instance, these findings made us suspect that the use of presentation software is different depending on the use case (e.g. topic of the presentation). We hypothesised that context affects the frequency of specific content types as well as the slide design, and that some situations require more than just images. We later confirmed this theory based on existing literature and we have discussed these findings earlier in Section 3.1. Other observation-based studies describe similar differences in slide content for different academic subjects [133]. We also observed different degrees of interaction between the presenter and the audience, but a passive audience was observed more often. The fact that we observed multiple cases where presenters would go back to previous slides also indicated some limitations of using a linear sequence of slides. This suspicion was further strengthened by the fact that backwards navigation caused issues for all of the presenters that attempted to return to earlier slides. We further saw indications that presenters use a wide variety of content types, including various charts, diagrams, video, tables and even interactive and dynamic content. As a final point of interest we point out that we observed four situations where the presenter interrupted the presentation to check the time. We believe this might be cause by a lack of support for time management in presentation software. Furthermore, these interruptions might negatively affect the flow of presentations and might even affect how the audience perceives the presenter.

3.3 Programmatic Analysis of PowerPoint Documents

In order to gain further insight into user preferences and how presentations are currently used, we examined a large set of PowerPoint files. Given the many possible presentation styles and purposes, a large sample size is required and it would be infeasible to manually analyse these presentations. We therefore opted for a mainly programmatic approach and developed software for extracting and calculating values

for a variety of metrics from existing PowerPoint documents. This has resulted in a database with metadata for each individual presentation, down to the level of individual slides, text fragments and images. A complete description of the extracted information is provided in Appendix B. By not aggregating the results, the database can later be reused and queried by us, as well as by other researchers to test hypotheses and to provide answers to questions that are not discussed in this dissertation. In this section we discuss the methodology, our software framework for analysing presentations as well as some findings that helped us to define the requirements for the solutions presented in later sections.

3.3.1 A Representative Sample of Presentations

To obtain a sufficiently large set of presentation files that cover various use cases and scenarios, we decided that the best option would be to obtain them from a slide sharing service such as SlideShare[1]. However, SlideShare was only able to provide presentations as images or as PDF documents, but not the original PowerPoint files. This would make the processing much more difficult and some information can only be gathered from the original PowerPoint documents. As an alternative, we retrieved our dataset from SlideServe[2], a similar service that hosts millions of presentations, all of them available for download as the original PowerPoint files. However, when visiting the website it is clear that SlideServe ranks uploaded presentations based on factors such as the amount of views and downloads. On the front page, and also when browsing presentations by category, only the very best presentations are offered, making it difficult to collect a truly representative sample. In order to avoid a biased sample, we created a basic web crawler that visited all pages of the SlideServe website and gathered a list of all the available presentations. Special care was taken to not disrupt the SlideShare service. We limited the request rate and built the index over multiple days. This resulted in a database containing 6 777 508 links to individual PowerPoint presentations. Over the course of multiple weeks we then started sampling the set of available presentations and programmatically downloaded randomly selected PowerPoint files from SlideServe. Over 13 000 presentations were retrieved resulting in 28GB of files in the ppt, pptx, pps, ppsx and ppsm PowerPoint document formats. The sample size was chosen while keeping in mind that in the future we, or other researchers, might have very specific hypotheses that need to be tested. The chosen sample size allows one, for instance, to ignore presentations that are not in English and that are not university lectures, and still have a sufficiently large sample to work with. Although we did not allow the same presentation to be downloaded more than once, we noticed some duplicates in our sample. This is due to the fact that the same presentation can be uploaded to SlideServe multiple times (e.g. by

[1] `https://www.slideshare.net`
[2] `https://www.slideserve.com`

different users). To address this we used the FDUPES[1] tool to remove duplicate files. This tool takes an MD5 hash of each file's content, allowing it to reliably find duplicates even if file metadata such as the filename is different. Furthermore, from the remaining presentations we removed another 52 presentations that had corrupt document formats which could not be processed by PowerPoint or by our own analysis software.

After the removal of duplicates and corrupt files, we were left with 12 558 unique PowerPoint files. Surprisingly, even though anyone can download these files from SlideServe the PowerPoint documents were not anonymised and the original metadata was intact. In other words, each presentation contains the original content (including audio but not video fragments), animations and effects, as well as presenter notes. However, they also contain metadata stored by PowerPoint including for instance the author's name, creation date, last modified date, total time spent on creating the presentation, or details about the operating system and PowerPoint version that was used. A basic manual inspection of the sample confirmed that the sample contains many of the expected academic lectures and business presentations, but we also encountered many presentations for less professional use cases. This includes for instance student presentations, religion themed presentations, slideshows with holiday pictures, illustrated stories for children or presentations with song lyrics used for karaoke. Given our sampling method and the observations made from manual inspection, we believe that the sample is sufficiently random and covers all the common use cases of presentations. However, there is one source of bias that should be considered. One can argue that some user types are more likely to upload their presentation to services such as SlideServe than others. For instance, it might be possible that lecturers in Computer Science are more likely to upload their content to SlideServe than for instance a student that creates presentations for school. We acknowledge this as a potential issue and although a manual inspection confirms a better-than-expected variety of presentations, some presentation types might be represented more than others due to the source of our data. Even so, we do believe our approach is more random and less biased than alternative approaches such as collecting and combining presentations from individual schools or businesses.

3.3.2 A Software Framework for Analysing the Sample

In order to analyse the 12 558 PowerPoint presentations covered by our the sample, we created a small modular software framework. The main goal of the framework is to process presentation files and extract metadata and statistics, and store them in a database. We anticipated that we would have to run different experiments on the data so a first feature of the framework is that it allows users to define small modular

[1] `https://github.com/adrianlopezroche/fdupes`

tasks that gather specific data from a presentation and write the result to a shared database. As an example, one task might determine the language of the presentation, another task counts the amount of slides, another counts how much text is used per slide, and so on. This way, one can choose which tasks to execute during a particular run, and later runs with different tasks simply add to the already existing database of extracted information. This feature paid off when the results of initial runs triggered ideas for new experiments and tasks that we were able to execute without having to execute the old tasks. Each task specifies the type and structure of the information that it extracts, and the framework automatically evolves the database schema and ensures that no existing data from other tasks is overwritten or deleted.

A second feature of the framework relates to performance and scalability. Given the size of the sample and the variety of experiments that we had in mind, a single run over all presentations can take multiple days. In order to make use of more powerful computer hardware, the framework uses a configurable threadpool to optimally use the available CPU cores and to execute tasks in parallel. Tasks can also exchange data and depend on other tasks. For instance one task might extract all text from a slide and write it to a thread-safe in-memory "registry" so that other tasks such as language detection or character counting can perform their duty without having to extract the text from the document again. Similarly, a presentation document is only parsed once and the document model is kept in memory for other tasks, until all tasks have been performed on that presentation. Note that some plug-ins might write results to the filesystem instead of to the database. For instance, one of our tasks creates and stores a thumbnail of every slide and some of our other tasks analyse these files for other metrics related to the use of colours or signal-to-noise ratio.

The framework was developed in Java 8 and uses SQLite[1] for persisting results to a portable file-based database. To parse the PowerPoint document and to gain access to the smaller components within, we make use of the Aspose.Slides[2] software library. Aspose products are used by many large applications and services that require document processing. It was selected over the free Apache POI[3] project due to its ability to handle earlier versions of the PowerPoint document format that are not based on OOXML. Aspose.Slides provides us with a consistent document model and API, regardless of the version of the underlying document format. Most tasks in our framework make use of this document model and the provided utility functions to extract or derive information. Earlier we mentioned we encountered some corrupt PowerPoint documents in our dataset. A very small amount of them would cause unexpected errors and even crashes in the Aspose library. Given the widespread use of the library in industry, we believe these issues would have been solved if they were

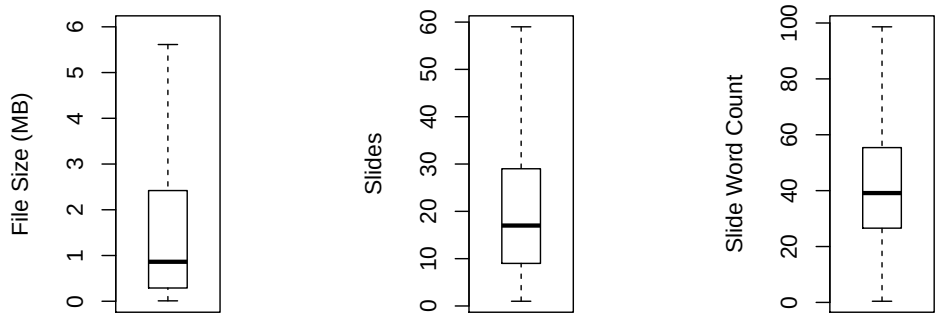
[1] `https://www.sqlite.org`
[2] `https://products.aspose.com/slides`
[3] `https://poi.apache.org`

known. For this reason we documented these rare cases and sent Aspose a detailed bug report with the presentation documents so that the issues can be reproduced and addressed.

3.3.3 Analysis Findings

After running various experiments, we had processed 12 558 PowerPoint documents containing 282 392 slides and 3 887 267 fragments of text. We extracted metadata for every presentation, slide, image and text fragment. A complete description of the extracted information is provided in Appendix B. We now present the most important findings. The documents had an average file size of 1.99MB and contained an average of 23 slides (without counting hidden slides). The distribution of this data is shown in Figure 3.1. However, for the plot of the file sizes we had to disable the visualisation of the outliers which are as high as 120MB. To determine the language of every presentation, we used the Optimaize language detection library[1] which is based on N-grams derived from Wikipedia articles. We detected 52 different languages, with the most common languages being English (66.98%), Spanish (5.39%) and French (2.79%). In 4.37% of the cases the language could not be determined, potentially due to a lack of textual content. To find out how much text is on the slides, we focussed on English presentations and we found that these presentations have an average of 50.40 words per slide. We did not simply average the word count of all slides as presentations with many slides carry more weight if the corresponding authors use very low or very large quantities of words. For this reason we calculated the mean word count by first calculating the average for every presentation and then taking the average of averages. The distribution of the word count is further illustrated in Figure 3.1c but we disabled the visualisation of outliers that go as high as 1800 words. PowerPoint keeps track of the time users spend working on a presentation. However, PowerPoint simply keeps track of how long the document is open in the editor and even if the editor is not being actively used, the time is still counted. For this reason the average editing time of 94 hours has little meaning. We investigated the use of presenter notes, and after filtering out hidden slides and notes with only whitespace characters, we found that 16.81% of all the presentations contained at least one presenter note. When investigating the use of hyperlinks we found that 42.11% of the presentations contained at least one hyperlink (with an internal or external target). However, note that some PowerPoint templates automatically embed some clickable elements on slides. When we analysed the use of multimedia we saw that 4.73% of presentations contained audio fragments. Sadly, the downloaded documents had their video content removed so we could not make any observations on the use of video.

[1] `https://github.com/optimaize/language-detector`

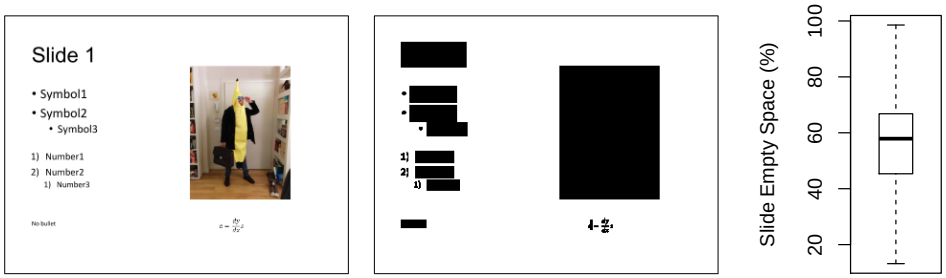


(a) File Size (b) Slide Count (c) Slide Word Count

Figure 3.1: The distributions of file size, slide count and words per slide

In order to investigate how presenters make use of the available space on slides, we performed an experiment where we calculated how much space is left empty on slides. To achieve this, we blacked out all the content (e.g. text, images, bullets, formulas and diagrams) as shown in Figure 3.2a and Figure 3.2b. We did not count slide backgrounds as content and we simply set the background to plain white for every slide. We then calculated the ratio between the amount of white pixels and the total amount of pixels to find out how much surface is covered by content. We calculated that on average, slides are 53.11% empty. Similar to the word count for slides, we first calculated the average for every presentation, and then took the average of averages. The distribution of the whitespace percentages is shown in Figure 3.2c.

We also investigated the use of images and found that on average, there are 0.82 images per slide. We again calculated this by averaging the average of every presentation. The distribution of the image frequencies is shown in Figure 3.3. Note that we only counted real images such as embedded JPG, BMP or PNG files. Graphical content such as diagrams created natively in PowerPoint were not counted as images.



(a) Original slide (b) Blacking out all content (c) Empty Space

Figure 3.2: Calculating the spatial coverage of content in slides

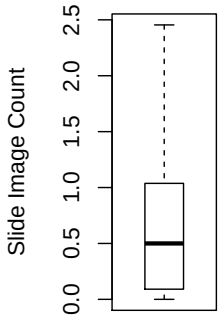


Figure 3.3: The distribution of images per slide

One major issue was that we could easily count the amount of images with our framework for content analysis, but we could not determine what these images represent. An image can for example be a chart, a diagram or a screenshot of source code. Because a chart can be created natively in PowerPoint but it can also be imported as an image, it would be wrong to calculate the frequency of charts based on the native charts our framework encountered. For a correct calculation of the frequency of charts one would have to count native charts as well as charts that were included as images. The same is true for some other content types such as diagrams or tables. In a first attempt to detect charts in images we used Keras[1] to train a neural network for image classification. We assumed that the distinct features of charts such as white grounds and bright colours would allow us to automatically detect charts in images with a reasonable level of accuracy. However, we were not able to obtain a high enough success rate with this approach. We believe that this is due to the fact that that clipart, charts, diagrams and other forms of images share similar features, from the perspective of a neural network. We believe that it was important to classify the images one way or another, so we decided to manually process a subset of the data. We manually processed 254 presentation documents and we classified all images found within 6 304 slides. These 254 presentations were randomly selected from the total set of 12 558 PowerPoint documents. The benefit of manually processing some documents is that we were able to classify images into many different categories and because of our effort we were able to calculate correct estimates for the frequency of content such as charts and diagrams. We present the frequencies of the manually observed content types in Table 3.1. The "None" column indicates the percentage of presentations that did not contain that specific content type at all, and the "Any" column indicates the percentage of presentations that contained at least one instance of the specified content type. For the content types that can be created either natively or by embedding an image we grouped the presentations with

[1] `https://keras.io`

only native instances of the content type ("Native Only"), the presentations with only images for this content type ("Image Only") and presentations that contained a mix of native and image instances of the content type ("Mixed"). To clarify some of the content types, a photo is a photograph taken with a camera, or in other words it is a snapshot of the physical world. An annotated figure is a photo or abstract drawing that has been annotated with text, usually for learning purposes. Examples that we encountered are anatomical cross-sections, cross-sections of complex machinery, architectural designs and circuit designs. Simulations are images that visualise a simulated environment, for instance as a 3D computer-rendered model or as an snapshot of a simulated experiment.

Content Type	Native Only	Image Only	Mixed	None	Any
Charts	7%	18%	3%	72%	28%
Tables	16%	7%	3%	74%	26%
Diagrams	15%	14%	8%	63%	37%
Source Code	2%	2%	0%	96%	4%
Math. Formulas	3%	4%	1%	93%	7%
Photos	-	51%	-	49%	51%
Clipart	-	15%	-	85%	15%
Annotated Figures	-	11%	-	89%	11%
Screenshots	-	11%	-	89%	11%
Maps	-	11%	-	89%	11%
Simulations	-	3%	-	97%	3%

Table 3.1: The percentages of presentations that contain the specified content types

When interpreting these results there are some interesting observations to be made. First of all, in our manually processed sample most presentations exclusively embedded charts as images that were created externally. Charts that were created in Excel were counted as native charts and did not even count as external content as it is the default solution offered by PowerPoint. This might indicate that the built in support for charts is not sufficient. A second related observation is that there were also presentations that used both native as well as image charts. Although we have no concrete numbers to support this, there are indications that these presenters use the native chart tools for simple charts but were forced to use external tools for more complex plots. Another explanation for this behaviour is that some presenters reuse charts from for instance academic papers and then supplement them with native charts. We also noticed that a lot of the diagrams were animated to make parts appear one by one or to highlight certain areas. Some presenters even used an image-based diagram as background and would use native diagram components as an overlay. To gain better insight into the distribution of native content and content important as images we created the scatterplots provided in Figure 3.4. Every dot

represents a presentation document, and the axes represent the amount of native and image content. The diagonal represents the situations where both approaches are used equally. The plots make it clear that some presenters use only native content or only imported images, but others create mixes with varying ratios. As a final finding of the manual analysis we observed that 18% of presentations end with a contentless slide such as “Questions?”, “The End” or “Thank You”.

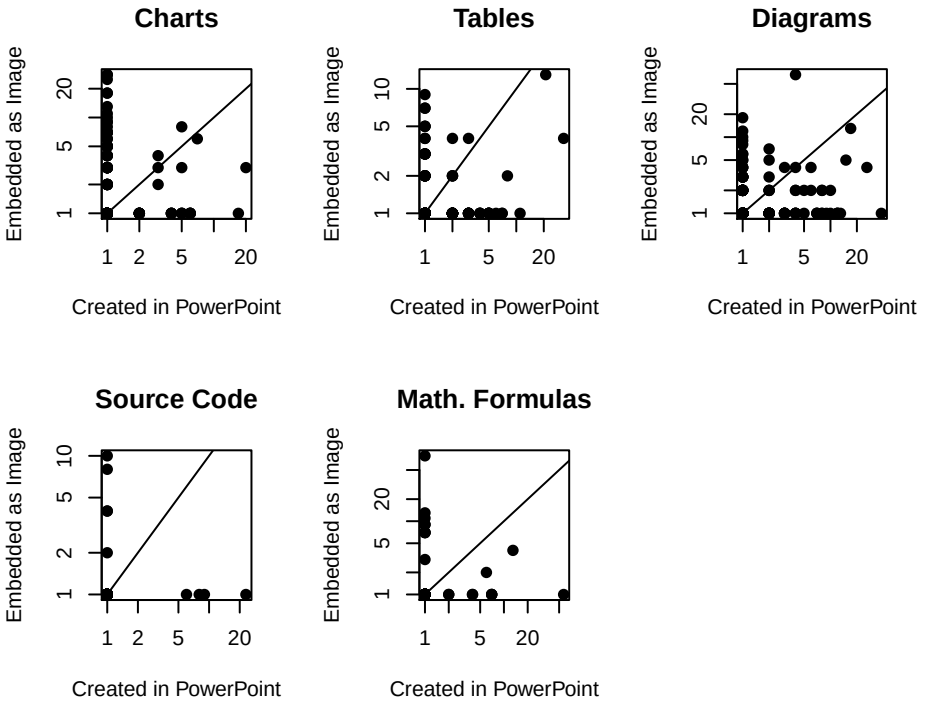


Figure 3.4: Plotting the presentations (black dots) to see the distribution of content created natively and content imported as an image

We further programmatically investigated the use of colours in slide backgrounds in the original sample of 12 558 documents. For every presentation we iterated over all slide backgrounds and composed a colour histogram. To facilitate the processing of this information and to be able to store these histograms in our database we rounded the colours to the palette of 216 web-safe colours. We chose this palette because the 216 web-safe colours are evenly spread in the RGB colour space and the palette is composed by making all possible combinations of 6 evenly spread red values, 6 evenly spread green values and 6 evenly spread blue values. For every presentation, we store a histogram as an array of 216 frequencies. The entire array is then converted to a

compact byte array and encoded using base64 so that it can be stored as a single value for every presentation. Note that the histograms in the database contain the non-normalised colour frequencies for all slides in a presentation. For this reason presentations with more slides have higher frequency values in the histogram. We deliberately made this choice so that the users of the data can decide for themselves if they want to perform a weighted calculation where presentations with many slides count more, or if they want to normalise the histogram and use the same weight for every presentation.

For our analysis of the colour histograms, we combined the normalised histograms of all presentations and we show the frequency of all the colours in Figure 3.5. It is evident that white is by far the most used slide background colour, with black and some shades of blue as other popular colours. However, the visualisation is difficult to interpret, so we provide an overview of the 25 most frequent colours (in the palette of 216 web-safe colours) in Figure 3.6a. It seems that black is indeed the second most popular background colour, but this result is deceptive because there are many different shades of blue that are each counted as a different colour. We further investigated this by mapping the histograms down to a 4-bit palette with 16 colours. It then becomes clear that light blue and dark blue are used more often than black in slide backgrounds. While experimenting with different sorting algorithms for a creating a nice colour spectrum visualisation we made another interesting observation. When applying a stepsort in the HSV colour space[1] we noticed that the colours that occurred somewhat often formed three distinct clusters of seemingly unrelated colours in the histogram. This is shown in Figure 3.6c. We have no concrete explanation for this phenomenon. One possible explanation is that the stepsort algorithm unintentionally groups some complementary colours together and that these colours are used in some of the default slide templates. On the other hand, it might also be a coincidence and this was not investigated any further.

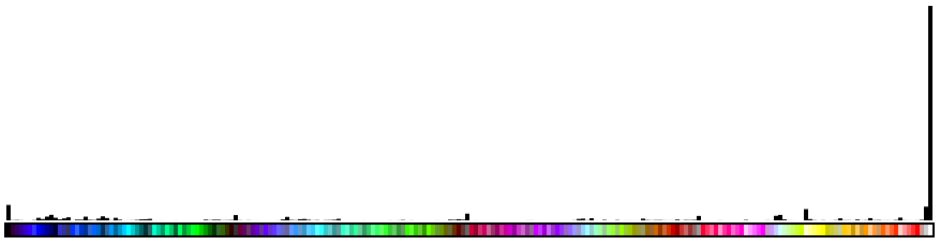
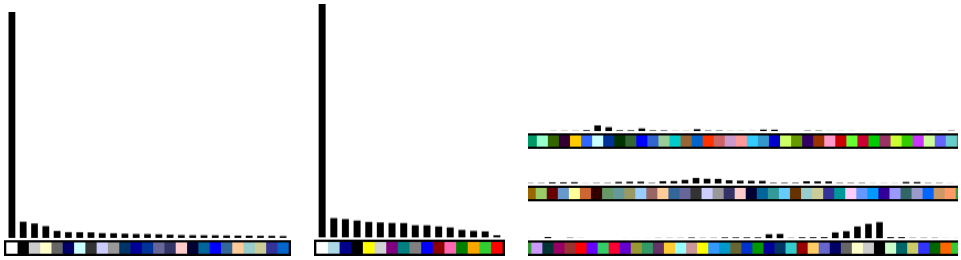


Figure 3.5: Frequency of individual colour occurrence in slide backgrounds

[1] `https://www.alanzucconi.com/2015/09/30/colour-sorting/`

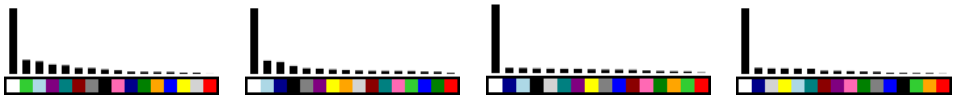


(a) Top 25 colours (b) 4-Bit palette (c) Clusters in the stepsorted spectrum

Figure 3.6: The most common colours in presentation backgrounds when (a) mapped to the 216 web-safe colours, (b) mapped to a palette of 16 colours and (c) on a stepsorted spectrum of web-safe colours

After these more general findings on the use of colour in slide backgrounds, we performed another experiment with the goal of determining if colours are used differently in different countries. To test this hypothesis we grouped the presentations based on language and we focussed on four languages: Arabic, Chinese, English and Russian. We then performed the same analysis of the histograms for every language group. The results are shown in Figure 3.7 and they seem to indicate that there is indeed a difference in the use of colour between these language groups. As the colour histograms represent the distribution of different colours, we ensured that both histograms were normalised and we used chi-squared tests to compare the colour distribution of all presentations with the colour distributions of our individual language groups. The tests confirm that the colour distributions of the language groups are significantly different compared to the colour distribution of all presentations ($p < 0.05$). When looking at the exact colours for every language group these findings become even more interesting. After conversion to a 4-bit palette with 16 colours, the second most used background colour for Arabic presentations is light green, a colour that is associated with luck in Islamic cultures [7]. In the Chinese language group one can see that orange is ranked much higher than in other groups. It is important to note that we mapped the 216 web-safe palette to the visualised 4-bit colour pallet by measuring colour distance in the HSL colour space. As a result, many lighter shades of red are "rounded down" to the orange colour category in the 4-bit palette. Red is considered a lucky colour in Chinese culture [7], but as dark red would not work well as a slide background so we see a more frequent use of light red backgrounds in Chinese presentations. We did not find any specific colours that stand out in the English or Russian language group, but they are also significantly different from the colour distribution of the total set of presentations. Although these findings may seem obvious, to the best of our knowledge we are the first to demonstrate this with our large dataset. Furthermore, this seemingly trivial finding has important

implications. In Section 3.1.4 we have already established that different users have different requirements for presentation software, and slide design can be significantly different for different use cases. The cultural differences we identified in the use of colours is an indication that slide design is not only affected by the use case or by the type of content, but that it can also be influenced by cultural elements.



(a) Arabic (n=127) (b) Chinese (n=158) (c) English (n=8412) (d) Russian (n=185)

Figure 3.7: Analysing cultural differences in slide design

As a final experiment we revisited the slide counts of all presentations in the dataset. We thought it would be interesting to fit this data to a specific distribution. Such a distribution could then be used as a model and would allow us to quantify slide counts (e.g. is 40 slides a lot or not?). To our surprise the slide counts are not normally distributed and the empirical distribution is visualised in Figure 3.8 as the black curve. The shape resembles distributions such as the Log-normal, Gamma, Weibull or Inverse Gaussian distributions. We estimated the parameters for the best fit of these distributions, and we also show these in Figure 3.8. We also tested the distributions for statistical significance. As the Kolmogorov–Smirnov test should not be used with pre-estimated parameters we used dedicated goodness-of-fit tests for every distribution, provided by the goft R package[1]. We were not able to match any of the distributions with statistical significance. However, a visual inspection of the distributions using Q-Q plots indicates that the Log-normal distribution might be the closest match. In fact, the Q-Q plot for the Log-normal distribution (also shown in Figure 3.8) indicates that the mismatch likely occurs in the right tail of the distribution, caused by some outliers with absurdly high amounts of slides (149 presentations have between 100 and 474 slides). Given the wide range of possible values for the slide count ($0 - 474$) and our sample size, such outliers can have a big impact in some tests. We did not investigate this any further, but we see the Log-normal distribution as the best match for our sample.

[1] `https://www.rdocumentation.org/packages/goft/`

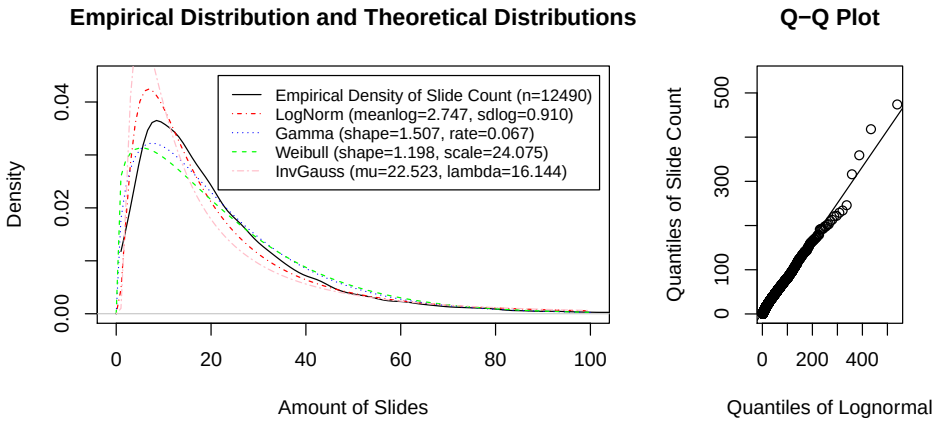


Figure 3.8: The potential distributions of slide counts

3.4 Survey

From the literature study, observations and programmatic analysis of PowerPoint documents we identified some potential shortcomings and unmet user needs. In order to further verify these findings we performed an additional survey. The goal of the survey was to obtain information about the general use of presentation software but also about some very concrete aspects that came up as potential shortcomings in literature, the observations or in the programmatic analysis of documents. For this reason the questionnaire was relatively long, with 47 questions, and took 15 to 20 minutes to complete. We hosted the questionnaire on a LimeSurvey[1] instance and we have included the questions in Appendix C.3. To motivate the participants we promised to give a €50 voucher for Amazon to three randomly selected participants. We took the appropriate measures to keep participants anonymous but if participants wanted to be included for the random draw they were asked to provide an email address so that we could contact them in case they won a prize. Participants were only allowed to submit once. To summarise the topics covered by the survey we asked how, and how often the participants create or deliver presentations, how much effort it takes them, which tools they use, which content types they work with and which issues they encounter. We also asked participants to rate the importance of slides compared to the other material that is available for their use case, such as textbooks. The participants where further asked about how they practice their presentations and how they share, collaborate or reuse content. We used a combination of quantitative and qualitative questions and LimeSurvey helped us to provide appropriate input methods

[1] https://www.limesurvey.org

for the participants (e.g. Likert scales, radio buttons, checkboxes, sliders or text boxes). Note that our questionnaire also contained some conditional logic and certain questions would only be shown if the participant's previous answers indicated that it made sense to ask further questions on this topic. We analysed the quantitative data with the R statistics package[1]. The qualitative data such as the replies to open questions were analysed using MAXQDA. Similar to the observations, we coded the responses for the open questions and we created a hierarchy of codes, grouping similar responses together and organising them into themes.

3.4.1 Survey Findings

In the following subsections we discuss our findings and present them as different groups of subjects.

Participant Information

We only considered complete submissions where the participant filled in all the questions. After cutting more than half of the entries and keeping only complete submissions we ended up with 43 valid entries. The group of 43 participants consisted of 29 males and 14 females. Most participants had an age between 19 and 39, with 26 participants aged between 19 and 29, and 14 participants aged between 30 and 39. Participants were further asked to rate their computer skill level on a Likert scale ranging from "Poor" to "Excellent". Most of the participants (81%) rated their computer skill level as "Very Good" or "Excellent". When asked for the purpose of their presentations 29 participants selected academic knowledge transfer, 18 participants selected corporate knowledge transfer, 4 participants selected sales and 5 participants selected entertainment. Note that participants were able to select more than one option.

Presentation Authoring and Delivery Frequency

We asked participants how often they created presentations and how often they delivered presentations. About half the participants performed these activities at least a few times a year, and 16% of the participants performed these activities at least once a week. The exact results are visualised in Figure 3.9.

Presentation Software and Hardware

When asked about the preferred presentation software, 58% of the participants mentioned PowerPoint as their tool of choice. All participants use some form of presen-

[1] `https://www.r-project.org`

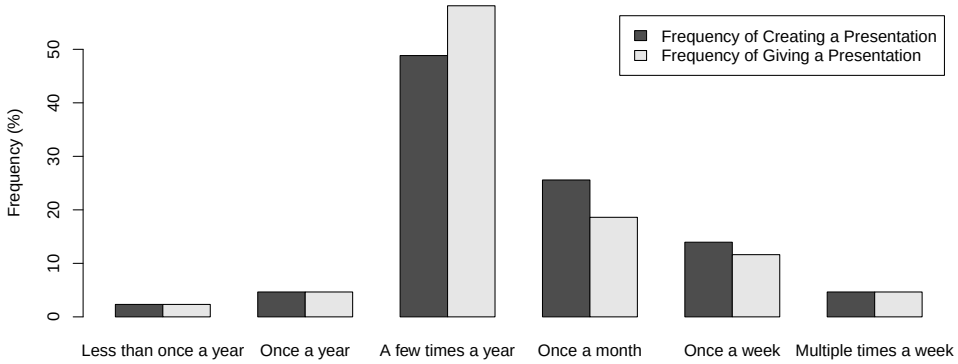


Figure 3.9: Frequency of creating and delivering presentations

tation software. Figure 3.10 shows the distribution of the different tools that were mentioned. The two participants that selected the "Other" option specified reveal.js[1] as their preferred tool.

In addition to dedicated presentation software, presenters also use other software to visualise certain information. We asked the participants to specify all the additional software they use during presentations. We found that the participants use web browsers (35%), PDF viewers (16%), programming IDEs (12%) and video playback software (9%) during presentations. There are also various scenarios where the presenter demonstrates a piece of software that is relevant to the topic of the presentation (4 mentions) or some operations in a terminal (1 mention).

We further asked presenters to specify the hardware that they use during their presentations. All participants mentioned the use of a laptop computer. The most common operating systems are Windows (25 mentions), macOS (9 mentions) and Linux (7 mentions). Note that participants were able to mention more that one operating system and some participants did not mention an operating system at all.

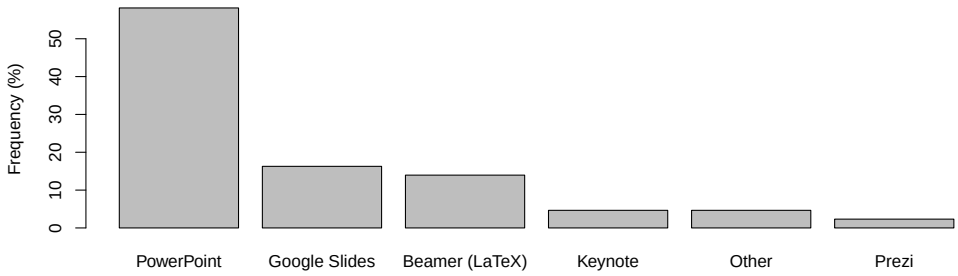


Figure 3.10: The presentation software used by the participants

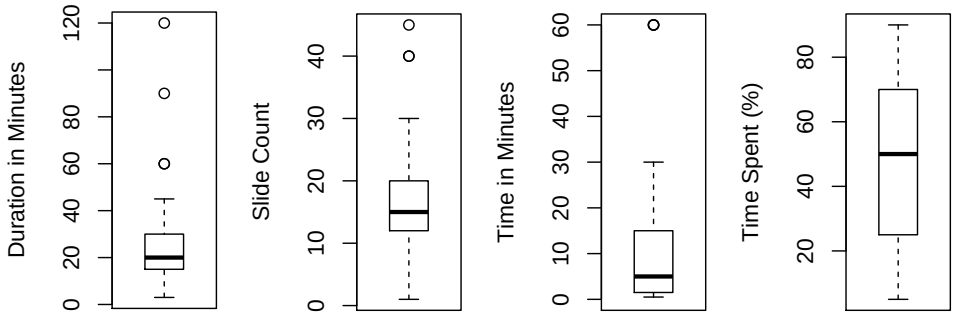
[1] `https://revealjs.com`

The use of a projector was mentioned by 56% of the participants and the use of a monitor or television was mentioned by 28% of the participants. The additional tools that are used by the participants are blackboards or whiteboards (26%), laser pointers (21%), handouts (12%), presentation remotes (7%), smartphones (7%), digital pens (5%), flipcharts (2%), tablets (2%), paper (2%) and a portable keyboard with a trackpad (2%).

We also asked specifically if any of the participants had used some tools for including the audience in presentations (e.g. clicker systems). Two participants mentioned the use of an online tool for real-time quizzes and polls via mobile devices. However, one of these two participants also mentioned that there were many issues, and another unrelated participant mentioned that they are not willing to use such systems because of the hassle.

Duration, Slide Count, Authoring Time per Slide and Time Spent on Visual Details

Participants were asked to estimate the average duration of the presentations they give. On average, participants presented for 25 minutes and 30 seconds. Keep in mind that we specifically asked for an average duration. Users such as academics often give a combination of short presentations (e.g. conference presentations) and very long presentations (e.g. hour-long lectures), and their average presentation duration is therefore lower than one might expect. A more detailed distribution of the presentation durations is shown in Figure 3.11a. The participants reported presenting an average of 18 slides as shown in Figure 3.11b. On average, our participants estimated that they needed 21 minutes to create one slide. However, as shown in Figure 3.11c the median is only 5 minutes and the mean is heavily skewed by outliers that go as high as 200 minutes. One possible explanation is that some participants might have misunderstood the question and that they provided the time needed to prepare an entire presentation. It might of course also be possible that some use cases actually require very high amounts of preparation time. In either cases the median of 5 minutes better reflects the time needed to prepare a slide. We further asked the participants to specify the percentage of time that goes towards visual details such as layout and aesthetics. As shown in Figure 3.11d, the results are more or less centered around the mean of 50% with a spread that almost covers the entire spectrum. In other words, the median presenter spends 50% of their time on visual details, but one quartile of presenters spend very little time on visual details and another quartile of presenters spends a large portion of their time on visual details.



(a) Duration (b) Slide Count (c) Authoring Time (d) Visuals

Figure 3.11: The average duration of presentations, the average slide count of presentations, the average time required to author one slide and the percentage of time presenters spend on visual details

Themes and Animation

When asking participants to specify the frequency at which they used animations in their presentations, 56% said they never or rarely use animations. If the participant did use animations we asked for the reason. Within the group of the 36 participants that did use animations, 42% mentioned the use of animation to create structure in the presentation or to synchronise the content with the oral narrative. The most common uses are making bullets appear one by one and for introducing some dynamic elements in diagrams or charts. Other reasons include capturing attention (28%), to make it easier for the audience to process information (21%), to highlight content (5%) and to fit more content on a slide (2%).

Related to visual details, another question investigates the use of themes and templates in presentation software. We asked which themes were used by the participants. The results include a school or company theme (26 votes), one of the themes offered by the presentation software (20 votes), a modified theme for personal use (11 votes) or plain white slides (9 votes).

The Importance of Presentations in Education

Next, we also wanted to find out how slides are perceived as learning material by the teachers themselves. We asked participants to provide a list of the learning material that is relevant for their courses (e.g. slides, textbooks, handouts or papers) and we asked them to sort the learning material by importance. Participants were only shown this question if they had selected "academic knowledge transfer" as one of their reasons for giving presentations. Participants were able to provide and sort up to 10 different learning materials. We show an overview of the top three choices in

Figure 3.12. An important finding is that the majority of the participants state that their slides are the most important learning material. Concretely, when considering only the first choice of each participant, 79% of the participants voted for slides as the most important learning material. The second most important learning materials (as perceived by the participants) include images, academic papers, textbooks, video material and practical exercises. Few participants provided more than three learning materials and therefore Figure 3.12 reflects the most important findings.

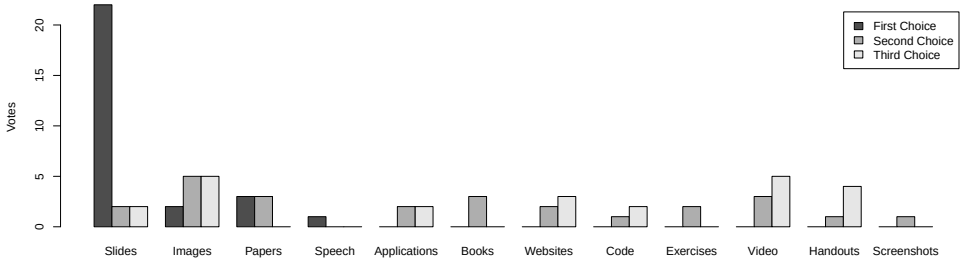


Figure 3.12: The perceived importance of different media as learning material

Presentation Content

We also wanted to gain some insight into the type of content the participants work with. The participants were asked to select all the content types that were relevant for their use cases and it was also possible to specify types that were not in the list. The results are shown in Figure 3.13. We present the results as the percentages of participants that use the specified content types. The following content types are used by most participants: images (100%), text (98%), charts (79%), tables (70%) and diagrams (68%). Less common are content types such as video (49%), hyperlinks (46%), source code (44%) and mathematical formulas (35%). Audio (9%) and geographical maps (2%) are the least used content types. As part of the "other" category participants mentioned Gantt charts, interactive code environments and interactive data visualisations such as animated simulations.

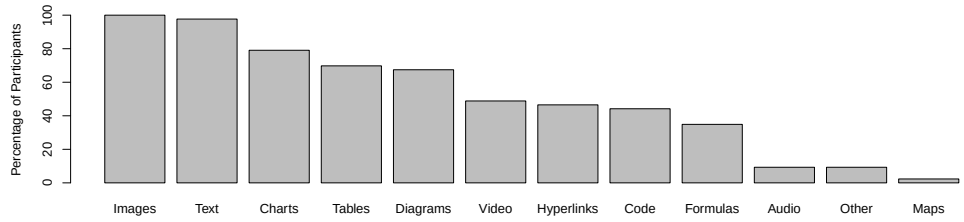


Figure 3.13: The percentages of participants that use the specified content types

In order to see if presenters pre-process some content with external applications we asked the participants to specify any additional tools they used while preparing presentation content. The results indicate that a lot of content is prepared outside of the presentation software even if the presentation software offers some support for these content types. The most common tools that the participants use during the authoring of presentations are chart and diagram tools (37%, this category does not include Excel), image editors (35%), Excel (21%), a programming IDE to take screenshots of code (12%), statistics packages (7%) and a dedicated syntax highlighter tool for source code (2%). The use of LaTeX or similar tools to create screenshots of mathematical formulas is only mentioned once as part of this question, but from the results of other questions that will be discussed later, it becomes clear that this practice is much more common.

Reuse of Own and External Content

We also asked participants to specify how often they reuse their own content and how often they reused external content. External content for instance includes images found on the Internet. The reuse of their own content is mostly within the frequency categories of "rarely" (28%), "sometimes" (28%) and "often" (28%). The reuse of external content seems to be more prevalent and is mainly categorised as "sometimes" (40%), "often" (21%) and "always" (19%). In other words, 60% of the participants reuse their own content at least sometimes, and 79% of participants reuse external content at least sometimes. These results are shown in Figure 3.14.

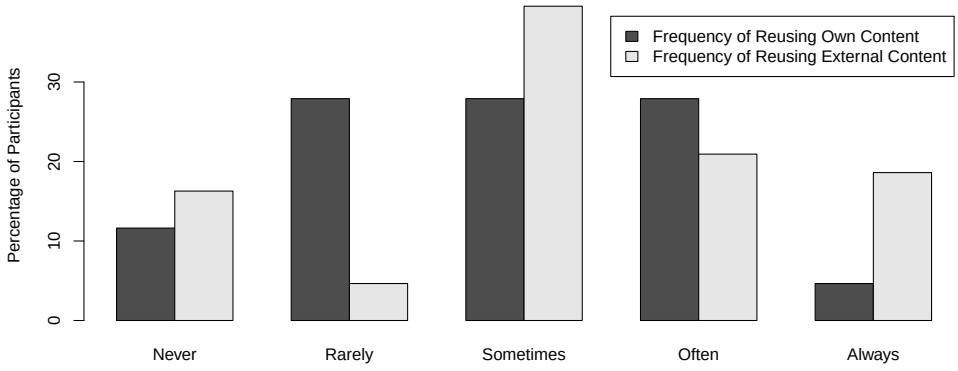


Figure 3.14: The frequency of reusing own and external content

We further investigated the topic of reuse by asking the participants to specify the sources for any external material that they reuse. Google Images is by far the most popular source of external content, used by 63% of the presenters. Other common sources include Wikipedia (23%), related websites found via Google (12%), websites

with stock photos and clip-art (9%), academic papers (7%) and slide hosting services such as SlideShare (5%).

Use of the Presenter View, Tracking Time and Practice Before Presentations

When we asked about the use of a presenter view such as the one provided by PowerPoint, 65% of the participants mentioned that they made use of such an overview during presentations. We then also asked the participants to specify why they thought such as overview was useful. Out of the 28 participants that made use of a presenter view, 79% mentioned they used a presenter view for the overview of the currently shown content, 54% mentioned using the overview for navigation and 61% mentioned time management as a benefit. From the group of participants that did not use presenter views, eight participants specific they have no need for a presenter view, one participant mentioned they moved too much to see the computer screen and one participant said such a view was too distracting. When we asked whether any features are missing, one participant specified the need for using different colours in presenter notes and one participant specified the need for a countdown timer instead of having the elapsed time shown on the screen.

We further investigated how presenters keep track of the time during a presentation. Forty percent of participants stated that they did not keep track of time at all. The remaining 26 participants were asked to specify all the ways they keep track of time. The resulting options provided by the participants are using the presenter view (15 votes), a wristwatch (11 votes), the computer's system clock (9 votes), a smartphone (9 votes), practising beforehand (2 votes), a wall clock (1 vote) and getting time queues from other people (1 vote). Note that the participants were allowed to vote for more than one option.

Some questions were related to how often and how the participants practised their presentations beforehand. The most common frequency categories were "rarely" (19%), "sometimes" (35%) and "always" (33%), as shown in Figure 3.15. When we asked the participants to list the methods they used to practice presentations, the common options were going over the content (27 votes), talking out loud (27 votes), presenting to friends or family (16 votes), filming themselves (3 votes) and practising in front of a mirror (1 vote). Participants were able to specify more than one method for practising presentations.

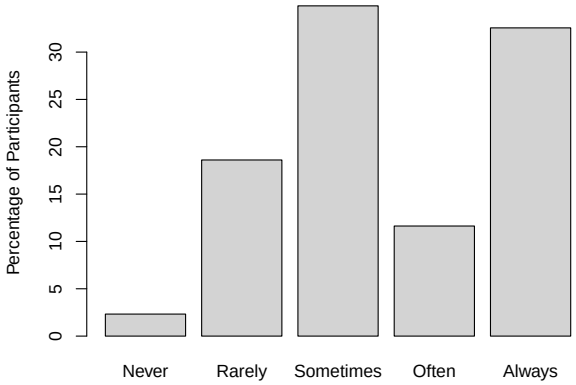


Figure 3.15: The frequency of practising a presentation beforehand

Slide Navigation

A next set of questions was related to how presenters navigate their presentation. When we asked how presenters switch between slides, the participants mentioned the keyboard (34 votes), mouse (13 votes), presenter remote (23 votes) and smartphone (1 vote) as input modalities. With the aim of investigating the frequency of non-linear navigation, we asked how often presenters went backwards of forwards by more than one slide during presentations. All responses fall within the frequency categories "never" (30%), "rarely" (47%) and "sometimes" (23%). To find out more about non-linear navigation we asked the participants if they sometimes showed the same content multiple times during the same presentation. We saw that 72% of the participants show the same content more than once in a presentation. There are however different strategies for revisiting content. The strategies described by the participants are: including the same content on multiple slides (15 votes), including the same content on multiple slides but with changes (25 votes) and navigating backwards or forwards to revisit previously presented content (13 votes). Participants were able to specify more than one strategy.

Collaboration

We also gathered some information about collaborative authoring. Figure 3.16 shows that most participants collaborate with others "rarely" (33%), "sometimes" (28%) and "often" (21%). 51% of participants collaborate more frequently than "rarely" ("sometimes", "often" or "always"). When we asked how the participants collaborate they mentioned the following strategies: creating content in isolations and merging everyone's content afterwards (15 votes), taking turns working on the PowerPoint document (10 votes), using collaborative software such as Google Slides (16 votes), synchronising a presentation via SharePoint, SVN or Git (11 votes) and physically

sitting together to create a presentation (2 votes). Participants were able to specify more than one strategy.

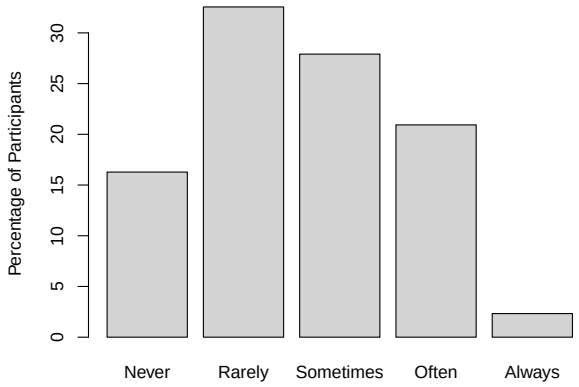


Figure 3.16: The frequency of collaborative authoring

Making Presentation Content Available Afterwards

One group of questions in the survey is related to how presenters make presentation content available after the presentation session. Figure 3.17 shows that most participants selected the frequency category "sometimes" (28%). The results further show that 65% of participants share their content afterwards at least sometimes. When we asked participants to specify how they made content available, the responses were email and instant messaging (22 votes), the school or company platform, or server (22 votes), a file sharing service such as Dropbox or OneDrive (18 votes), a personal website (2 votes) and GitHub (1 vote). Participants were able to specify more than one strategy.

The participants were also asked to rate the importance of their presentation content after the presentation session. About one third (30%) of the participants mentioned that their content serves some purpose after the presentation session. An open question asking about the specific purpose of their content after the presentation session revealed that content is used as study material (6 mentions), technical reports (5 mentions) and reference material (4 mentions).

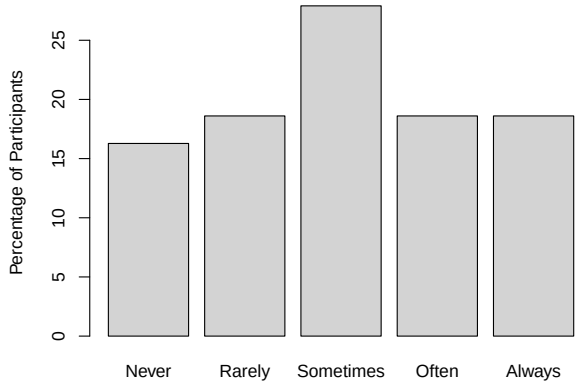


Figure 3.17: The frequency of making content available after the presentation session

If presenters shared their content afterwards, we asked them to rate the importance of their content as reference material for later use. As shown in Figure 3.18, 64% of the 36 participants that qualified for this question rated their content as "important" or higher. We also asked the participants to clarify their response in an open question. The most common reasons for participants to believe that their content is not very important is that there is often additional material with more complete information (9 mentions), the content is quickly outdated (4 mentions) and that the content will never be relevant to anyone else (1 mention). There are also reasons that explain why participants believe that their content is important, including that the presentation brings content from various sources together (4 mentions), that there is no additional material on the topic (2 mentions), for others to use as reference material (3 mentions) and for others to use their content (1 mention).

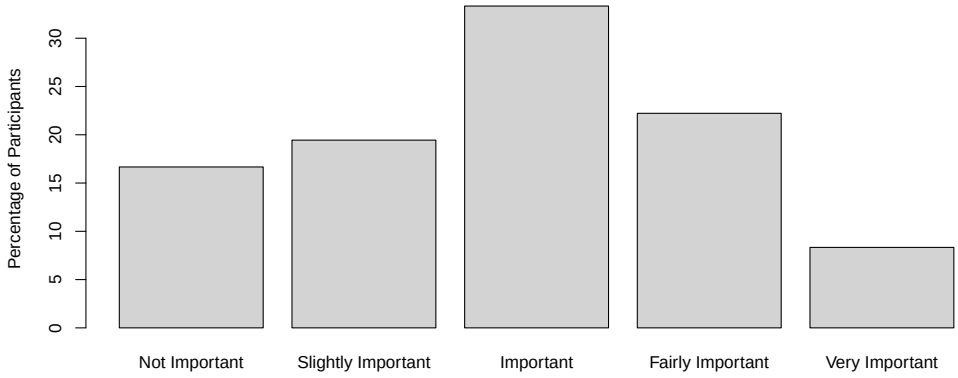


Figure 3.18: The importance of presentation content after the presentation session

Shortcomings and Unmet User Needs

From a final group of open questions related to the benefits and issues of presentation software, as well as from other open questions throughout the survey, we compiled a list of recurring themes. We start by discussing the positive aspects of presentations, as perceived by the participants. The ease of use is explicitly mentioned by 37% of the participants as part of an open question. The flexibility of themes is mentioned by 7% of the participants and the fact that PowerPoint presentations can be used on both Windows and macOS systems is mentioned by 7% of participants. The participants that use a code-based approach for creating presentations (e.g. via LaTeX or reveal.js) mention the ability to use code as a benefit (5%). The same participants also mention how the code-based approach allows them to use versioning systems such as Git (5%). Some users of Google Slides mention the features for collaboration as a benefit (5%). Some participants mention the ability to print to PDF (1 mention), the ability to print handouts (1 mention) and the ability to print an overview of the slides with the presenter notes (1 mention). Other noteworthy perceived benefits include the WYSIWYG editor (5%), support for resizing and aligning images (5%), the presenter view (5%) and the ability to work in a cloud-based application (2%).

Considering the goals of this dissertation, the perceived shortcomings are particularly interesting. The most frequent complaints are related to the styling and layout of content, mentioned by 28% of the participants. The most common reasons for these complaints are that it is difficult to fit complex content on a slide and the styling and layout of content is a tedious and time-consuming process. The second most frequent complaint, mentioned by 23% of the participants, is that the support for charts and diagrams is too basic and that it does not meet their needs. Next, also issues related to the playback of video material are reported by 19% of participants. Some of the frustrations are caused by unsupported video formats, technical issues with audio and internet connectivity, difficulties to display only the relevant parts of the video and the fact that embedded videos make your presentation document harder to share and distribute. Furthermore, 16% of participants explicitly mentioned difficulties associated with presenting source code. Some of the mentioned reasons are the lack of syntax highlighting and formatting, the limited space that is available on a slide and the lack of interactivity. Similar complaints are made on the topic of integrating mathematical formulas by 12% of our participants. The lack of an all-inclusive presentation solution is explicitly cited by 7% of participants and they mention that they need multiple applications to create the presentations and further also need multiple applications to deliver the presentation.

An interesting finding is that 9% of our participants mentioned the lack of features related to non-linear navigation as an issue. We have previously mentioned how some participants praised the fact that PowerPoint can be used on both Windows

and macOS. However, 7% of participants also had complaints on this topic and mentioned how content can look different of different platforms (2 mentions), and how there is no support for the real PowerPoint software suite in Linux (1 mention). A final category of complaints involves the reuse of content, brought up by 12% of participants. The participants mentioned that it is difficult to reuse some existing content in their presentation (3 mentions), that finding content for reuse is a major time-sink (1 mention) and one participant explicitly mentioned the lack of inclusion by reference ("transclusion") as a shortcoming.

There were also some complaints that were only mentioned once, but we quickly mention the ones that are relevant to other parts of this dissertation. One participant mentioned the need for an easier way to write on slides. Another participant mentioned that it is difficult to see the different states of an animated visualisation in PowerPoint. Last but not least, one participant mentioned the lack of interaction with the audience.

Time Spent on Visual Details for Different Presentation Software

One additional hypothesis we wanted to test is to see if there is a difference in time spent on visual details between PowerPoint and Beamer (LaTeX) users. LaTeX takes a declarative approach and tries to manage the layout and typesetting automatically where possible. To test this hypothesis we used the responses of the question where participants had to specify which percentage of their time went towards visual details. When we grouped these results we obtained a mean percentage of 53% for PowerPoint users (n=25) and 40% for LaTeX users (n=6). We attempted to test this promising finding for statistical significance. However, we were not able to demonstrate any significance. One of the issues is that it is unclear if we can assume a normal distribution for these percentages. We tried to verify normality with a Shapiro–Wilk test, but with a p-value of 0.02479 we had to reject the null hypothesis that the population of percentages is normally distributed. On the other hand, a visual inspection of the density plot and Q-Q plot indicated that one might be able to demonstrate normality with a larger sample size. However, given that we did not have any proof to support this assumption, we used the independent 2-group Mann-Whitney U test (a nonparametric test) to compare the means of the two groups. We specifically tested for the case where the mean of the LaTeX group is less than the mean of the PowerPoint group. The resulting p-value of 0.1392 does not permit us to make any claims about the difference in means. On the other hand, we were also not able to disprove our hypothesis. We believe that a larger quantity of survey participants might help us by either allowing us to demonstrate normality or by increasing the sample size of the LaTeX group (n=6) which was rather small compared to the PowerPoint group (n=25).

3.4.2 Survey Conclusions

Although we did not intend for this to happen, we believe there is a slight bias caused by the participants. There are several indications that many of our participants have a strong technical background. For instance, 81% rated their computer skill level as "very good" or "excellent", 16% of our participants use LaTeX or similar tools for presentations and 44% of participants mentioned source code as a content type they use. It is clear that these findings are not representative for all users of presentation software. On the one hand these participants can be seen as "power users" and the issues they have are concrete and are not likely to be caused by a lack of technical knowledge. On the other hand, we might have missed out on feedback related to the general usability of existing presentation software. We do not believe that this bias causes any major issues. The findings of the survey have been combined with the findings of our literature study, observations and programmatic analysis of PowerPoint documents, providing some balance. Furthermore, even if we address some shortcomings that are only relevant for a subset of the population, we still see this as a valid contribution. It is however important to keep this bias in mind when interpreting the survey results.

We summarise the most important findings that will be relevant again at the end this chapter when the results of the observations, survey, literature study and programmatic analysis of PowerPoint documents are compiled into a set of requirements. All participants use some form of presentation software. However, 26% of the participants augments the software with the use of a blackboard or whiteboard. The median presenter spends about half of their authoring time on visual details such as layout and styling. The issues and shortcomings related to layout and styling are explicitly mentioned by 28% of participants and is the biggest source of frustration. Only 53% of the participants specify ever using a presentation remote and for navigating content 79% of the participants make use of the computer's keyboard and 30% of participants make use of the mouse. This might be an indication for issues related to the presenter's mobility during presentations. Animations are applied by 44% of the participants at least sometimes. To clarify, when we state "at least sometimes", we aggregated the users that selected "sometimes", "often" or "always" and we did not count users that selected "rarely" or "never". This applies throughout this section where we will make this statement multiple times. An important use of animation is to create flow or structure in the presentation and to synchronise the content with the oral narrative (e.g. making parts of a diagram appear in different phases). During the presentation, 72% of the presenters sometimes revisit content that was shown earlier. There are different strategies for revisiting content, including duplicating content on multiple slides, duplicating content with some changes and manually navigating to earlier content. When asked whether participants traverse their content

in a non-linear fashion, 47% said they do this rarely and 23% mention they do this sometimes. However, 9% of participants explicitly mention the lack of support for non-linear navigation as a shortcoming, a possible indication that presenters do not use non-linear navigation that often because of issues.

Most presenters reuse their own (60%) as well as external content (79%) at least sometimes. This is further reflected by the fact that 63% of participants use Google Images and 23% use Wikipedia while creating presentations. Presentations are collaboratively authored at least sometimes by 51% of the participants. We further investigated the importance of presentations and presentation content. For the participants that use presentations for academic knowledge transfer, 79% rank their slides as the most important learning material, more important than for instance textbooks. When considering all participants, 65% share their content afterwards and 30% state that their content serves another purpose after the presentation session, for instance as learning material, technical reports or as documentation. For this reason 64% of participants state that their content is important as reference material. As a final interesting finding we point out that content types such as charts, diagrams, video, source code and mathematical formulas are commonly used. However, 37% of all participants use external chart and diagram tools, 35% use external image editors and 12% use external IDEs to pre-process presentation content. To us this indicates that although these content types are very much needed by users, integrated support for these content types might be lacking. This assumption is further strengthened by the fact that even though we did not hint towards any issues in the questionnaire, 21% of all participants explicitly mentioned issues related to charts and diagrams. Furthermore, 19% mentioned issues related to video, 16% mentioned issues related to source code and 12% mentioned issues related to mathematical formulas. Other sources of frustration are the need to use many different applications to create a single presentation (7%), the need for better cross-platform support (7%) and the need for better features related to the reuse of content (12%). We would further like to mention one anonymous quote that stood out in the survey results: "*Powerpoint is a scourge from hell and has ruined an entire generation that will never understand how to give a proper presentation*".

3.5 Summary of User Needs, Preferences and Requirements

In the previous sections of this chapter we have already presented some findings that can be interpreted as shortcomings and unmet user needs. We briefly summarise the shortcomings and unmet user needs obtained from the literature study, observations, survey and programmatic analysis of PowerPoint documents. As we aim to keep the

summary short and to the point, we refer back to the previous sections in this chapter for the concrete references and findings that back up our claims.

Improving Support for Specific Content Types and Dynamic and Interactive Visualisations

It is clear that some content types are difficult to visualise with the existing presentation software. From the survey we learn that many presenters create some of the presentation content externally and integrate it as an image. We also confirmed this in the analysis of PowerPoint documents, where the majority of presentations used charts that were created externally and embedded as images. The survey confirms that presenters often create their content externally because the built-in support is too limited or even non-existent. However, our findings also indicated that creating content externally is not ideal as it takes more time, it requires yet another application, it becomes more difficult to make changes, images sometimes appear blurry due to scaling and externally created content does not match the presentation's style. Concretely, from the survey, observations and document analysis we identified charts, diagrams, video, source code and mathematical formulas as content types that are difficult to work with. Especially when it comes to dynamic and interactive content, PowerPoint offers little to no support. Presenters are often required to use time-consuming workarounds such as creating many different charts to simulate animation, or they have to resort to using external tools. However, using external applications during the presentation disrupts the flow and the need for more than one application was another source of complaints in the survey. In our observation, survey and document analysis we encountered many cases of dynamic charts or diagrams that were painstakingly assembled as as series of hundreds of preprogrammed animations. One presenter who was part of our observations went as far as creating their own presentation tool in order to support the dynamic content they needed. Literature from information visualisation and cognitive psychology demonstrates the beneficial results that could be obtained with proper tools. However, unless these tools are provided, many presenters will not make the effort to make good visualisations and they will resort to less effective solutions.

Helping Presenters to Improve their Performance

In the literature study we have shown that businesses see presentation skills as a critical skill to have. More than that, businesses have an explicit need for better presenters. A good presentation has many facets so there are also different ways to provide potential solutions. For instance, providing feedback on a presenter's oral performance or content might bring benefits. Helping presenters to create good content that is easy to process can also improve their performance during a presentation.

Non-linear navigation and improved presenter views could further contribute to a better flow of the presentation. In the observations and the survey we have also seen that time management can be an issue and even disrupt the presentation. Therefore solutions that help presenters to practice the timing and flow of their presentation beforehand can improve their performance. Finally, the observations and survey indicate that presenters often navigate their content with the computer keyboard or mouse, and therefore presenters are restricted in their mobility which might negatively impact the perceived performance.

Helping Presenters with Slide Design

The literature study and our survey show that many presenters see slide design as a tedious and time-consuming process. Our survey shows that the average presenter spends at least half of their authoring time on visual details. Furthermore, we saw that issues related to layout and styling were the primary source of complaints and were explicitly mentioned by 28% of the participants. For this reason an effort should be made to reduce the time and effort needed to create effective presentations. Presenters should focus on the actual content itself and should not spend most of their time on visual details. However, the literature study, observations, survey and document analysis have shown that different users have very different requirements. Slide design is influenced by the use case, the type of content, the academic field or industry segment, and as shown by our document analysis also by culture. Therefore special care should be taken not to enforce specific guidelines as they do not apply to all users. However, there are some guidelines that are universally applicable. As an example, some guidelines are created from a psychological perspective, and these guidelines are provided to make optimal use of the capabilities of the human mind [98, 158].

Improving Support for Reuse, Sharing and Collaboration

It is clear that reuse and sharing is an important part of the authoring process. Most presenters reuse their own (60%) as well as external content (79%) at least sometimes. The survey shows that the web browser, and Google Images in particular, is the most used peripheral tool while creating presentations. Our survey as well as the studies we found as part of the literature study confirm that there is a need for improved tool support for reusing content. In particular, we saw the need for reusing content without duplication, finding related external content, the seamless visual integration of external content, versioning, user access management and the ability to keep track of the source origin, ownership and usage rights. However, in the same way that users have different requirements for slide design, literature also shows that different categories of users have different requirements for content reuse. Our sur-

vey finds that presentations are collaboratively authored at least sometimes by 51% of participants, and the literate study further confirms the importance of supporting the remote collaborative editing of presentation content.

Involving the Audience and Improving Communication

One major critique on presentation software is that it encourages a uni-directional flow of information. Usually the presenter does all the talking and the audience members simply listen. In our observations we saw that unless the presenter makes a big effort to include the audience, the audience members will remain passive. The desire for better communication was also brought up by some survey participants, but existing solutions were perceived as troublesome. Findings from research domains such as pedagogy show that interacting with the audience and also allowing the audience to interact with the content can be a very effective way of transferring knowledge. However, our presentation tools should provide better support for these concepts before they can be applied to their full potential.

Providing Additional Interactions and Mobile Solutions

We have already mentioned how the majority of presenters still use the keyboard and mouse for interacting with the presentation. The observations and survey results show that this restricts the mobility of the presenter. For this reason we aim to investigate additional methods to interact with the presentation content so that presenters can be more mobile. Our observations and survey reveal that many presenters make use of a blackboard or whiteboard for answering unexpected questions when the predefined presentation content did not provide an answer. First of all, given that these presenters choose to use the blackboard or whiteboard over using their computer indicates that some interactions, such as writing or drawing, are not optimally supported in existing presentation software. Second, this can also indicate that presentation software lacks facilities to quickly find and show related content that was not originally part of the presentation. Third, the blackboard or whiteboard is sometimes used to persist some additional content next to the changing presentation content. It has been shown that audiences benefit when key ideas persist on the screen [186]. As physical blackboards and whiteboards are slowly being replaced by other technologies, it might be useful to bring the affordances of physical boards to digital presentation software. Persisting information on the blackboard or whiteboard can also be seen as a way to revisit earlier content. Our survey shows that 72% of presenters sometimes revisits content that was shown earlier. This further indicates that there is a need to bring up other (earlier) content, sometimes unexpectedly. Presentation software could also provide additional methods for quickly navigating to a specific point in the presentation, without disrupting the flow. Improved interactions for non-linear navi-

gation might further make presenters less reluctant to make unforeseen jumps in the navigational path when the need arises, which is an issue described by Farkas [86].

Supporting the Afterlife of Presentation Content

Our survey as well as the literature study shows that some presentation content has a purpose even after the presentation session. For instance, presentation content is often used as learning material or as a form of report. Our survey strengthens this finding as we saw that 30% of participants state that their presentation content is important for later use, and 79% of the teachers saw their slides as the most important learning material, more important than for instance textbooks. However, our survey also found some related issues. For instance, the participants mentioned issues with compatibility across platforms and mentioned that some content looks different on different platforms. In the case of non-linear navigation, dynamic content, audience interactions or annotations during the presentation, much of these events will be lost once the presentation is over. Therefore, presentation software should keep in mind that some presentation content plays an important role after the presentation and all features and functionality should still be available afterwards.

Part III

MindXpres: Conceptual and Technical Foundations

Chapter 4

A Conceptual Framework for Next Generation Presentations

In Chapters 2 and 3 we identified some shortcomings and unmet user needs in presentation software. We further discussed how existing solutions are too rigid to be extended with novel functionality and that more fundamental changes are required to address certain shortcomings. Instead of building on top of the existing paradigms and their limitations, we aim to rethink the concept of a presentation and to rebuild the conceptual foundation. This will allow us to provide proper fundamental support for enabling the potential improvements discussed in Chapters 2 and 3. The goal of this chapter is to gain insight into how presentations are evolving and what kind of concepts and functionality need to be taken into account for a unifying platform and for interoperable solutions. For instance, in related work we see new approaches for visualising and navigating content, context-awareness, sharing and semantics. For this reason there is a need to revisit the concepts that define a presentation and to create a new conceptual framework that incorporates these recent developments. We present a new *conceptual framework* that aims to unify existing state-of-the-art presentation concepts and functionality in a way that is future-proof to a certain degree. The concepts in this framework will later help us to define solutions that are fundamentally compatible with the wide variety of existing approaches and that are also flexible enough to support radically new ideas for addressing existing problems.

Some results presented in this chapter have been accepted for publication in the Proceedings of the ACM on Human-Computer Interaction and will be presented at the 11th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2019) [229].

4.1 Next Generation Presentation Concepts and Functionality

To obtain an overview of the concepts that drive state-of-the-art academic presentation solutions, we compile a summary of novel concepts related to presentations. Most of the related work referenced here has already been discussed in greater detail earlier so details are kept to a minimum.

- **Visualisation Styles** In traditional slideware a presentation is visualised as a linear sequence of rectangular slides. Some tools offer slide transition animations, but they rarely provide benefits other than arguably drawing the audience's attention. ZUIs such as Fly [171] and iMapping [116] use a 2D zoomable canvas to visualise content and navigation between content is visualised as spatial movement (e.g. panning or zooming) to make use of spatial cues. The visualisation aspect of presentation tools not only defines how content is laid out visually, but also navigation and (potential) interactions with content can be visualised differently depending on the tool. Note that the visualisation of content is decoupled from its internal structure. For example, content that is structured as a graph of linked data (internally) can be visualised as both a linear sequence of slides as well as a ZUI.

- **Complex Navigation** Traditional presentations are usually navigated by going forwards or backwards in the slide sequence. However, other approaches offer more flexible ways of navigating a presentation. Related work such as NextSlidePlease [259], HyperSlides [80] and MultiPresenter [164] allow users to create complex non-linear navigational meshes over traditional slides. Tools might automatically create a navigation path based on content such as the audience type or time restrictions. Also ZUIs encourage non-linear navigation as a 2D canvas layout does not impose a natural order on the content the way a linear sequence does. Note that navigation is not necessary driven by interactions in the digital space. For instance, Palette [201] and PaperPoint [251] offer physical paper-based interfaces to support non-linear content traversal.

- **Content Structures** Some presentation tools use specific spatial structures to visualise content (e.g. a chain of slides), but as mentioned before content can also be structured internally for reasons not related to visualisation. For instance, slides might be aggregated into sections and sections can be aggregated into a presentation, forming a hierarchical structure that might, or might not be, reflected in the visualisation of the presentation. There is of course significant overlap with the concepts of visualisation and navigation discussed earlier. As

an example, revealjs.com[1] allows slides to be laid out in a 2D grid to create "side-paths" in the main slide sequence, but this mental structure is also used to visualise and navigate the slides. Similarly, iMapping [116] uses a graph-like structure internally composed of linked data, and the same metaphor is used for visualisation and navigation.

- **New Content Types** Current presentation tools support content such as text, images and video, but overall content is still rather static. There are many commercial PowerPoint plug-ins that, for instance, provide better charts, but some content types (e.g. source code, geographical information or raw numerical data) remain difficult to present. In order not to restrict ourselves later on, it is important to consider that content is becoming more dynamic and interactive, as shown by for instance Slithy [311].

- **Content Reuse** Reusing content does not only mean that we show the same content multiple times in the same presentation as enabled by HyperSlides [80] and MultiPresenter [164]. Content can also be reused between different presentations, but this also comes with some shortcomings that were discussed in Section 2.4.3. One can only reuse presentation content by duplicating it, resulting in various negative consequences. Work such as SliDL [46], SidePoint [176] and even Microsoft Research [74] acknowledge this problem and propose different solutions. It is clear that content reuse should also be taken into account when developing new solutions.

- **Collaboration** The results of the literature study and questionnaire presented in Chapter 3 show that the collaborative authoring of presentations is a common occurrence. For this reason it should be taken into account that multiple people can work on the same presentation content, introducing various complications. The most notable related work that focusses on collaborative authoring includes Google Slides, SlideWiki [145] and Webstrates [149].

- **Semantics** Academic work such as SidePoint [176], iMapping [116] or CPoint [154] explore the benefits of giving meaning to content and associations. As an example, by specifying that a certain piece of text is a theorem, and that it is related to specific topic, our presentation tools could provide additional functionality. For instance, the content would be easy to find via semantic queries such as "show me the theorems related to this topic" or the theorem might be recommended for inclusion when the presenter creates a new presentation on this topic. Presentations could even be composed automatically based on existing content if the system is aware of what it represents [22, 184, 286].

[1] `https://revealjs.com`

- **Context Awareness** There are some systems that take into account that presentations might change their content or navigation based on a given context. For instance, NextSlidePlease [259] allows the navigational path over the content to adapt automatically depending on time constraints. One might also imagine that a presentation could change its structure depending on the knowledge of the audience, or that the presentation might change its colour scheme depending on the brightness of the ambient light. The topic of adaptive and context-aware content has received a lot of attention in the domains of e-learning and adaptive hypermedia, but we also see a lot of opportunities for these concepts to be applied specifically in the context of presentations.

- **Users and Roles** In the classic presentation scenario the classification of users is relatively simple. There is an author, a presenter and an audience. However, if we take current advancements into account, this classification becomes a lot more complex. Because of content reuse and collaboration, content within a single presentation might be owned and authored by multiple users and concepts such as ownership and access rights become more relevant. Audience response systems [141] allow the audience to be involved and enable them to also interact with the presentation content. Furthermore, some systems that for instance offer quizzes will keep track of participants' individual results, which breaks down "the audience" into a group of individual users. In summary, these new systems require us to reconsider the different parties and roles involved during the presentation process, as well as how they can interact with the presentation.

- **Presentation Hardware** Related work shows that hardware related to presentations is becoming more advanced. For instance, classrooms might be equipped with smartboards [254] and multiple projection areas. Audience response systems might require clickers or audience members will use smartphones or laptops to interact with the presentation, as discussed in Section 2.4.6. We also see mobile presenter views which might be based on tablets or digital pens, as explored by PaperPoint [251].

In Table 4.1 we provide a summary of the next generation concepts we presented so far. This summary of next generation presentation concepts acts as set of minimum requirements if we want to support all the existing concepts in our conceptual framework.

	Complex Content Structures	Non-linear Navigation	Complex Content Types	Content Reuse (within presentations)	Content Reuse (across presentations)	Collaboration	Semantics	User Roles and Access Management	Automatic Context Adaptation
Fly [171]	✔	✔							
iMappping [116]	✔	✔					✔		
NextSlidePlease [259]		✔							✔*
HyperSlides [80]	✔	✔							
MultiPresenter [164]				✔					
Palette [201]		✔							
PaperPoint [251]		✔							
Slithy [311]			✔						
SliDL [46]					✔				
SidePoint [176]					✔		✔		
Drucker2006 [74]					✔				
Webstrates [149]						✔			
CPoint [154]							✔		
Moscovich2004 [196]	✔	✔							
SlideWiki [145]				✔	✔	✔		✔	

* NextSlidePlease only adapts content automatically based on time constraints

Table 4.1: A summary of high-level features of academic next generation presentation tools, used as minimum requirements for our unifying content model

4.2 A Unifying Conceptual Framework

In order to unify and conceptually represent the different content, parties, associations and interactions presented in Section 4.1, we have abstracted them in a single generic conceptual framework for next-generation presentation solutions. This framework is generic enough to cover the paradigms and features seen in commercial tools and related work, and will serve as a foundation for our own solutions presented in later

chapters. We show a diagram of the conceptual framework in Figure 4.1. The five conceptual layers of the framework are the *Users*, *Content*, *Structure*, *Semantics* and *Navigation* layer. Note that some concepts such as visualisation details and context are not explicitly shown but they are taken into account and are motivated later in this section.

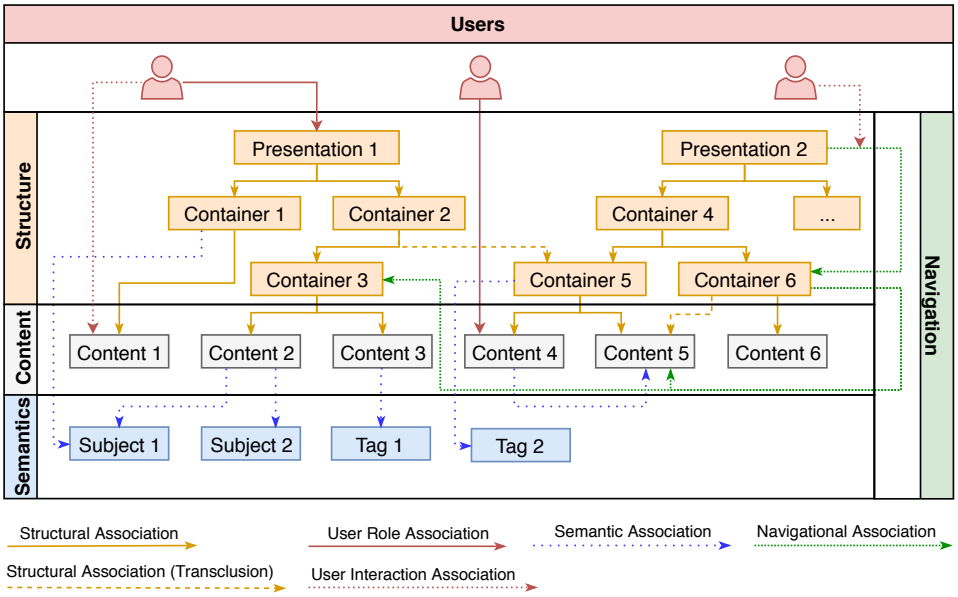


Figure 4.1: The five layers of the conceptual framework: *Users*, *Content*, *Structure*, *Semantics* and *Navigation*

First of all, the *content layer* represents individual pieces of content that form the basic building blocks of a presentation. These could for instance be fragments of text, images, source code or raw numerical data. Next, the *structure layer* provides a way to group pieces of content into structures. In the most simple case the content is aggregated into slides, and slides are aggregated into a presentation. However, our framework allows any kind of multi-level content aggregation by using the concept of nestable containers. For example, containers could be used to express that some image content is grouped into an image gallery container, which is part of a slide container, which is part of a presentation. The same concept can also be used to express nested spatial regions on a 2D canvas for a ZUI, or to collect slides into "chapters" for a book-like presentation structure. In order to compose any of these presentation structures from a set of content and containers, we define a specific type of association, the *structural association* (represented by solid orange arrows in Figure 4.1). In other words, these structural associations are the "glue" that tie content to containers, and containers to other containers, to form a graph-like presentation

structure. Content might be part of multiple containers or even presentations if it is included via more than one structural association. A container and its entire substructure might be reused elsewhere without duplication. This allows for instance an entire slide, or an entire chapter to be reused in another structure. In other words, content can be integrated in other presentations by simply pointing to it, rather than creating a duplicate, which was coined as "transclusion" by Ted Nelson [202] in the context of hypermedia. Under the assumption that the content is easily available across computers, complex presentation structures and substructures can be built on top of potentially distributed content. By enabling presentation structures that span multiple content repositories, we also see possibilities for more efficient sharing and collaboration, on a fine granularity. Figure 4.1 shows transclusion as dashed orange arrows and we can see how it can be used to reuse a piece of content in multiple containers, or how a container is used in multiple presentations. This type of flexible reuse and sharing introduces new issues related to, for instance, unwanted changes, ownership and access rights. To keep the conceptual framework generic and abstract, these issues are not included here, but they are addressed in Chapter 5 where we transition into a concrete information system based on the presented framework.

Next, the *navigation layer* is used to represent concepts such as *navigational paths* or pre-defined state changes in a content visualisation. Navigational associations are provided to link content or containers for navigation, and complex graph-like navigational structures as seen in NextSlidePlease [259] or HyperSlides [80] are possible. These associations may have multiple sources and targets, which can be fragments of content but also containers. In the most simple scenario navigational associations mimic the navigational links seen on the Web, where the tool navigates to a specific content fragment or container such as a slide when activated. However, the same principle enables, for instance, a predefined spatial path on a 2D canvas in a ZUI. Navigational associations can also be used to represent state changes. As an example, different pre-defined states of a data visualisation could be chained into a sequence via navigational associations.

The *semantic layer* is used to represent concepts that give meaning to content, containers and associations. We define the *Semantic Association* (represented by blue arrows) which allows semantic associations between content or containers, but they also allow content, containers or associations to be annotated with semantic metadata. They can, for instance, be used to relate content or containers to a specific subject or they might be tagged with a semantic value such as "proof" or "example". These semantic associations can also be used to directly associate content or containers with each other to represent relationships such as "that other slide contains a more detailed explanation of this text".

The *user layer* abstracts the relationships between users and the components of the other conceptual layers. We define two kinds of associations concerning users, the *Role Association* and the *Interaction Association*. The role association (solid red arrows) is used to specify situations where a user has a certain status in relation to specific content (or other association). Role associations can, for example, be used to represent the fact that a specific user is the author of certain content or containers, or that a specific user is a presenter for a given presentation. The second type of user association, the interaction association, is used to represent possible interactions a user can have with components of the other layers. For instance, certain users might have the ability to trigger navigational links while others might not. Similarly, an interaction association might specify that a user is able to change the visualisation style of some content during the presentation (e.g. switching from a table representation to a chart visualisation). Note that these concepts are optional and they are offered for these cases where this level of expressiveness is required in a content model. Applications might, for instance, provide default interactions without reflecting this in the underlying model, or only special cases that override the defaults might be modelled with the components provided by the user layer.

Last but not least, we would like to mention context and contextual reasoning. This is not represented as a separate layer because context is determined at runtime and it can effect every aspect of every layer shown in Figure 4.1. As such, it applies to semantics, content, structure, users and all the related associations. Based on the context, any of these components might change. As an example, context might determine if a user interaction is possible or not, a user's role might change, the structure of the presentation or navigational path might change, specific content might become inaccessible or the semantics of certain content might change. Context and context adaptation are core components of the conceptual framework and should be kept in mind at all times, but they do not have their own distinct layer.

When comparing the five layers in the conceptual framework with the categories of functionality provided in Section 4.1 one might notice that some categories (other than context) are also not literally represented in the framework. The main reason for this is that they are already covered by abstractions in layers created for other categories of functionality. For instance, collaboration is covered by concepts provided in the structure and users layer (e.g. transclusion, interaction and roles). Presentation hardware is covered by the user layer, as this hardware is just an interface to provide interactions with the presentation in one way or another. We would further like to clarify the absence of concepts related to layout, styling and visualisation. Given how our framework makes it easy to reuse and seamlessly integrate content, it is important to keep content and visualisation-specific details separated. We argue that visualisation-specific concepts are a part of the Structure layer. Layouts and

style-related metadata (e.g. fonts, colours or absolute positions and offsets) can be different every time a node is included in a presentation structure, so it should be attached to the structural association that makes content or containers part of a presentation. This way, style-related metadata is not associated with content or containers directly, and different metadata can be applied every time they are reused. It can also be argued that style and layout only become relevant at runtime and not on the conceptual level.

By basing ourselves on related work we have generalised and unified existing next generation presentation concepts and to some degree also concepts that have yet to come. In contrast to existing presentation software our goal is to prevent design choices that will place restrictions on solutions later on. The resulting conceptual framework is generic enough for a broad spectrum of new presentation solutions and the framework does not make any major assumptions. Presentations and their navigation can take any form and we incorporated concepts related to reuse, sharing, users, semantics and context awareness. This research artefact acts as a conceptual foundation for the artefacts that will be presented later in this dissertation. By adopting the framework's concepts in our later contributions we make sure that we maintain the targeted degree of freedom for all of our later solutions.

Chapter 5

A Hypermedia-based Content Model and Information System

With the goal of building a new foundation for next generation presentation solutions, we continue with the next step. Before we address high-level issues related to user interactions and visualisations, we need to think about how presentation content and structures are modelled and stored in the lower levels of the presentation stack. In order to fulfil some of the needs presented in Chapter 3, we need to provide support at the data level. We further need to ensure that we also support the concepts defined in our conceptual framework in the data layer, so that support for these concepts can propagate further to higher levels. As we will show, existing content models cannot represent all the concepts defined in the conceptual framework. For this reason we define a new content model, based on the RSL hypermedia metamodel. By addressing concepts such as content reuse, collaboration, semantics and context resolving directly in the core of the underlying data models, we enable richer solutions higher up the presentation stack. In this chapter we also discuss a concrete implementation of this new information system which will be used as a working back-end for our implementations of the upper layers of the presentation stack.

The results presented in this chapter have been accepted for publication in the Proceedings of the ACM on Human-Computer Interaction [229] and will be presented at the 11th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2019).

5.1 Existing Document Formats and Content Models

In order to motivate the need for a new content model we discuss some of the related work in the domain of document formats and content models.

5.1.1 OOXML, ODF and Similar Presentation Document Formats

We start with the most obvious choice, the PowerPoint document format. PowerPoint (and similar tools) store their content in monolithic documents which makes it difficult to reuse content without duplication. The structure of these PowerPoint documents is defined by the PresentationML[1] standard, which is part of the Office Open XML (OOXML) standard that also provides specifications for other Microsoft Office documents. The main reason that PresentationML cannot easily be used for our purpose is that it is tailored specifically for PowerPoint and it is not generic enough. As an example, Figure 5.1 shows how PresentationML is hardwired to define a presentation as a set of slides, and as such the format only supports the paradigms enforced by PowerPoint. Apache's OpenOffice Impress presentations are based on similar XML specifications, the Open Document Format for Office Applications[2] (ODF). ODF is slightly more generic with its semantics and specifies that a presentation consists of a series of "pages". However, other features of the specification make it clear that also ODF-based presentations are modelled specifically for the PowerPoint paradigm. Concretely, presentations in ODF are again hardwired as a set of ordered slides and have little to no support for next generation presentation concepts. Both PresentationML and ODF only support a fixed set of content types (e.g. text, images, formulas or charts) and lack support for representing the complex structures, navigational paths, semantics and user management we require. One might argue that we could use generic OOXML (ignoring the PresentationML subset) or ODF as a foundation and that we could have extended it with the concepts we require. While this is technically true, there are two reasons why we did not do so. First of all, so much would have to be changed and added that the effort required would be almost the same as creating a new document format from scratch. Second, we aim to avoid the shortcomings caused by using monolithic XML-based documents for storing presentation content. As such, we argue that XML is not the most effective technology for achieving our vision on content reuse and collaboration, potentially across different systems. Therefore the argument of being able to use existing XML tools (e.g. parsers and validators) and standards (e.g. DTD) does not apply. There are also some HTML-based alternatives such as the Simple Standards-

[1] `https://docs.microsoft.com/en-us/office/open-xml/structure-of-a-presentationml-document`

[2] `http://docs.oasis-open.org/office/v1.2/OpenDocument-v1.2.html`

based Slide ShowSystem (S5)[1] and the W3C's Slidy [217], or markup languages such as slideDeck.js [272], but all our previous arguments also apply to these formats.

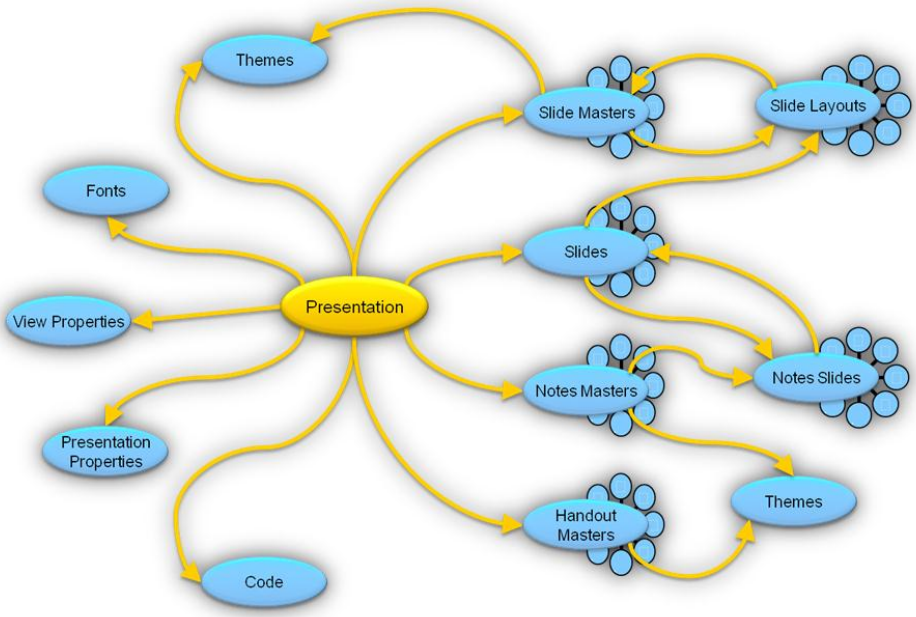


Figure 5.1: Structure of a PresentationML document (Open XML SDK), as shown in the official documentation. Source: `https://docs.microsoft.com/en-us/office/open-xml/structure-of-a-presentationml-document` (last checked 13/12/2018)

5.1.2 Learning Material Content Models

Other than the more technical document formats discussed above, in related work we did not encounter any formal models made specifically for presentations. For this reason we expanded our scope to the domain of learning material in general, and we investigated other conceptual frameworks, markup languages and content models. Given that they are built for general learning, they often describe pedagogical concepts that are not relevant to us (e.g. activities, learning objectives or assessment). Even though none of the models could be directly applied for our use, we gained insights on how they deal with concepts such as structural aggregation, semantics or reuse. We investigated projects such as PCML [160], eLML [92], SCORM [132], NETg [163], NCOM [2], Learnativity [292], CISCO RLO/RIO [23], dLCMS [240], New Economy [178], PaKMaS [263] and LMML [262] as well as SLM [87]. It would

[1] `http://meyerweb.com/eric/tools/s5/`

be redundant to describe and compare these projects in detail given that detailed summaries and comparisons have already been made by for instance Verbert et al. [285] or Yassin [301]. We rather summarise the findings that are relevant to our goals and that will help drive the requirements for our content model.

Each of these models for learning material is unique and has distinct characteristics, but the general ideas are more or less the same. The models define a set of content types such as text, images or videos and aggregate them into multi-level hierarchies. For instance, a model might specify that content is aggregated into chapters, chapters are aggregated into a lesson, and lessons are aggregated into a course. It is interesting to point out that these models allow for reuse at different levels of the aggregation hierarchy. For example, one can reuse an entire lesson, a specific exercise or just a single image. Most models also name their content types or their aggregations (e.g. "exercise", "definition", "example", "story" or "course"). These semantics play an important role for interoperability and indexing, enabling applications such as content recommender systems and semantic search engines. Interoperability between semantic models is further explored by ALOCOM [285] which defines a mapping to a generic model, unifying most of the above models for information exchange. In the context of semantics and interoperability it is also important to mention the IEEE Learning Object Metadata standard (LOM) [1]. LOM defines syntax and semantics to further annotate learning material to enhance interoperability and indexing, and is used by for instance the SCORM content model.

5.2 Towards a Unifying "Future-Proof" Content Model for Presentations

As motivated earlier, existing document formats and content models cannot easily represent the functionality we described in previous chapters and the definition of a new model is required. Some fundamental shortcomings of existing work can be addressed directly in this new content model if designed properly. New contributions at this level will also propagate up to higher levels of the presentation stack and offer potential for new functionality. In this section we specify which functionality and concepts we should provide directly in the content model and why. In other words, the following list can be seen as a set of requirements that will also be used to evaluate the model later on.

5.2.1 Model Requirements

Model Requirement 1: Extensible and Flexible Core Components

One characteristic of content models for learning material is that a lot of concepts and semantics are predefined in the core. For instance, some models limit content types to only text, images and video. However, this is a potential issue for our goals as content is becoming richer in general, and we also aim to introduce new content types in our solutions. For this reason it would be a mistake to predefine and limit the supported content types. Instead they should be extensible to meet future demands. From learning material we also see that multi-level content aggregation should be supported in our model. Presentation content is, for instance, aggregated into slides, which are aggregated to form a presentation. However, we intend to support more paradigms than those offered in traditional presentation software. For instance, in a ZUI content might be aggregated into nestable "regions" on a canvas, grouped into titled "sections". To support this form of presentations our aggregation hierarchy needs to be able to be able to take on any form, potentially with many aggregation levels, and needs to support a wide variety of aggregation types such as slides, regions or sections. Also here it does not make sense to hardwire these semantics in the core if a wide range of use cases is desired. The Learnativity [292] model is one of the few models that takes a more generic approach. It defines abstract concepts such as a "collection", which can be extended to define aggregates such as stories or courses. Similarly, content types and semantics can be extended. However, Learnativity still imposes a fixed number of aggregation levels and provides restrictions on the order of the levels, which should also be avoided in our model. To summarise, our model should be able to represent all existing approaches, including state-of-the-art concepts presented as part of the conceptual framework, but up to a certain degree it should also be flexible enough to adapt to approaches that have not been thought of yet.

Model Requirement 2: Fundamental Support for Reuse and Collaboration

The need for reuse and collaboration has already been motivated in Chapters 3 and 4. We aim to address related issues in the content model so that it does not become an afterthought with reduced functionality because of restrictions in lower levels of the presentation stack. This requirement is more complex than it seems because it immediately triggers additional challenges and brings up new requirements. For instance, to avoid issues the *separation of content, structure and styling* is implied and content should be *reusable at a fine granularity* (e.g. not only complete slides, but also fragments of text or a smaller region of an existing image). Proper support for reuse and collaboration also introduces some smaller requirements such as interoperability, ownership and user access rights [188], which are classified under some of the other requirements that are discussed next. Finally, if reuse by transclusion is offered, it

also raises the question "*If the referenced content changes, should this be reflected in all the documents that use it?*". Also here we state that our model should be impartial and it should allow users to choose if this is desired or not. In order to support the option where documents should not show updates of transcluded content it is necessary to support a form of *versioning*. This would allow a user to transclude a specific version of a content fragment or bigger structure, and even if they change, the older version remains available for reuse. Note that transclusion on a small granularity saves disk space since content is only stored once, even if it is part of multiple documents. The proposed versioning would be applied on a fine granularity such as individual content fragments and associations, making it much more efficient than versioning at the document level. For instance, when keeping track of a PowerPoint file using the Git version control software, it would store a new copy of the entire PowerPoint document even if only one character is changed. In contrast, our system would only copy the small fragment that was changed, and keeping track of previous versions could be further optimised by using for instance delta compression in the implementation.

Model Requirement 3: Extensible Semantics and Semantic Interoperability

In addition to the motivation in Chapter 4, the models for learning material have shown the importance of giving meaning to content and associations. It is clear that for the sake of interoperability and improved reuse, semantics should be a core component of our content model. However, we draw another important lesson for the learning material models. Even though many of these models use the LOM standard for semantic metadata, there is no real consensus on the vocabulary used to name specific content units or their aggregated forms. This is illustrated by for instance Verbert et al. [286] who motivate and present a mapping between existing models. For this reason we conclude that introducing yet another ontology for our use case would not make much sense. Instead, we set the requirement that users of the model should be able to introduce their own semantics. This might be one of the existing ontologies but could also be a new one. As such, our model remains interoperable with existing models instead of trying to enforce yet another ontology specifically for our use case. This approach requires a certain degree of extensibility and flexibility in our model so that different ontologies can be applied or mapped, if needed. As shown by our Model Requirement 1, this is in line with our view on the model as a generic and extensible platform. By allowing our model to be interoperable with others, we also offer the possibility for including the plethora of existing learning material in our presentations. Additionally, this might lead to a lot of new functionality related to content recommender systems or semantic querying. For instance, it could lead to improved tools for the (semi-)automatic assembly of presentations as explored by [286, 22, 184, 175]. Related work has also shown the

benefits of finding reusable content via contextual cues or episodic memory [119, 246, 140], and semantic annotations can be used to augment presentation content with this metadata.

Model Requirement 4: User Roles and Access Management

Also some concepts related to *users*, *groups of users* and *roles* should be incorporated in the content model to avoid issues higher up in the presentation stack. Current systems might keep track of ownership on the document level, but considering our vision we would need to store user-related metadata on a much finer granularity. As a presentation might consist of many reused content fragments from various sources, it is important to keep track of who created what, who owns what, and who made changes to what. In order to enforce *privacy* and *security* in the implementation there should also be a notion of access rights, so that owners could decide who can view, change and reuse their content. Note that this does not only apply to content fragments, but also to associations such as structural association or navigational paths. We refer back to Chapter 3 where we presented some studies that explicitly state the need for presentation solutions that deal with issues related to ownership and that better keep track of the origin and history of content.

Model Requirement 5: Complex Navigational Structures

In some content models the order of navigation is for instance implied by the order of the content. Other models do incorporate the concept of navigational paths but usually they are limited to navigational links such as those found on the Web. In order to support the complex navigation detailed in Section 4.1, we need to incorporate the option of complex navigational structures in the document model. In order to support complete navigational graphs within or even across documents we require navigational associations with multiple sources or targets. These targets and sources could be larger structures, individual content fragments or even a small selection of a content fragment. The user access metadata described above could prevent users from resolving certain links or the navigational graph might even adapt based on context.

Model Requirement 6: Context-aware and Adaptive Presentations

Last but not least, we require support for context-aware and adaptive presentations as described in Section 4.1. The need for context-aware content delivery is often mentioned [241] but has barely been explored in presentation tools. Some aspects such as how context is determined and the logic for context-aware reasoning is a runtime matter. However, the fact that context might influence various parts of a presentation should be represented in the document model. For instance, depending on a given context, the document structure, content, navigation or semantics might

be adapted. Content adaptivity is or course not a new concept and has previously been explored in domains such as e-learning or adaptive hypermedia. Nevertheless, we argue that the benefits of adaptivity also apply to presentation solutions.

The above requirements influenced the choices we made for our content model and at the end of this chapter we go through them again to show the concrete features of our model that help us to fulfil these requirements.

5.2.2 The Original Hypermedia Vision

It is evident that our approach for representing presentation content is based around linked data and inspired by the original hypermedia vision pioneered by Ted Nelson and Douglas Engelbart. The hypermedia concepts we see on the Web are a simplified version of the original vision and it is important to point out the relation of our requirements to the original hypermedia vision, which was richer but also more complex. The reuse of content without duplication (transclusion) and complex navigation were described by Ted Nelson [202] before the Web existed, as a way to go beyond the restrictions of physical documents. Similarly, Douglas Engelbart's landmark computer demonstration, often called "The Mother of All Demos" [84], showed practical applications based on hypermedia concepts, including a text editor with support for real-time collaborative editing. We also see that concepts from more recent academic domains such as semantic hypermedia and adaptive hypermedia have been applied successfully for e-learning, and also align well with our own goals and requirements. For this reason, we base ourselves on existing hypermedia concepts and the content model is presented as a hypermedia model. Reasons for not using for instance the Resource Description Framework (RDF) for linking data include the fact that we require more complex bi-directional associations, with multiple sources and targets which might in turn even be associations themselves. Although this is technically possible in RDF it would make the model unnecessarily complex while existing hypermedia models provide some higher-level constructs we need. Our hyper-media based model does also not prevent the run-time implementation of the model to be based on RDF and related technologies, but as will be motivated later this is also not the case and a custom implementation of the information system is detailed at the end of this chapter.

5.3 The Resource-Selector-Link (RSL) Hypermedia Metamodel

Our content model is presented as a domain-specific application of the Resource-Selector-Link (RSL) hypermedia metamodel [250], which is detailed in this section. However, before doing so, we would like motivate our choice. The hypertext and hypermedia research community has produced several models and conceptual frameworks that attempt to define the necessary abstractions for hypermedia systems. One of the earlier models is the Dexter Hypermedia Reference Model [115], which was often used as a starting point for later extensions. For instance, the Amsterdam Hypermedia Model [117] adds concepts related to time and chronology, AHAM [65] adds concepts related to adaptation and users, and the DeVise Hypermedia Model [111] addresses other smaller shortcomings. However, Signer and Norrie argue that "*most hypermedia models and systems are isolated solutions for specific domains (e.g. navigational or spatial hypertext) or even specific applications*" [250]. Compelling arguments for a unifying model are provided and the RSL hypermedia metamodel is presented to address some of the limitations. As a metamodel, RSL provides abstractions for creating domain-specific hypermedia models. However, in contrast to other hypermedia models, concepts such as context and users are defined in the core of RSL and can be applied to any of the model's components. Links are treated as first-class citizens and as such, also benefit from these core features. Given the fact that the RSL hypermedia metamodel is highly extensible, offers concepts such as context and users that are relevant for our content model, and has already been applied in various application domains, we decided to base our content model and information system on RSL. It further helped that our research lab has done some active research on the RSL hypermedia metamodel over the last decade and therefore has a rich expertise in the RSL metamodel as well as its implementations. For a more detailed comparison of RSL with other hypermedia models we refer to [250].

We now provide a quick overview of the most important RSL functionality, but again we refer to the original work [250] for more details. First of all, we discuss the core components of the RSL metamodel shown in Figure 5.2. The original metamodel was presented in the OM data model notation [207] but in order to reach a wider audience, we present the RSL metamodel, as well as our own content model as UML class diagrams. The RSL metamodel revolves around three main components: *Resources*, *Selectors* and *Links*. All three are derived from a common parent, the `Entity`, which is a superclass that offers shared functionality. For instance, as `Entity` subtypes all resources, selectors and links have an associated set of key/value properties which can be used to flexibly store application-specific (meta)data. The `Resource` class is used to represent content fragments and has to be subtyped for specific media

types such as text, images or videos. The `Selector` is used to address parts of resources and also has to be subtyped for specific media types. For instance, an image selector would define a specific region within an image resource, while a video selector could define a specific time interval within a video. The `Link` is used to represent various associations between entities. RSL links are bidirectional and can have multiple sources (entities) and multiple targets (entities). Note that because links are also entities, links may point to other links.

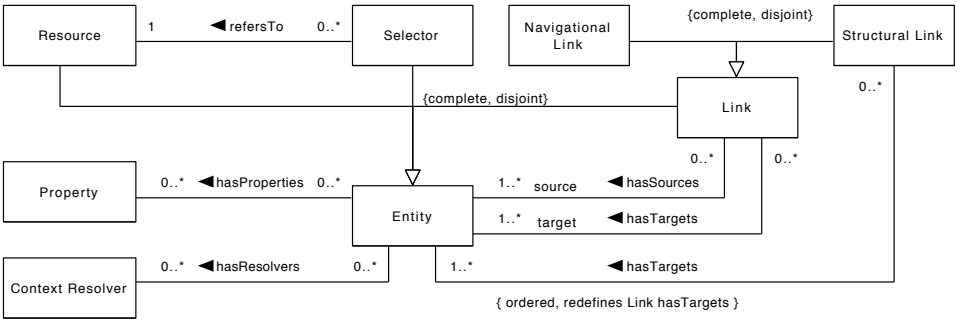


Figure 5.2: Core RSL components

The core model pre-defines two common subtypes of links, the `Navigational Link` and the `Structural Link`. Structural links are used to aggregate content into larger structures. For instance, text and image resources might be combined into a "chapter" by using a multi-targeted structural link. Targets of a structural link are ordered so that there is an implicit order in a structure. For instance, the order of the chapters in a book can be encoded in the structure by using the order in which the chapters are targeted by the structural link that ties the chapters into a book. Of course users are also free to ignore this implicit order and introduce their own ordering mechanism (e.g. via metadata). The navigational link is used to create navigational associations between resources, parts of resources (via a selector) or structures. Given how RSL links are an `Entity` subtype with all associated benefits, together with their ability to have multiple sources and targets, complex navigational graphs can be created. For other types of links that might be required, the `Link` class can easily be extended to create other kinds of typed links (e.g. semantic links).

Last but not least, the RSL core provides functionality for the context-dependent handling of entities. An `Entity` can optionally be associated with context resolvers that will determine whether the entity is available or not in a given context. At runtime, implementations of these context resolvers might use other data handled by the RSL metamodel as well as external data to determine if the entity is accessible or not. This mechanism allows resources to adapt to context, but given that links

are also entities the concepts becomes even more powerful. For example, context resolvers on structural links allow the document structure to change depending on external factors, and similarly resolvers on navigational links allow the navigational path to change.

In addition to these core components, RSL also defines concepts related to users and user access. Figure 5.3 shows these user-related components of the RSL metamodel. An entity is associated with its creator, which is always a single individual. For each entity it is possible to set which users, or groups of users, are granted or disallowed access. This combination of blacklisting and whitelisting strategies allow for powerful access management such as "*that group but without these users*" or "*this set of users unless they are part of that group*".

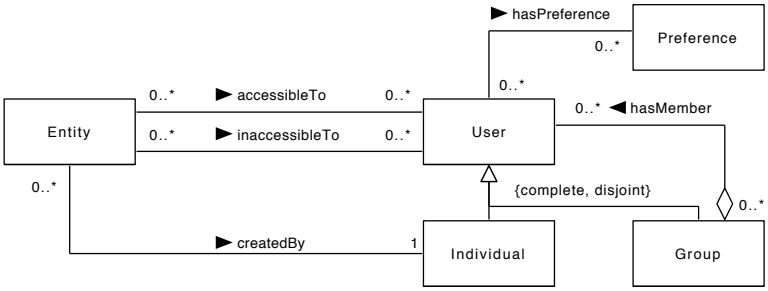


Figure 5.3: User-related RSL components

5.4 Extensions to the RSL Hypermedia Metamodel

It is clear that core RSL components related to structure, navigation, users and context enable us to port concepts from our from our conceptual model (see Chapter 4) to a content model. However, we also needed some functionality that is not offered in the original core RSL metamodel. For this reason we have made some small extensions to the core. For instance, we argue that is necessary to approach versioning directly in the core of the RSL metamodel instead of implementing versioning as an afterthought in a higher layer. By integrating versioning in the core we can offer versioning for associations and semantics, and not only for content fragments. Furthermore, as a core RSL concept, versioning would be interoperable with other core features of the model such as user access management or context resolvers.

Entity-Level Versioning

If transclusion is made possible, it might be undesirable that a document changes if some transcluded content is updated by the original author. In order to address

this issue, we propose a basic form of versioning so that users can choose between always transcluding the latest version, or including a specific version and ignoring any updates. Versioning should not be limited to resources but there is also benefit in versioning selectors and links (e.g. document structures or navigational paths). For this reason we added a simple form of versioning at the level of entities, in the RSL core. The idea behind our extension is simple yet powerful and is illustrated in Figure 5.4.

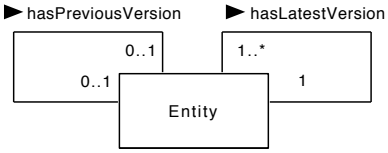


Figure 5.4: Versioning of RSL entities

When an entity is updated, a copy is created and the new version is associated with the old version via the `hasPreviousVersion` association. Over time this builds a doubly linked list of versions for a specific entity as shown in Figure 5.5. Note that even with the given semantics, the `hasPreviousVersion` is bidirectional and the list can be traversed forwards and backwards. However, for convenience, older versions are directly associated with the latest version via the `hasLatestVersion` association to avoid traversing the entire list. The order and the numbering of the versions is implicitly encoded so it is possible to retrieve a specific iteration of the content. However, this iteration is a rare use case and if users want to include a specific version, they should link directly to the version of the entity that they require. If the latest version is required they are able to specify this so that the system automatically follows the `hasLatestVersion` association and resolves the request to the latest version of the entity. For this reason the latest version points to itself with a `hasLatestVersion` association.

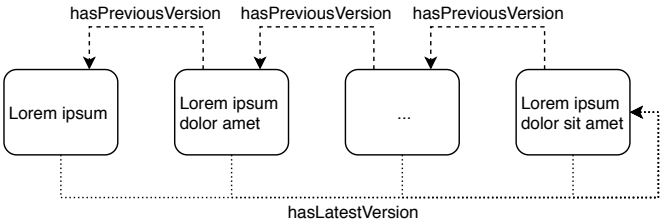
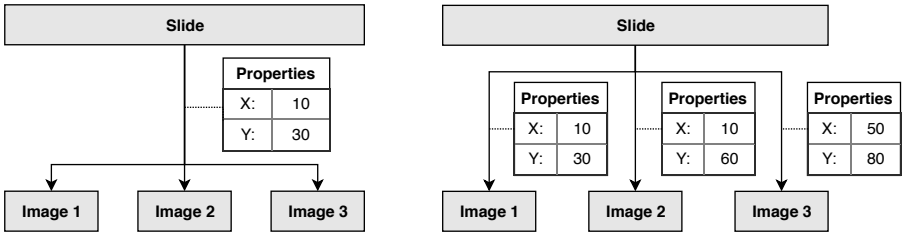


Figure 5.5: A doubly linked list of versions with implicit numbering

Source and Target-specific Link Properties

As an entity subtype, links can be associated with a set of properties. As is, this means that all sources and targets of an RSL link share the same set of properties that has been defined on the link. However, we foresee situations where it is desirable to have a different set of properties for each source or target. As an example, in order to allow content to have different styling and layout parameters every time it is reused, this data should be associated with the structural link and not with content itself. This way, every time content is included in a document with a structural link, different styling and layout parameters can be applied. Based on this example, we illustrate the restrictions of the current model in Figure 5.6a. Figure 5.6b shows the same example but with a different set of properties for each target, a feature that should be made possible in the RSL metamodel.



(a) Original RSL link properties (b) Target-specific RSL link properties

Figure 5.6: Adding support for separate link properties for each link target

To address this limitation we make a simple addition to the core model. Next to the properties associated directly to the link we also allow properties to be associated separately for each target. Figure 5.7 shows that this is achieved simply by associating properties to the `hasTargets` relationship. Note that general properties on the link level are still supported. Target-specific properties can also be reused without duplication if some properties are shared by some targets, but not all. A similar extension was made for allowing different properties for every source.

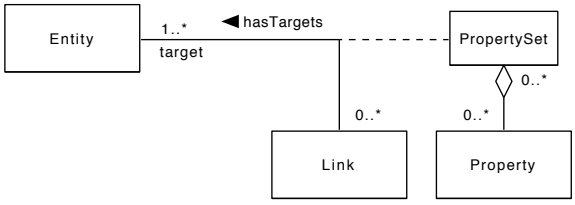


Figure 5.7: Extending RSL for target-specific properties

5.5 The MindXpres Content Model

After setting the requirements and introducing RSL, we present our new content model for presentations as a domain-specific RSL application. We present our model in Figure 5.8, with the core RSL components shown in white, presentation-specific extensions and changes in green and examples of application-specific extensions in red. In other words, the red components are the components that should be added or changed by developers to meet the requirements of their applications. In order to keep the model readable, we leave out some of the core RSL components shown earlier and only components that were modified or added are shown. However, all concepts and associations shown in Figure 5.2 and Figure 5.3 still apply.

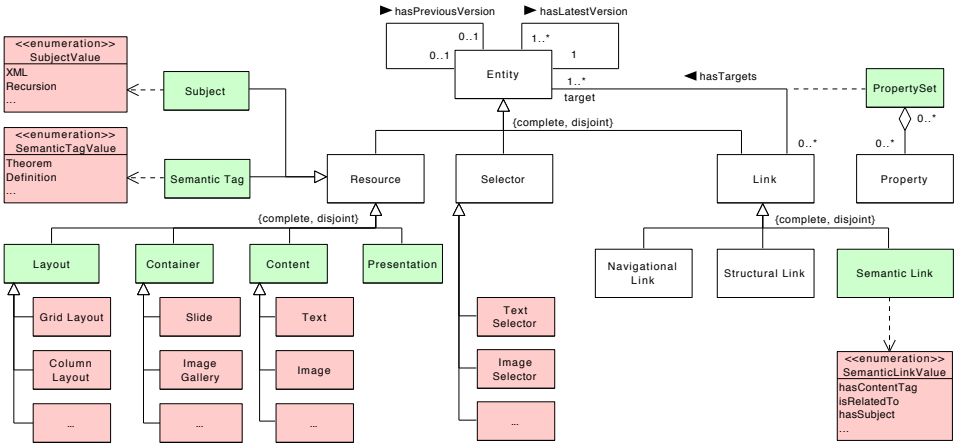


Figure 5.8: The MindXpres content model as a domain-specific RSL application

The basic content blocks (e.g. text or images) of a presentation are represented by the `Content` class, an RSL `Resource` subtype. Depending on the needs of the user, the `Content` class should be subtyped for each content type that needs to be considered. To aggregate content into larger structures we provide the `Container` class. Also this class should be subtyped to provide typed aggregations for content or containers. For instance, in the case of a classic slide-based presentation, the slide should be modelled as a `Container` subtype. Other examples of `Container` subtypes include chapters, sections or spatial regions (e.g. for a ZUI). Note that a bullet list can also be modelled as a container as it is an ordered aggregation of text fragments or sublists. To assign objects to a specific container, the `Structural Link` should be used. Concretely, this is achieved with one `Structural Link` instance with one source (the container instance) and multiple ordered targets (e.g. content instances or other containers). Containers can also contain other containers, allowing multiple levels of aggregation for a wide range of presentation types. For instance, content

can be grouped into slides, but slides might further be grouped into sections. This also allows slides to contain subcontainers such as an image gallery, a container for image resources. Containers should be subtyped to match the required semantics, but the highest level of aggregation, a presentation, is pre-defined. The reasoning behind this is that no matter how the model is further extended, the result is always a presentation.

A `Container` can only contain content, selectors or other containers, but this constraint could not easily be modelled in UML as children are not directly associated to the container, but via an instance of a `Structural Link`. Since structural links are also used to create other structures, and not just between for instance resources and slides, the constraints cannot be modelled on the structural link. Instead of a structural link subtype for every required combination of sources and targets, we decided to keep the model readable and present these constraints in Table 5.1.

Entity Type	Allowed Targets of the Structural Link	Min	Max
Presentation	{Container, Selector, Content}	0	*
Container	{Container, Selector, Content}	0	*
Selector	∅	-	-
Content	∅	-	-

Table 5.1: Constraints between nodes of a presentation structure

The `Layout` resource provides a construct for reusable layouts. Examples include a `Column Layout` or a `Grid Layout`. When content or substructures are incorporated in a presentation structure, a layout instance can optionally be attached to the corresponding structural link. This allows different layouts to be attached every time the content or substructure is (re)used. Note that the targets of structural links are always ordered and layouts should use this implicit information to position a link's targets accordingly. For instance, it is implied that the second target of a structural link should go in the second row of a `Row Layout`. Layout implementations should not use absolute dimensions, positions or distances as they should be able to adapt to the available space. This way, a layout can also be reused and seamlessly integrated in other presentations even if different container types are used.

Style data, including visualisation-specific data such as fonts, colours and themes should also be kept separate from the content so that content can seamlessly blend into other presentations when it is reused. For this reason styling data should also be attached to the structural link every time the content is included in a presentation structure. This way, different style data can be applied depending on the context. Concretely, the data itself should be stored in the property set associated with each structural link (inherited from Entity). This allows many different styling approaches

to be implemented, including CSS-like systems where styling rules can be inherited from containers higher up in the presentation structure. RSL's `Navigational Link` is used to represent navigational paths over the presentation structure. In the simplest case, navigational links could be used to lead a user from slide to slide to recreate the classic sequence of slides. However, they can also be applied to create complex non-linear presentations. First of all, navigational links can point to any container or content so one could define a path from a slide to an individual image to a "section" container. This is needed to represent navigational paths used in ZUIs such as Prezi. Furthermore, navigational links can have multiple targets and a path may offer different routes that can be followed by the presenter during the presentation or chosen by external factors such as the audience type or time constraints [80, 259]. Finally, sources of navigational links can also be any container or content, so they could be used to implement classic hyperlinks (e.g. the users clicks on a particular word or image to navigate to related material). Many presentation solutions offer transition animations when navigating content. Many presentation solutions offer transition animations when navigating content. Navigating from one slide to another, or from one location in a ZUI to another location, or going from one state of a data visualisation to another state is all modelled with navigational links. Therefore, if for example transition animations are desired, they should be specified in the properties of the navigational links. This might be the name of the effect that should be applied or in the case of zooming and panning in a ZUI the properties might store more complex data such as the velocity and zoom level of the zooming animation.

In terms of an ontology for semantic metadata, we have previously stated that there is no real consensus and it is therefore better to be interoperable with existing as well as future ontologies. First of all, the new `Semantic Link` is used to represent semantic associations. One should be able to type these links to distinguish between associations such as "is related to" or "has subject". However, in contrast to the previously described extensibility, users should not create new types of semantic links by subtyping. The goal is to simply *tag* a semantic link with a specific meaning. For that we deem it unnecessary to create subtypes as no additional functionality or information would be added to any subtypes. Instead, we use the semantic link's property set (inherited from `Entity`) to store the semantic meaning of each instance. The `Semantic Tag` resource is a similar concept that allows users to tag content or structures with semantics such as "theorem" or "definition". These semantic tags are entities themselves and a `Semantic Link` should be used to connect, for example, a content instance with the instance of the semantic tag that represents a "theorem". As semantic links can have multiple targets, content or structures can be associated with multiple semantic tags. RSL links are also bidirectional, so by making a new entity instance for each tag (instead of abusing properties as mentioned earlier) we can easily provide answers to queries such as "find all theorems" by following the associations

from the "theorem" tag to the content or structure. The `Subject` resource works in the same way and allows content or structures to be associated with one or more specific subjects. Semantic tags and subjects are also not extended by subtyping but by setting their meaning in the properties of each instance. The use of entity properties for storing the semantic meaning of `Semantic Link`, `Semantic Tag` and `Subject` instances makes them easy to be extended. As the model does not need to be extended to add new semantic types, there is also no need to make changes to the database schema when new semantics is added and semantic types can be treated in a uniform way in the application layer. More importantly, the chosen design makes our model interoperable with existing content models. An existing ontology can easily be adopted by using the same vocabulary for the semantic link and tag values. Ontologies might also be combined and used in parallel, or a custom ontology might be used and mapped to other ontologies in the application layer. Namespaces can optionally be used to avoid collisions between ontologies. Ontology integration and merging is of course a problem that is much more complex. In this content model we focus on enabling semantics and we argue that issues related to ontology integration and merging are a separate research topic, and are therefore not in the scope of our research objectives.

We also allow the use of selectors, one of the core RSL concepts. `Selector` subtypes can be used to refer to smaller parts of resources. For instance, selectors can be used if only a specific region of an image should be included in a presentation, or for creating a navigational link that points to a specific time interval in a movie.

5.6 Model Functionality and Discussion

In this section we show how the abstract concepts defined above help us to meet the model requirements (MR1 - MR6) presented in Section 5.2.1. We iterate over these requirements and explain how they are addressed concretely.

MR1: Extensible and Flexible Core Components The content model defines generic reusable components that can be extended to match the requirements of many different presentation styles. The model can represent classic presentations by using slide containers, but other types of containers can be created for a zoomable canvas presentation, a presentation where content is aggregated into book-like chapters, or any other imaginable structure. In the same way, layouts, content types and semantics can be filled in as required. These customised and extended presentation models are easily translated into a working application through our model-driven link service for application-specific RSL models. Figure 5.9 shows an example instance of a model that represents one of the many MindXpres [228] presentation styles. The parts of the model annotated with numbers are similarly highlighted in the screenshot of the

resulting presentation on the right. In this case the presentation uses a hybrid visualisation approach, where a zoomable canvas is combined with classic slides. Multiple levels of content aggregation and layouts are used: "sections" are laid out in rows and each section contains a grid of content which can optionally be grouped in slides. A selector is used to include only a part of the image in the presentation. For clarity, concepts such as transclusion, users, context and navigation are not shown.

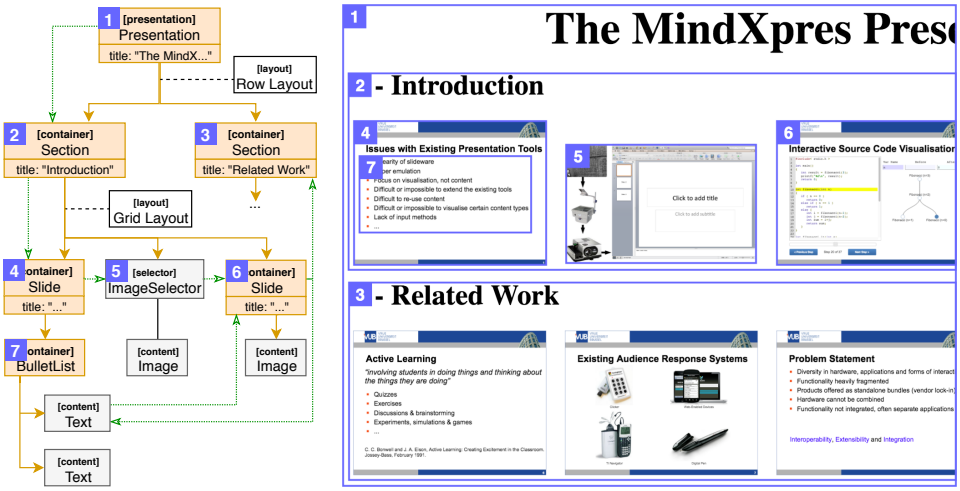


Figure 5.9: Model instance and annotated screenshot of the resulting hybrid canvas and slide-based zoomable MindXpres presentation

MR2: Fundamental Support for Reuse and Collaboration In contrast to some related work, reuse and collaboration is not added as an afterthought and all related concepts are present in the core of the content model. Content is stored at a fine granularity and can be transcluded across presentations, even across content repositories. We extended RSL to support versioning and RSL's user-related components provide the concepts needed to deal with other issues such as user access management and ownership. These features apply to all entities (including links), which is not always the case in other systems. Support for the envisioned reuse and collaboration was also been integrated in the model. Figure 5.10 illustrates how transclusion via structural links (dashed orange lines) enables flexible reuse of content and structures. Presentation A reuses a slide and its content from another presentation in the same content repository. Presentation A also reuses an image from an external repository, but an image selector is used to include only a part of the image. A user might even apply a colour filter to the extracted image region, without requiring duplication and without affecting the original image. Presentation A further includes another slide from an external repository. Multiple users can operate on the same content repository, which is not necessarily located on their own computer, allowing collaboration.

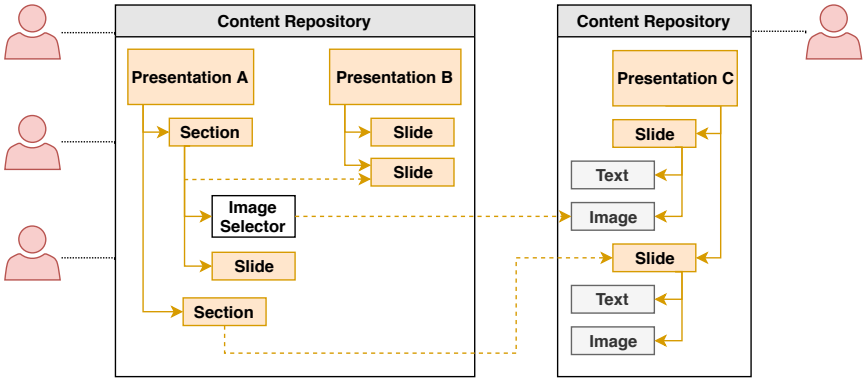


Figure 5.10: Collaboration and reuse of content across content repositories

MR3: Extensible Semantics and Semantic Interoperability We provide semantic associations as well as semantic tagging to allow any entity, or relationship between entities to be semantically annotated. To avoid shortcomings of other document models, we do not enforce a specific ontology. Instead, semantics can be extended to either match existing ontologies or to implement a custom vocabulary. Also note that semantics can be applied on a small granularity (e.g. individual content fragments or associations), avoiding some limitations of existing content models [87]. Figure 5.11 shows an example where various parts of a presentation structure are semantically annotated. For the sake of the example, a mixed vocabulary from different existing ontologies is used. Parts of the presentation can also be directly associated for semantic purposes, for instance via a `isRelated` association, a custom typed extension based on the `Semantic Link`. The flexible semantic annotations enable functionality related to semantic queries or content recommendations for presentations, and the flexible vocabulary makes the presentation content interoperable with existing systems. Although optional, namespaces can be used in semantic tags to avoid collisions between ontologies.

MR4: User Roles and Access Management RSL's user components (shown in Figure 5.3) allowed us to easily fulfil this requirement and they are also provided in the implementation to further support other requirements such as R2. This functionality is for instance used to keep track of creators, authors and owners of content (and their versions) and also allows users to decide who can view, edit or copy their content. This satisfies currently unfulfilled user requirements identified by studies such as [188]. The context adaptation described in R6 can make use of this information to adapt presentations to specific people or groups.

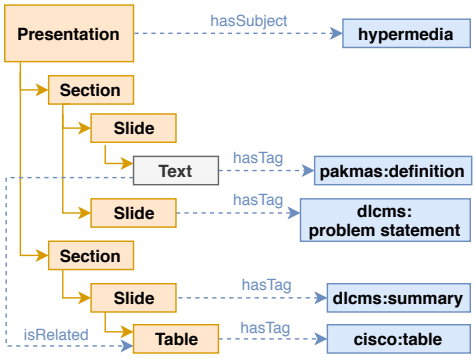
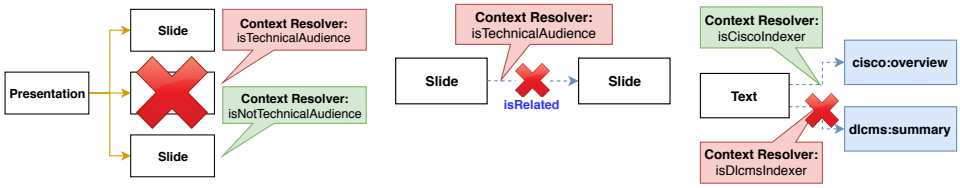


Figure 5.11: Semantically annotated presentation content

MR5: Complex Navigational Structures The model's `Navigational Link`, based on the generic but powerful RSL `Link` makes it possible to represent the complex navigational meshes required by some tools. The concept of selectors also allows arbitrary small subparts of any resource (not just text) to act as sources or targets of links ("anchor points"). Figure 5.9 illustrates some example navigational links in green. In this case, the navigation starts at the root of the presentation, which in the case of a ZUI might show the entire canvas from a distance. We then navigate to a particular section of the presentation, and then move to a specific slide. From the slide, we navigate to a (partial) image and then to another slide. From there, two different paths are available for navigation. Although the example is relatively basic, it illustrates how navigational links can form navigational structures in presentations with multiple levels of content aggregation. We also show how navigational links with multiple targets can be used for more complex navigational structures. How these structures are visualised and triggered at runtime is application specific and up to the developers.

MR6: Context-aware and Adaptive Presentations Last but not least, RSL's context resolvers allow us to control access to any entity based on the current context. Basic support for context resolvers is also provided in the implementation allowing a presentation to adapt content, structure, navigation or semantics. This allows presentations to go beyond related work such as NextSlidePlease [259] which only adapts to time restrictions. Figure 5.12 shows three examples of how context resolvers can be used to adapt a presentation to context. In Figure 5.12a the document structure is adapted based on the background knowledge of the audience. Assuming a non-specialist audience for this scenario, the second slide is not part of the presentation while the third one is. Note that context resolvers can also be placed directly on links, and a context resolver on a structural link might also affect the document structure. Similarly, navigational or semantic links can be disabled in certain contexts. This is

demonstrated in Figure 5.12b where two slides are only considered related (a semantic association) in the case of a specialist audience. Figure 5.12c shows a scenario where the semantics of a piece of context can change. In this scenario the resource is tagged with similar semantics, but from different ontologies. Depending on the capabilities of the content indexer (e.g. the ontology that it supports) different semantic tags are given.



(a) Adaptive document structure (b) Adaptive associations (c) Adaptive semantics

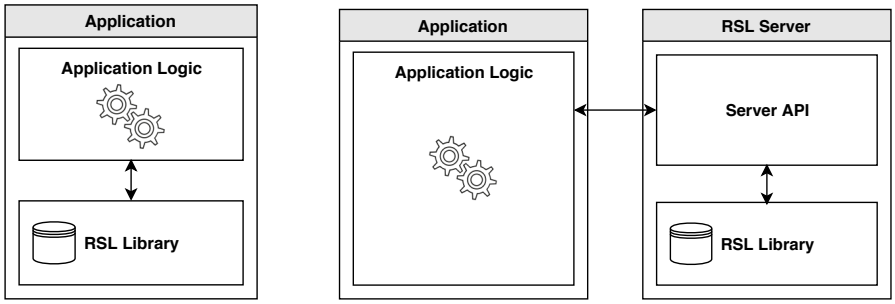
Figure 5.12: Adapting presentation content, structure, navigation or semantics to context

5.7 Implementation

We have developed a working information system based on our new content model for two main reasons. First of all, it demonstrates the viability of the concepts we described in practice. Second, it will facilitate the technical implementation and evaluation of solutions in higher layers of the presentation stack that depend on functionality offered by the model. There have been implementations of RSL-based linking servers before (e.g. iServer [249]) but due to issues related to unmaintained dependencies, performance, extensibility and usability we saw the need for a new implementation. As RSL is a common denominator in our lab's research we saw this as an opportunity to create an improved reusable RSL software framework. By improving the usability we also make it easier for our students as well as external researchers to quickly get started with RSL. For this reason one of the main priorities of the new implementation was to minimise the complexity of turning a conceptual hypermedia solution into a working application. In order to avoid shortcomings of existing implementations we also focus on implementing the framework as a set of decoupled modules so that components of the framework can be replaced or extended without refactoring the entire codebase.

As a first design choice we decided that our implementation should be usable in two ways. The first option would be that applications can directly make use of hypermedia functionality by embedding our framework as a software library. The benefit of this approach is performance as there is no need for intermediary exchange protocols and formats. A limitation is that this requires the application to be created in the same

programming language as the library, otherwise bindings need to be created. As a second option we provide the possibility to run the software as an external linking service. Although the same core RSL library is used, it is wrapped in a server component that for instance exposes functionality via a RESTful or WebSocket interface. Benefits include better compatibility with application platforms and technologies. This approach also allows devices, applications and users to operate on the same content repository. A limitation is performance as the server interface and transport channels introduce significant overhead and performance penalties. Both options are illustrated in Figure 5.13.



(a) Embedded as software library (b) As an external service

Figure 5.13: Two ways of using our generic RSL implementation

Our model is highly extensible and offers a lot of freedom on the conceptual level, but it would be inconvenient if large parts of the implementation would have to be refactored for every application-specific presentation model. To address this issue, we designed the implementation as a reusable model-driven information system. This means that users should be able to load their RSL-based model into the system and all functionality is automatically provided without having to write any code related to hypermedia concepts or persistence. As a result, users do not have to understand how the system is implemented to use it, lowering the barrier for adoption. Currently, models are declared in a JSON format as shown in Listing 5.1. The format is still undergoing frequent changes so the format has not yet been formalised. This is also not in the research scope of this dissertation. However, a formal definition of the format in JSON Schema is planned for the near future. A model starts by declaring a name and a version. The version is used to keep track of the chronology of model updates. If a newer version of an existing model is detected, the system is triggered to update various internal components and to migrate existing data in storage to match the new model. The rest of the model format specifies domain-specific extensions of the core RSL `Resource`, `Selector` and `Link` classes. It is also possible to further extend an existing subtype instead of the core RSL classes. Subtypes have a name and op-

tionally a set of key/value pairs with typed values. A link definition also provides the option to set restrictions for its sources and targets. For these restrictions a whitelist and blacklist mechanism is used to limit the object types that can be connected by that particular link.

```
{
  "name": "MindXpres",
  "version": "0.0.1",

  "resources": [
    {
      "name": "Presentation",
      "properties": [
        {"name": "title", "type": "string"},
        {"name": "theme", "type": "string"}
      ]
    },
    ...
  ],

  "links": [
    {
      "name": "NavigationalLink",
      "sourceRestrictions": {"type": "whitelist", "list": ["Slide"]},
      "targetRestrictions": {"type": "whitelist", "list": ["Slide"]},
      "properties": [
        {"name": "followCount", "type": "int64"}
      ]
    },
    ...
  ]
}
```

Listing 5.1: JSON format for defining an RSL model

To explain how a model is turned into a working information system we present the architecture of the core RSL library, without the server wrapper, in Figure 5.14. We considered the use of RDF and related technologies but we decided against it because RDF is very low-level. Any of the higher-level features of the RSL metamodel would have to be translated into huge quantities of operations on RDF triples, which would result in bad performance. Instead, we opted to implement the desired functionality directly in more efficient ways. Our implementation was made in Java 8 but the architecture is language independent. The core library defines base classes for the core RSL model, including for instance `Resource`, `Selector`, `Link` and `User` classes. These classes implement the functionality described by the core RSL model. For instance, all entities have methods for managing associated properties and user access, and for following outgoing or incoming links. Similarly, links have methods for managing their sources and targets. Also on the technical level links are first-class citizens and object instances of their own. When a source is linked to a target, a new link instance is created and the source entity is associated with the link instance, and the link instance is associated with the target entity. Internally a link maintains a list of object references to sources and targets, but references to the link are also kept in the source and target objects for efficient bidirectional link traversal. Note that we make use of the facilities provided by the Java Persistence API (JPA) to synchronise these bidirectional entity associations automatically and efficiently.

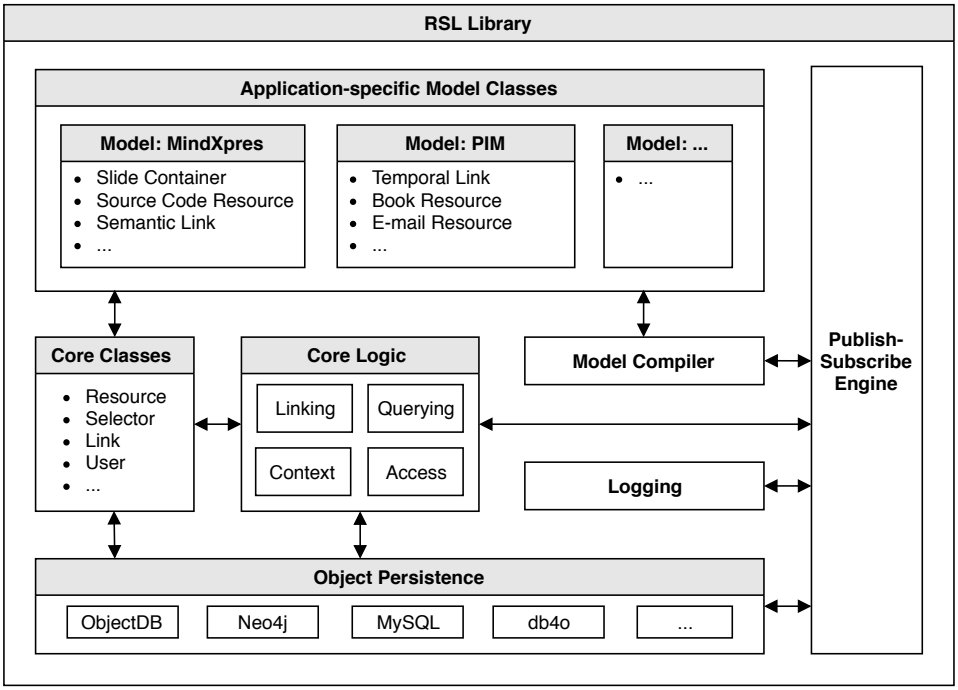


Figure 5.14: Overview of the core RSL framework

These core classes allow graphs of linked data to be created in memory but of course they also need to be persisted in permanent storage. The debate of which database type is more performant for our use case is still ongoing. However, an exploratory comparison was performed by one of our students (see Appendix A). It is clear that a relational database has the worst performance because of all the table joins that are required to follow links. A graph database seems ideal, but current implementations such as Neo4j do not treat links as first-class citizens and do, for instance, not allow links to links. As a workaround one can store RSL links as nodes (not edges) in the graph database but this partially counters the performance benefits we would get from using a graph database. Another possibility is an object database which would allow us to store object instances directly in the database. If an object contains references to other object instances these can be followed efficiently much like in a graph database. In summary, further research is required to determine the optimal persistence approach for our use case but this is not in the research scope of this dissertation. With this in mind, we created an abstract persistence layer so that the persistence engine is decoupled and easily replaceable for future experiments. An initial investigation of the different kinds of storage backends and their performance was executed by Jonas Vereyken as part of his Master's thesis "Investigating Graph-

based Storage Backends for Link Services" [287]. Based on these preliminary results we implemented one storage backend that is compatible with our abstract persistence layer, based on the ObjectDB[1] object database.

When a model in the JSON format is loaded into our system, it is parsed and converted to an internal representation. Then, for each of the subtypes defined in the model, code for a matching Java class is generated using JavaPoet[2]. For instance, if the model defines a `TextResource` as an extension of `Resource`, a Java class called TextResource is generated. Also on the technical level, the generated Java class inherits functionality from one of the predefined `Resource`, `Selector` or `Link` core classes. Furthermore, properties defined in the model are created as fields of the generated class, and various methods such as getters, setters, constructors and utility functions are added. The generated code is then dynamically compiled at runtime and (re)loaded. By using generics these new classes that are derived from the model are immediately compatible with other high-level functionality of the library, such as linking utilities or persistence. In the typical use case, the generated classes are compiled and loaded directly into memory but there is also an option to write a model's compiled classes to a JAR file in the filesystem. This allows users to load the generated jar as project dependency in the IDE so that the generated classes are available at compile time, for instance allowing them to be further extended. However, this should be a rare use case as generic functions are provided to interact with these generated classes at runtime, using reflection. We would also like to point out that more than one model can be loaded and instances from different models can live side by side in the repository. Instances from different models can also be linked, and since all entities are based on the same superclasses, applications can process objects from an unknown model up to a certain degree. This approach of compiling a JSON model to running Java code was deliberate with the main reason being performance. By doing this we avoid multiple layers of further abstractions, metamodels and reflection, allowing the objects in memory to map directly to the object database. We also make heavy use of ObjectDB's JPA interface to enhance performance, for instance by using lazy fetching or cascading persists.

Next, we discuss the remaining components of the architecture (Figure 5.14) that have not been discussed yet. Some higher-level functionality is provided in the core to deal with the graph of linked data, for which the types might not be known at compile time (because they are defined in the JSON file). This includes the creation, retrieval or deletion of entities based on their ID, model or name. A basic crawler is also provided to help users extract the part of the potentially complex graph that is relevant to them. This is done by providing the crawler with a list of entity types and link types

[1] `https://www.objectdb.com`
[2] `https://github.com/square/javapoet`

it is allowed to follow, in combination with a starting object. The crawler then keeps on following unvisited links from the whitelist, aggregating all encountered objects to return to the user. This allows users to quickly get the results for queries such as "get all the resources related to this presentation". At the core of the RSL library there is also a publish-subscribe engine that ties the various components together and orchestrates the flow of events. The engine is based on Google Guava's EventBus[1] and is used for a variety of reasons. For instance, whenever objects are created, deleted or persisted an appropriate event is published to the eventbus so that other components can react if needed. The logging component of the system can also use these events to create detailed logs if desired.

The RSL library previously shown in Figure 5.14 is built as a software library that can be embedded in applications. For cases where a client-server architecture is desired we provide a server wrapper. The server wrapper uses the same RSL software library but adds some additional server functionality on top. An overview of the server wrapper is provided in Figure 5.15. The RSL library, shown with dotted lines, is the same library as discussed earlier but to save space we left out its subcomponents that were already discussed. The server supports many different interfaces in an extensible manner. Some applications might require a RESTful interface while others will prefer a WebSocket for performance reasons. The system defines an abstract interface that "interface drivers" need to implement to be compatible. When the server receives a request, it is translated into a generic internal representation that will be processed by the `Request Handler`. The job of an interface driver is to listen for requests via a specific protocol or channel, and then it should translate requests to the generic representation used internally. Similarly, when the system handled the request, a response in a generic internal format is returned which will be translated back by the interface driver, before it is sent back to the client. This way, all technical details related to a specific interface are isolated in the interface driver and it is easy to add new interfaces if required.

The system uses reflection to map requests to methods of the RSL library. However, the RSL library provides mostly low-level hypermedia functionality and to reduce the amount of requests that need to be made we introduced `Micro Services`. Higher level application-specific logic can be added on the server side as a microservice. For instance, in the case of presentations it would be useful to have a method that returns all content for a given presentation, instead of having to make hundreds of individual requests. Also here reflection is used to map requests to microservice methods. In the case that developers do not use the server wrapper they are of course able to implement these higher level abstractions directly as part of their own application logic. The server also offers methods for storing files. For instance, if one of the

[1] `https://github.com/google/guava/wiki/EventBusExplained`

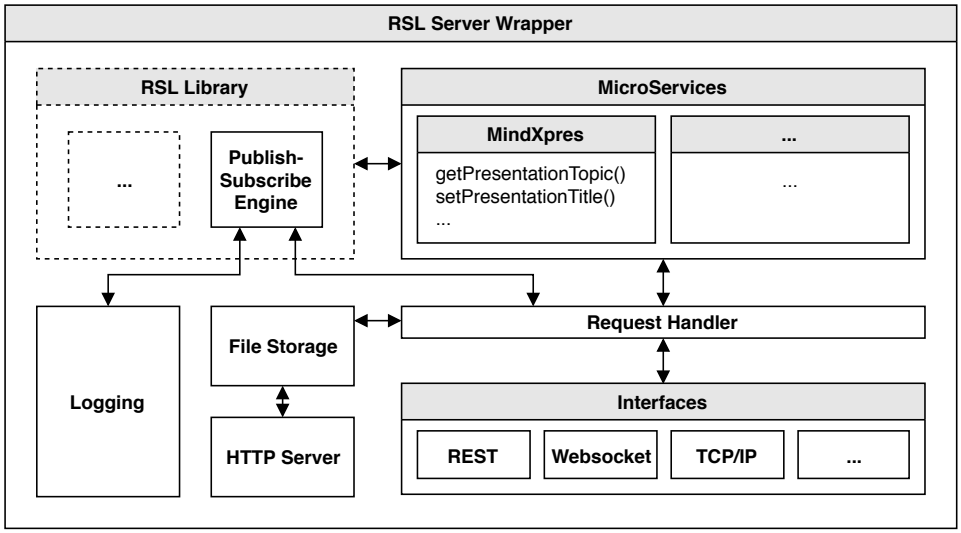


Figure 5.15: Overview of the RSL server wrapper

models defines an Image Resource, the image data itself should not be stored in the matching Java class but instead it should store a pointer to a file in the file storage. When a file is requested, a URL is returned that allows access to the file via the built-in HTTP server. Just like the RSL library, the server also makes heavy use of the library's publish-subscribe engine. For instance, events are emitted when a client connects or disconnects, or when a request or response is made. The server-specific logger can use these events to create detailed logs. However, in the case of the server wrapper the eventbus is also used to efficiently orchestrate the communication between various components. As an example, when a request is received by an interface driver it is published to the eventbus where the request handler will pick it up. When a response is created, it it published back to the eventbus where it will be picked up by the interface driver to send it back to the client. The benefit of our event-based approach is that requests can be handled in an asynchronous and parallel manner, allowing the server to handle large amounts of requests simultaneously.

We end this section by discussing some limitations of the implementation. First of all, currently model migration is not completely implemented and only supports adding and removing complete entities between model versions. In the future also the addition or removal of entity properties or link restrictions should be considered so that models can evolve properly over time. It is worth noting that ObjectDB even provides a mechanism for schema evolution[1]. Next, context resolving is also implemented in a basic way. In the RSL library a context resolver is essentially a

[1] `https://www.objectdb.com/java/jpa/entity/schema`

callback to a user-provided method, delegating the context resolution to user-provided application logic which should determine if the object is accessible or not in a given context. In the server wrapper, context resolvers can be mapped to methods provided in the microservices. In the future a more complex system should be investigated. For instance, the AHAM hypermedia model [65] defines a custom syntax for defining rules. This would allow some context resolution rules to be provided directly in our JSON models. As mentioned before, further research should also be done to optimise the persistence layer. Other that the choice between database types such as object databases and graph databases, certain RSL features can be implemented in radically different ways with different performance profiles. Most database systems also offer complex proprietary mechanisms for caching, crawling and indexing which can further improve performance. Finally, versioning is implemented exactly as it was discussed in the content model, by chaining different versions of entities. Specific media types such as text can be optimised, for instance by applying delta compression and by only storing the differences between versions. As discussed before we also prioritised a solid foundation so that it can serve as a platform for other researchers and that others can easily make improvements. For instance, Gradle is used to manage the project and its dependencies so that it can be built and used on any operating system with little setup, and hundreds of unit tests were created to cover most of the code. Given the scale of this implementation it would take up too much space to provide class diagrams and matching explanations, but a more compact view of the project's code structure is provided in Appendix D.

Chapter 6

The MindXpres Presentation Engine

Our new approach for managing and storing presentation content allows us to provide new solutions that address some of the current shortcomings. However, even after providing a new information system for presentation content, we cannot easily modify existing software to make use of the concepts provided by our content model. For instance, PowerPoint does offer a plug-in mechanism, but the API is based around the typical linear sequence of slides. It is difficult, if not impossible, to implement a zoomable user interface or other radically different presentation structures on top of PowerPoint. We have presented some academic related work, but these tools are often isolated solutions that focus on a specific problem with no way to extend them or apply them in a broader context. This problem is often described, and the need for improved modularity and interoperability in learning technology is motivated in existing literature [43]. In order to experiment with new approaches and to provide a complete package of well-integrated solutions, we deemed it necessary to create our own presentation platform with a new visualisation engine. As detailed later, the main goal of the new presentation engine is to be generic and extensible so that we, as well as other researchers working in the same domain, can more easily prototype and evaluate radically new presentation features instead of having to start from scratch. Note that this chapter focusses on the architecture and components or the MindXpres presentation engine. Example plug-ins and solutions that build on this platform will be detailed later in Chapters 7, 8, 9 and 10.

Parts of this chapter are based on work that we have published at the BCS Conference on Human Computer Interaction (HCI 2013) [227], the International Conference on

Web Information System Engineering (WISE 2014) [228] and the IEEE International Conference on Advanced Learning Technologies (ICALT 2014) [230].

6.1 Technical Platform Requirements

We start by describing some technical requirements, which also help to clarify the goals of the platform. Note that these requirements only apply to the runtime engine of MindXpres, and requirements for the content layer (e.g. reuse or collaboration) and authoring tools are presented in other chapters.

Platform Requirement 1: Extensibility

The platform is created to address the lack of extensibility of existing tools so to avoid similar issues it is important that our platform can be extended for any kind of runtime functionality. For instance, the platform should support *different types of presentation styles* such as classic slide sequences or zoomable user interfaces and offer the possibility to add new ones. Similarly, the media types that are supported, and how they are visualised, should not be fixed and might be highly *dynamic* and *interactive*. Also the way users interact with the content (e.g. for navigation) should be flexible and adaptable by developers to meet their requirements. This requirements falls in line with the documented need for modularity in learning technology [43].

Platform Requirement 2: Connectivity

It is also important that the runtime engine is able to *communicate* with other instances of the same presentation or with other applications. Related work shows the importance of involving the audience in presentations, and this requirement will form the backbone for *audience response systems*, allowing audience members to participate in integrated presentation activities via clickers or other mobile devices. The same connectivity infrastructure can be used to connect *mobile presenter views* or other special presentation hardware. Also functionality such as synchronising the navigational state of the presentation across devices or asking questions digitally can be enabled if presentation instances can communicate. The need for these features are motivated in more detail in Chapter 8 when we describe related solutions built on top of the MindXpres platform. The MindXpres content model described in previous chapters already provides interoperability on the content level, but this requirement provides further runtime interoperability with other applications. This is again in line with the need for improved interoperability in learning technology [43].

Platform Requirement 3: Portability

In order to not exclude some interesting new functionality we aim to minimise the software and hardware requirements for running, or participating in, a MindXpres

presentation. We want to avoid situations where it is necessary to install large, sometimes expensive, software suites and where only a limited set of devices are supported. It should also be possible to run MindXpres presentations on mobile devices such as tablets or smartphones. This is further supported by a study from Thielsch et al. [270], who found that users chose cross-platform support for presentation tools as the third most important technical property, with a mean score of 4.2 out of 5. The same study also mentions the need for the improved compatibility of presentation files, which further related to the portability of presentations.

Platform Requirement 4: Support for the Post-Presentation Phase

A presentation is given the most attention at the time when it is given, but already in existing tools we see that presentations also have an "afterlife". For instance, they might be used as study material by students or anecdotal evidence suggests that some companies archive important presentations as a formal report of the related event. This claim is further strengthened by the fact that Slideshare, one of the many presentation sharing platforms, has over 80 million users and hosts over 18 million presentations[1]. RIMES [147] is a related tool for embedding interactive multimedia exercises in lecture videos, a common learning material for online learning. However, RIMES only focusses on lecture videos and as an extension of Office Mix, the exercises are not a part of the presentation content. Furthermore, our future vision includes non-linear presentations with interactive content and as briefly described in Platform Requirement 2, also social functionality. We state that special care should be taken to also make use of these benefits after the actual presentation. Related to Platform Requirement 3, it should be easy to share the presentation with others and all the interactive and social components should remain intact, also after the presentation session. In the case of a non-linear presentation, it should be possible to freely explore the presentation but also to play back the navigational path that was followed by the presenter during the actual presentation. Even at this stage, some of the social features might still be enabled, for instance allowing viewers to discuss content.

6.2 Architecture

The general idea behind the MindXpres platform is illustrated in Figure 6.1. To summarise the figure, a presentation is described in a MindXpres document which is then compiled into a portable and interactive presentation. As will be shown later, a MindXpres document can be an XML-based document format or it might be a presentation structure stored in the information system presented in Chapter 5. In the following sections we describe this process in more detail.

[1] `https://www.slideshare.net/about`

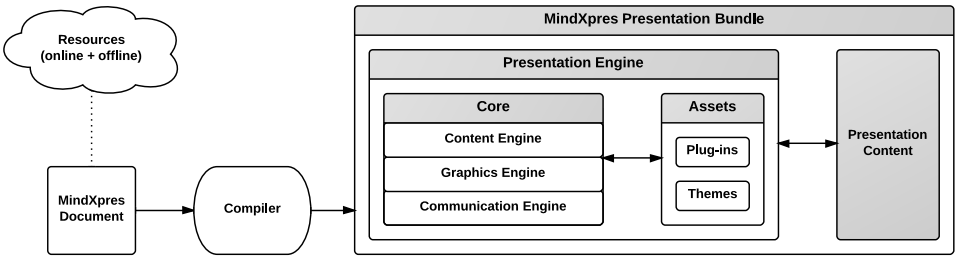


Figure 6.1: General architecture of the MindXpres platform

6.2.1 Compiling MindXpres Documents into Presentations

The system expects a presentation to be provided in the form of a MindXpres document. The format of storage behind such a document can be implemented in different ways. As discussed in more detail later, our initial implementation required presentations to be described as a single monolithic XML-based file. However, in later versions a MindXpres document is represented and stored as a presentation structure stored in the information system for presentations detailed in Chapter 5. MindXpres documents are then passed to a compiler which transforms a MindXpres document into a functional portable presentation. Initially simple transformations (e.g. based on XSLT[1]) were considered to turn the document into a presentation, but it quickly became clear that a more complex compilation process is required. First of all, a dedicated compiler would make it easier to support multiple output formats. For instance, for dynamic and interactive presentations the resulting presentation might be based on web technologies, but for printing (e.g. handouts) the compiler might generate a PDF document. Second, given our focus on supporting true content reuse and transclusion content might be distributed across different locations. For performance reasons, or for cases where there will be no stable Internet connection during the presentation session, the compiler can already aggregate all referenced content and create a self-contained offline version of the presentation if desired. Finally, the compiler can be used to pre-process referenced content. As an example, if the presentation includes a video in a format that is not supported by the output medium, the compiler can still convert the video at compile time. This way, the lengthy conversion does not need to occur at runtime, if runtime conversion is at all possible in the requested output medium. Note that the compiler is also able to extract information from proprietary document formats such as spreadsheets. Another example is that one of our plug-ins for visualising source code runs the code through a language-appropriate debugger at compile time to obtain metadata about the provided code's execution flow [226, 225].

[1] `https://www.w3.org/TR/xslt/all/`

6.2.2 MindXpres Runtime Engine

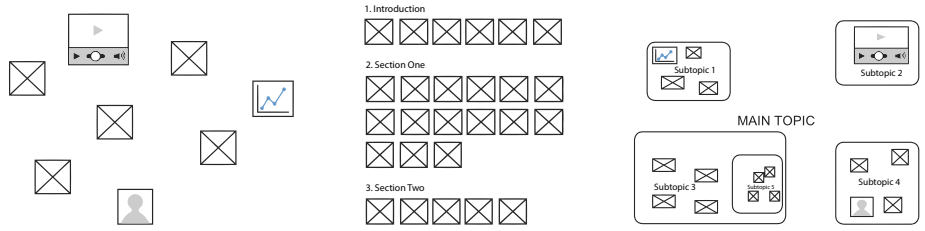
At the end of the compilation process the relevant content is bundled together with the MindXpres runtime engine to form a portable cross-platform self-con- tained presentation bundle. The MindXpres runtime engine is responsible for visualising the content in the bundle and to make it dynamic and interactive. In order to be truly extensible on every level, almost all functionality is implemented as a plug-in so that features can easily be replaced or added. However, the runtime engine offers support for fundamental functionality commonly required by plug-ins. The *content engine* is responsible for processing the content and linking it to the corresponding visualisation plug-ins. The *graphics engine* abstracts all rendering-related functionality. For instance, it offers high-level abstractions for zoomable user interfaces, animations or transition effects. The *communication engine* exposes a communication API that can be used by plug-ins. It provides some basic functionality for fetching external content but also offers the possibility to form networks between multiple MindXpres presentation instances as well as to connect to third-party hardware such as digital pens or clicker systems. The communication engine also allows event-based information exchange between plug-ins for plug-in interoperability. Next to these core modules, the compiler also adds all plug-ins and themes required by the presentation to the resulting presentation bundle. Themes may contain visual styling on a global as well as on a plug-in level. When the content engine encounters different content types, they are handed over to the specific plug-in which uses the graphics engine to visualise the content.

In order to provide the necessary flexibility, all non-core modules are implemented as plug-ins. These plug-ins determine how content is visualised, even for basic content types such as text and images. Plug-ins also add higher level functionality. For instance, they might determine the structure and visualisation of the presentation (e.g. a ZUI, a grid of slides or a classic slide sequence) or they might add functionality for the presenter in the form of a presenter view. Plug-ins are categorised in four major groups:

- *Content Plug-ins* form the basic building blocks of a presentation. They are represented by plug-ins that handle the visualisation for specific content types such as text, images, bullet lists, graphs or videos. The content engine invokes the corresponding plug-ins in order to visualise the content. Plug-ins may further use functionality provided by the content, visualisation or communication engine to provide rich functionality.

- *Container Plug-ins* are responsible for aggregating and organising content in higher level structures. As an example, an image gallery container plug-in might group images together and visualise them accordingly, for instance in a grid or a

scrollable list. A bullet list is another example of a container plug-in, as it groups together fragments of text and visualises them as bullets. Containers can be nested to create multi-level structures, similar to how containers are used in our content model. For instance, Figure 6.2 shows some higher-level presentation structures that are implemented as container plug-ins. These structures might contain slides, which are also containers, and these slides might in turn contain other containers such as bullet lists. Although aspects such as transition effects or navigational paths can always be overridden by users, containers should offer suitable defaults. For instance, in the case of the *Structured Container* shown in Figure 6.2b, a prezi-style pan and zoom transition effect is enabled and by default a navigational path is created to traverse the subcontainers in their logical order.



(a) Graph layout (b) Structured layout (c) Clustered layout

Figure 6.2: MindXpres plug-ins can affect a presentation's macro structure, providing options such as a graph, structured or clustered layout

- *Core Plug-ins* are used to extend the core functionality of the platform; functionality that goes beyond providing an appropriate visualisation for specific content. For example, a plug-in that records the navigational path taken by the presenter for later playback falls under this category. Another example is a plug-in that determines contextual clues from the physical world so that other plug-ins may use this information for context adaptation.

- *UI Plug-ins* offer developers to plug in additional user interface components. For instance, these plug-ins are used to for instance show a "minimap" of the canvas for ZUI presentations, or in the case of a presentation feedback system a UI plug-in might show discrete popups to notify the presenter of some issues. Note that these UI elements can be provided only for certain users, also allowing them to be used for custom presenter overviews and controls.

6.3 Implementation

In order to help fulfil the requirements of portability much of the platform is implemented using web technologies such as HTML5, CSS3, JavaScript. As a result only a modern web browser is required to run a MindXpres presentation bundle. This ensures that presentations can be run on different operating systems without the need for additional software. It also allows mobile devices such as smartphones and tablets to run the presentation or to participate in it, for instance for audience-driven activities or as dedicated presenter view. Additionally, developers can make use of the many existing JavaScript libraries to visualise content.

6.3.1 Document Format and Authoring Language

MindXpres presentations are represented in XML and a simplified example is provided in Listing 6.1. The XML can be generated by a graphical authoring tool but special care was taken to keep it human readable. This way, the format can also be used as a declarative authoring language, similar to how LaTeX is used for text documents. The vocabulary of the format is entirely determined by the plug-ins. Every plug-in exports a set of tags (potentially with attributes) that can be used in the XML format, and when the tag is encountered by the runtime engine the corresponding plug-in is invoked to handle it. As an example, a source code visualisation plug-in might export the `<code>` tag, and when it is used in the format the runtime engine will invoke the plug-in when the presentation is started to visualise the source code provided between the tags. The format allows content to be embedded directly in the XML like in the example in Listing 6.1, but one can also provide an external source for every tag, for instance referring to content or structures stored in the information system presented earlier. The result of the example XML is shown later in Listing 6.2.

```
1 <presentation>
2   <slide title="Vannevar Bush">
3     <bulletlist>
4       <item>March 11, 1890 - June 28, 1974</item>
5       <item>Amercian Engineer , founder of Raytheon</item>
6     </bulletlist>
7     <image source="bush.jpg"/>
8   </slide>
9 </presentation>
```

Listing 6.1: An example of a slide definition in the XML-based MindXpres authoring language

6.3.2 Compiler

The compiler is realised as a Node.js[1] application implemented in JavaScript. The Node.js runtime is necessary to facilitate some functionality that could not easily be implemented in purely client-side web technologies. For instance, the compiler needs free access to the filesystem to read files and assemble the presentation bundle. As mentioned before the compiler also needs to be able to perform some heavy-duty tasks such as video conversion or extracting content from proprietary documents. As an added benefit the Node.js ecosystem provides packages for this more complex functionality via the npm[2] package manager and repositories.

When a MindXpres document in the XML format is provided, it is checked for validity. However, since the vocabulary is not fixed and instead defined by the plug-ins, an XML Schema template is taken as starting point and then augmented by the compiler based on the vocabulary and constraints provided by the plug-ins. After validation the document is parsed. Every time a tag is encountered, the corresponding plug-in's preprocessor actions are triggered if provided, allowing for instance a video plug-in to convert an unsupported video format into something usable. The tag is then converted to HTML5 by simply encoding the information in the attributes of a `div` element. The HTML5 standard allows custom attributes if they start with a the "data-" prefix. Listing 6.2 shows parts of the transformed XML document shown earlier in Listing 6.1. Note that the transformation does not include visualisation-specific information but merely results in a valid HTML5 document which is bundled into a self-contained package together with the presentation engine. The compiler also adds all referenced plug-ins and themes to the bundle. Webpack[3] is used to create more compact bundles from the referenced material.

```
1 <div data-type="presentation">
2   <div data-type="slide" data-title="Vannevar Bush">
3     <div data-type="bulletlist">
4     ...
```

Listing 6.2: A snippet of the MindXpres format after compilation to HTML

6.3.3 Runtime Engine

To run the presentation it is sufficient to open the generated HTML document in a modern web browser. Although the document itself only contains structured data, the included runtime engine (together with the bundled plug-ins) will transform the

[1] `https://nodejs.org`
[2] `https://www.npmjs.com`
[3] `https://webpack.js.org`

HTML document into a visually appealing and interactive presentation. We now describe how each of the runtime engine's subcomponents are implemented.

Content Engine

When a presentation is loaded, the content engine is the first component that is activated. It processes the content of the HTML document and whenever a `div` element is discovered, the data-type attribute is read and the corresponding plug-ins are notified in order to visualise the content.

Graphics Engine

The graphics engine provides support for interesting new visualisation and navigation styles. It offers helper functions for animation, efficient panning, scaling and rotation as well as some predefined effects (e.g. blurring or fading). These are all implemented via CSS3 transformations and can for instance be used for zoomable interfaces, transition effects or content visualisations. Our graphics engine pushes the capabilities of CSS3 transformations to their limits and as the graphics engine was developed during the early stages of CSS3 support in web browsers, we identified various bugs and submitted multiple bug reports for Chrome and Firefox that were acknowledged and addressed.

Plug-ins

Plug-ins are implemented as JavaScript bundles which consist of a folder containing JavaScript files and other resources such as CSS files, images or other JavaScript libraries. By convention, a plug-in should also provide a manifest file with a predefined name. The manifest contains metadata such as the plug-in name, version, name of the main source file, and a list of tags that can be used in the document format. The plug-in claims unique ownership for these tags and is in charge for their visualisation if they are encountered by the content engine. A plug-in must implement two methods, optionally three, to be compatible with the platform:

- `init()`: This method is called when the plug-in is first loaded by the presentation engine. The method gives the plug-in the opportunity to perform some initial setup before it is used, for instance to load additional JavaScript or CSS via the provided dependency injection functionality or to set up some data structures.
- `render()`: When the presentation engine processes the HTML document it iterates through the presentation structure and delegates the processing of the DOM nodes (originally XML tags) to the plug-in that provides functionality for

that tag. For instance, if the DOM node is tagged as a slide, the node is passed to the `render()` method of the slide plug-in which will then visualise it as a slide and lay out the content within accordingly. A plug-in is free to modify the DOM tree and may also register callbacks to handle future interaction with the content.

- (optional) `preprocess()`: As mentioned before, plug-ins can provide functionality that will be executed when the XML-based MindXpres format is compiled into a presentation bundle. If the method is provided, it will be invoked when tags claimed by the plug-in are encountered in the XML-based MindXpres format. The plug-in can for instance use this to transform the referenced data at compile time for easier processing at runtime. As the compiler is also based on JavaScript the method can be invoked directly from the compiler, and code in this method can make use of Node.js's powerful server-oriented packages.

Themes

We currently use CSS to provide a basic templating system. Plug-ins should include their own CSS files for a default style, but these themes allow styling to be overridden either on a global or on a plug-in level. For instance, a theme might set the text font for the entire presentation, or it might redefine the font only for a specific plug-in.

Communication Engine

The concrete requirements and details of the communication capabilities of MindXpres have been published as a separate paper [230] and allow MindXpres to be used as a platform for active learning components. We refer to the original paper for a detailed explanation but we provide a brief summary. Active learning ("*involving students in doing things and thinking about the things they are doing*" [37]) has been thoroughly evaluated [114, 123, 209, 215] and in general a higher level of engagement, increases attendance as well as motivation and aids in the learning process [141]. In the context of presentations we see related work in the form of audience response systems, which use clickers or mobile devices to let audience member participate in for instance quizzes and polls. Some of the major manufacturers include Turning Technologies[1] and iClicker[2]. However, existing systems suffer from some shortcomings. For instance, the communication flow is often unidirectional (e.g. from the presenter to the audience) even though the benefits of communication between students themselves is important [137, 75]. Furthermore the various hardware solutions are incompatible, software is proprietary and the activities are limited, sometimes not well integrated in presentation tools and cannot be extended.

[1] `https://www.turningtechnologies.com`
[2] `https://www.iclicker.com`

To address these issues, MindXpres provides the architectural foundation for creating networks between MindXpres presentations, and integrating and unifying third party hardware. This allows MindXpres plug-ins to provide active learning components and to support a wide variety of the existing hardware for these activities. In other words, the communication engine is not only used to retrieve external content at runtime but also allows the exchange of information between MindXpres plug-ins, across different devices and allows plug-ins to make use of external hardware such as clickers or digital pens. This is achieved by providing an access point that orchestrates and mediates the flow of information between the involved devices, shown in Figure 6.3.

Figure 6.3: The MindXpres access point orchestrates and mediates communication between presentations, plug-ins and hardware peripherals

The access point is mainly a software solution and can be run on simple hardware such as a Raspberry Pi or the presenter's laptop. We used a small Intel Next Unit of Computing Kit (NUC) with high-end WiFi and Bluetooth modules to act as a central access point and to provide the underlying network support. MindXpres instances use WebSockets to communicate with other MindXpres instances via the access point. The access point further acts as a container for data adapters which translate input from third-party input and output devices into a generic representation that can be used by the MindXpres instances in the network. For instance, when a button is clicked on a third-party clicker device this is captured by the appropriate data adapter on the access point. The event is then translated into a more generic representation and broadcasted across the network so that MindXpres plug-ins may react accordingly (e.g. updating the amount of votes in the poll). The architecture that enables such scenarios is illustrated in Figure 6.4.

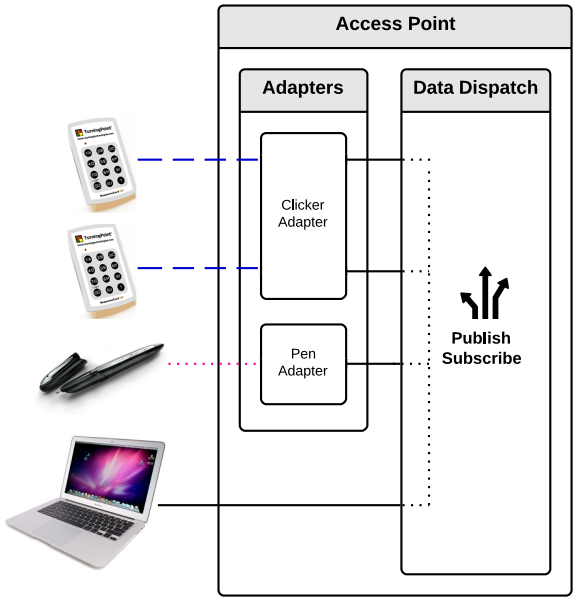


Figure 6.4: The MindXpres access point orchestrates communication via a publish-subscribe mechanism and a set of plug-in data adapters

This approach allows different clickers to be used together, and they might even be combined with other devices, for a wide range of well-integrated activities. Our approach gives "outdated" devices a new life, allowing them to be used longer before they need to be replaced which is a common issue with classroom technology [173]. In order to go beyond simple broadcast-based communication, we have implemented a routing mechanism based on the publish-subscribe pattern where plug-ins can subscribe to specific events or publish information, which is also shown in Figure 6.4. The communication engine provides the basis for audience response systems or even full classroom communication systems where functionality is only limited by the creativity of plug-in developers. More detailed examples of the use of our event-based communication engine are provided as part of Chapter 8, where we describe some novel solutions that make use of the concepts described in this section.

6.4 Example Plug-ins and Presentation

Some advanced plug-in solutions built on top of the MindXpres platform are detailed later in Chapters 7, 8, 9 and 10. However, in order to provide a better understanding we show some trivial plug-in examples. Figure 6.5 shows how the vocabulary of the slide, bullet list and image plug-ins are combined to declaratively describe a single slide in a presentation. The figure also shows the resulting slide. Note that a slide manages the layout of its children automatically but this automatic layout can of course be disabled if desired.

```
<slide layout="\60\40" title="Vannevar Bush">
  <bulletlist>
    <item>About
      <item>March 11, 1890 - June 28, 1974</item>
      <item>American Engineer</item>
      <item>Founder of Raytheon</item>
    </item>
    <item>
     ...
    </item>
  </bulletlist>
  <image source="http://example.com/bush.jpg">
    <crop bounds="10%, 5%, -10%, -20%" />
  </image>
</slide>
```

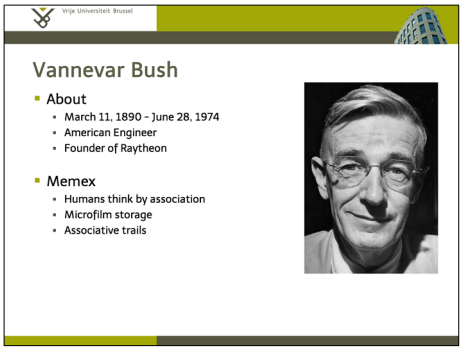


Figure 6.5: The slide container plug-in

Figure 6.6 illustrates the use of an enhanced video plug-in, allowing the presenter to define places in the video where the plug-in should, for instance, pause for a few seconds to highlight and caption some parts.

```
<video source="vid.mp4">
  <caption start="0:00" duration="1500ms">
    Lecture 3 - Butterfly Species
  </caption>
  <pause start="0:43" duration="5s">
    <caption>
      The peacock butterfly (aglais io) ...
    </caption>
    <highlight x="30%" y="9%"
               width="35%" height="40%" />
  </pause>
</video>
```

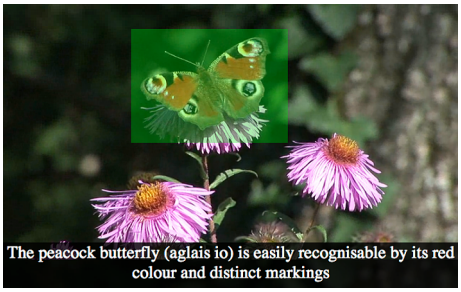


Figure 6.6: The enhanced video content plug-in

Figure 6.7 shows a basic source code plug-in which automatically formats and highlights the code and presents it as a scrollable listing.

```
<code>
 <publications>
  <publication type="inproceedings">
   <title>An Architecture for Open Cross-Media
         Annotation Services</title>
   <author>
    <surname>Signer</surname>
    <forename>Beat</forename>
    ...
   </author>
   <author>
    <surname>Norrie</surname>
    <forename>Moira</forename>
    ...
</code>
```

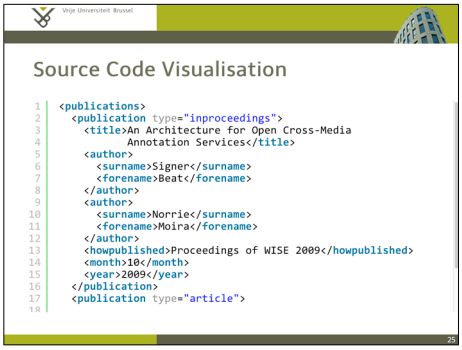


Figure 6.7: The source code content plug-in

Finally, Figure 6.8 shows a screenshot of a complete MindXpres presentation, based on a hybrid ZUI and structured slide approach. MindXpres also supports hyperlinks that can point to internal or external content, which is one of the ways to provide non-linear navigation in presentations [97]. Studies also found that hyperlinks in presentation are desired by users [270].

Figure 6.8: An example MindXpres presentation with the structured layout plug-in

6.5 Discussion

We see MindXpres as a highly extensible prototyping platform that can be used by us, as well as other researchers, to implement novel presentation ideas that would not be possible in existing tools. Almost every component of MindXpres can be extended or replaced, but core functionality is provided for concepts shared across plug-ins (e.g. graphics or communication). This level of extensibility fulfils Platform Require-

ment 1 and the communication engine fulfils Platform Requirement 2. Our choice of using web technologies makes MindXpres highly portable and minimises software dependencies, fulfilling Platform Requirement 3. This choice also helps us to fulfil Platform Requirement 4. As MindXpres presentations are based on HTML, they are essentially self-contained "websites" that can easily be placed online for sharing and reuse. Also the plug-ins remain active in this case, so an uploaded presentation maintains all its interactive and dynamic features. Furthermore, plug-ins can provide functionality specifically for the post-presentation phase, for instance allowing future viewers to play back the navigational path and annotations made by the presenter during the live presentation. The chosen technologies and platform independence also make it easier to integrate MindXpres presentations in other platforms. As an extreme example, MindXpres presentations can easily be embedded in Second Life or Tabletop Simulator[1] as shown in Figure 6.9. MindXpres presentations were initially authored in the XML-based authoring language, but later the XML-based format was used as an intermediary format to compile presentations from our hypermedia-based information system into portable presentation bundles. For the sake of clarity we have only discussed some simple examples in this chapter, but in the following chapters we present further contributions built on top of the platform and show how the MindXpres platform provides the foundation for a wide range of interesting new applications including active learning components, advanced content visualisations, feedback systems and novel presenter views.

Figure 6.9: As an extreme example, platform independence allows MindXpres presentations to be easily embedded in other environments such as Tabletop Simulator

[1] http://berserk-games.com/tabletop-simulator/

Part IV

Addressing Shortcomings and Unmet User Needs

Addressing Shortcomings and Unmet User Needs

In Chapter 3 we have identified various shortcomings and unmet user needs in existing presentation software. We argued that these issues are not easily addressed in the existing tools and their underlying technologies, so in Chapters 4, 5 and 6 we provided the conceptual and technical foundation that enables next generation presentation solutions. In the following chapters, Chapters 7, 8, 9 and 10, we discuss some proof-of-concept solutions to illustrate how the MindXpres platform enables new presentation solutions.

We would further like to make some clarifications that apply to Chapters 7, 8, 9 and 10. First off all, some plug-in solutions in these chapters have been created in collaboration with students, as part of Bachelor's or Master's theses under the supervision of Reinout Roels. If applicable, student contributions will be mentioned at the beginning of every section where a new plug-in solution is presented. A complete list of all student contributions is also provided in Appendix A.

Next, we would like to clarify how our plug-in solutions integrate with the conceptual and technical foundations presented earlier. Our solutions specifically target some of the shortcomings and unmet user needs that we presented earlier. As such, our solutions fall in line with the vision described by our conceptual framework and the solutions were designed to follow the principles of next generation presentations. Furthermore, all of the upcoming solutions are either directly embedded as a plug-in for the MindXpres presentation engine, or the solutions interface with the presentation engine at run-time. Our solutions make heavy use of the high-level functionality for content processing, visualisation and communication that are provided by the presentation engine. However, not all solutions make use of the underlying information system. For some solutions the content layer is not relevant and these solutions build on other parts of the foundation such as the presentation engine. There are also some solutions that were created before we developed the content model and information system, and these solutions are based on the original XML-based MindXpres document format. Regardless, the solutions based on the original document format address issues that are unrelated to how the information is stored. Therefore we do not foresee any problems that would prevent us from connecting these solutions to the information system as part of future work.

Chapter 7

Solutions for Content Visualisation

In Chapter 3 we have shown that one of the major categories of shortcomings and user needs converns the integration and visualisation of certain content types. In this chapter we discuss some plug-in solutions for MindXpres that aim to address the identified shortcomings related to mathemathical formulas, video, source code, interactive data visualisation as well as other interactive components.

7.1 Dynamic Diagram Visualisations

We have previously discussed how animation can help threngthen the viewer's mental model if used correctly. The results of the survey, observations, programmatic analysis and literature study presented in Chapter 3 further confirm that diagrams are used relatively often. However, at the same time our results show that support for this functionality is lacking, especially for creating animated and dynamic diagrams. In this section we present a plug-in solution for dynamic diagrams. The solution was created in collaboration with Kevin van Gyseghem, as part of his Bachelor's thesis "*Dynamic Graph Visualisations for the MindXpres Presentation Tool*" [281].

Tools such as PowerPoint do allow the creation of diagrams and animations can be used to create dynamic diagrams, but the process is tedious and presenters often do not make the effort even though the benefits have been clearly demonstrated in studies [244, 125]. The dynamic diagram visualisation plug-in aims to reduce the effort required to create static as well as animated diagrams to support a presenter's narrative. The plug-in takes a data-driven approach, a recurring theme in the Mind-

Xpres platform, and users should focus on the content and let the plug-in handle the visualisation automatically. To define diagrams and graphs the plug-in extends the MindXpres authoring language with new tags based on the GraphML [41] markup language. However, the markup language was extended to also support changes in the graph over time, a concept not present in the original language. The language supports tags for creating and removing nodes, edges, colours, images and text and by supporting phases one can create distinct states of the visualisation. Listing 7.1 shows an example definition of a graph with different phases.

```
<graph>
  <phase>
    <node id="node1">...</node>
    <node id="node2">...</node>
    <edge source="node1" target="node2">...</edge>
  </phase>
  <phase>
    <node id="node3">...</node>
    <edge source="node2" target="node3">...</edge>
  </phase>
</graph>
```

Listing 7.1: Defining different phases of a graph using GraphML-based tags in the MindXpres authoring language

In between phases, parts of the diagram might be added or removed, or different parts might be highlighted. Once the graph is defined, the presenter just needs to go forward in the navigational path of the presentation to switch to the next predefined phase of the visualisation. For the implementation of the plug-in the Cytoscape.js[1] JavaScript library was used for visualising the graphs and diagrams. The screenshots in Figure 7.1 show the plug-in in action and highlight a few different phases of a single graph.

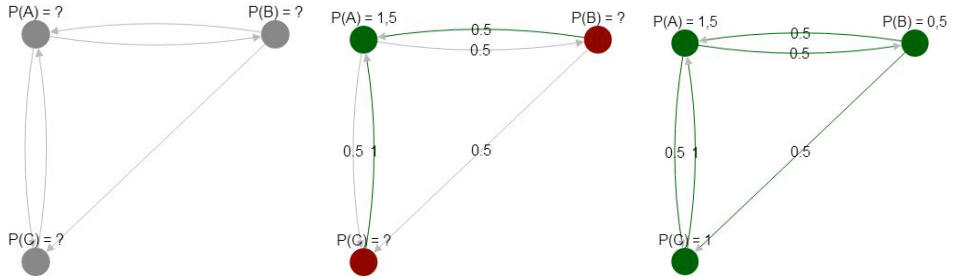


Figure 7.1: A MindXpres plug-in for defining diagrams and graphs with different phases [281]

[1] http://js.cytoscape.org

7.2 Mathematical Expressions via MathML or LaTeX

Our survey, observations, programmatic analysis and literature study presented in Chapter 3 show that the inclusion of mathematical expressions can be a troublesome experience for some users. Based on the indications that came forth out of the general investigation of shortcomings, we searched more specifically and found other literature that supports our findings [104, 245]. For some presenters this is a reason to author their presentations in LaTeX (via the Beamer package) and others create mathematical content externally and integrate it into a PowerPoint presentation as an image. Instead of creating yet another new graphical interface for creating equations, we aim to offer support for the authoring languages these users are likely already familiar with. Our MindXpres plug-in for mathematical expressions allows presenters to use established languages such as the Mathematical Markup Language[1] (MathML) or LaTeX to embed complex mathematical content in a MindXpres presentation. Other than authoring benefits for the presenter, supporting MathML also helps in making this complex content machine readable and it improves accessibility for impaired users [193]. The plug-in was implemented with the help of the MathJax[2] library and Figure 7.2 shows some mathematical expressions rendered by our plug-in.

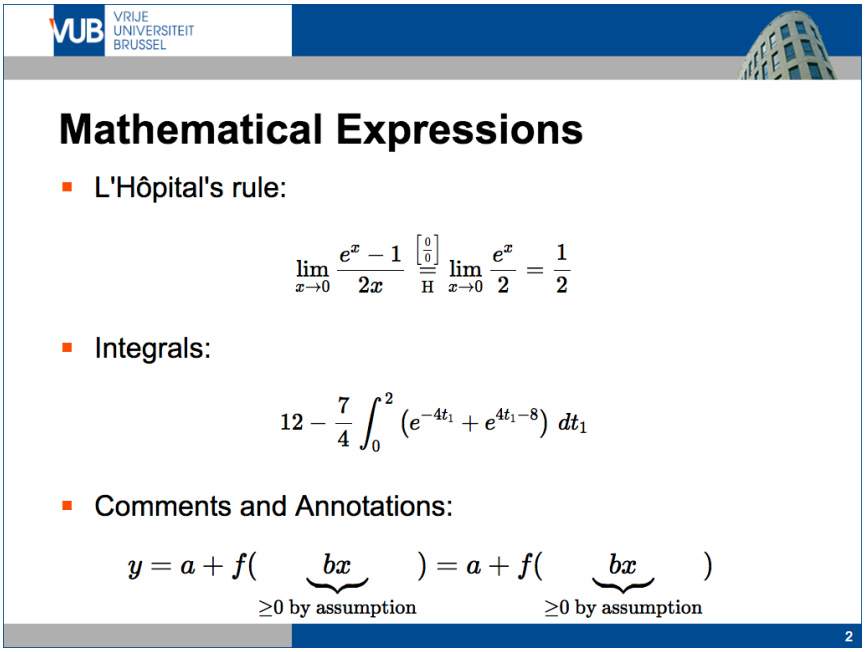


Figure 7.2: A MindXpres plug-in for defining mathemathical expressions via MathML or LaTeX

[1] https://www.w3.org/TR/MathML3/
[2] https://www.mathjax.org

7.3 Advanced Video Player

Tools such as PowerPoint support video playback but the offered functionality is often rather limited. PowerPoint does support bookmarks for embedded videos but nothing else is provided to give the presenter more control over the video during the presentation. Too often we see situations where the presenter attempts to talk during the playback of a video but the audience can only focus on the presenter or the video, but not both. Furthermore, the video often progresses too fast for the presenter to complete their narrative resulting in awkward situations where the presenter fails to synchronise the narrative with the video. To address these issues, we present a new video player plug-in for MindXpres. The plug-in allows the presenter to predefine a playback path through the video and add captions or highlights where needed. For instance, the presenter might specify that the video should pause for 15 seconds at a certain point, and the video can be annotated with shapes, highlights or captions, as illustrated in Figure 7.3. In other words, the presenter can predefine parts of the video that should be skipped, moments when the video should pause, and annotations that should be shown, allowing the playback of the video to be adapted to match the presenter's narrative. The plug-in exports tags that allows presenters to describe the flow of the video in the MindXpres format and an example for the video player has already been shown earlier in Figure 6.6.

Figure 7.3: A MindXpres plug-in for video playback, allowing presenters to predefine skips or pauses, as well as annotations such as shapes, highlights or captions on top of the video

7.4 Source Code Visualisation

The results of our survey and observations show that source code is a particularly difficult content type to present. In programming environments code is often nicely structured and a form of colour coding (syntax highlighting) is used to make the code more readable. However, when the code is copied and pasted into a tool such as PowerPoint, this formatting is lost and the code becomes unreadable as illustrated in Figure 7.4. Presenters are left with the tedious options of manually formatting the content again, to use third party tools or to include less readable screenshots of the code. Another reason why source code is hard to present is that the available space on a slide is very limited, but source code is often composed of many long lines of characters. Since a legible font size is desired, snippets of code have to spread across multiple slides and may have to be word-wrapped in unusual ways to fit horizontally.

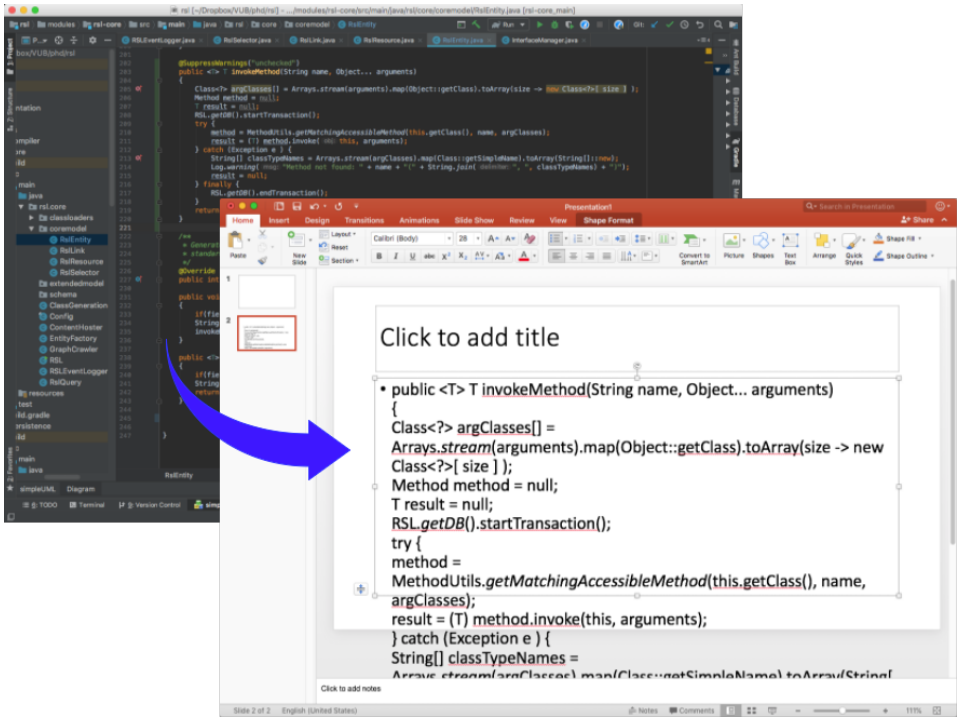


Figure 7.4: Copying and pasting source code often results in the loss of syntax highlighting and formatting

As a first step towards improving the visualisation of source code we present a plug-in for basic source code visualisation that drastically reduces the effort required to present source code in a proper way. In order to use the plug-in a piece of source code

is provided to the plug-in via the MindXpres format and the rest is handled automatically. At compilation time the plug-in first automatically detects the programming language that is used. As every programming language has some unique keywords or symbols, a simple Naive Bayes classifier is used, a common approach with high accuracy [146, 307]. At runtime, the plug-in uses the SyntaxHighlighter[1] JavaScript library to visualise the code with proper formatting and syntax highlighting. In order to avoid spreading code over multiple slides and losing the overview, the plug-in can present the code in a scrollable container; a trivial but currently unsupported solution in existing tools. Figure 7.5 shows the resulting visualisation of a snippet of XML code. At the moment the supported programming languages are limited to the 23 languages supported by SyntaxHighlighter[2] but this includes the most common languages and support for other languages might easily be added in the future.

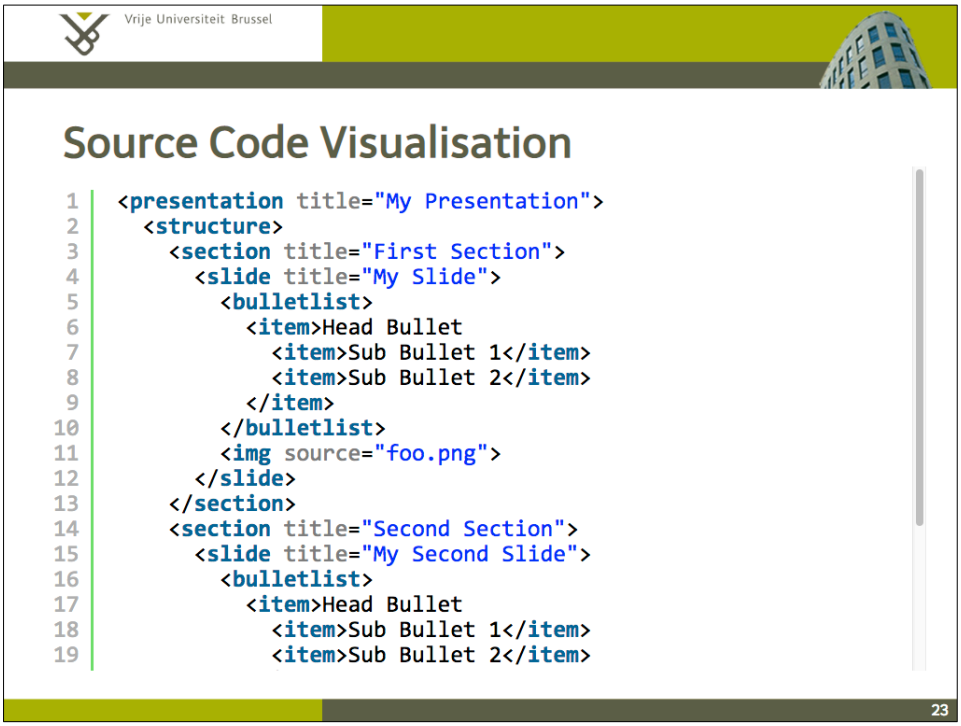


Figure 7.5: A MindXpres plug-in for static source code visualisations

[1] http://alexgorbatchev.com/SyntaxHighlighter/
[2] http://alexgorbatchev.com/SyntaxHighlighter/manual/brushes/

7.5 Interactive Source Code Visualisation for Teachers

The previously presented plug-in for source code makes it easier for presenters to present source code. As a next step we also present a plug-in for source code that makes the teaching of programming concepts via presentations more effective. This plug-in solution for interactive source code visualisation was created in collaboration with Paul Meștereagă, as part of his Master's thesis "*Intelligent Source Code Visualisation for the MindXpres Presentation Tool*" [190]. The plug-in and the research behind it has been published in two papers [226, 225] and has received a *student best paper award* at the International Conference on Computer Supported Education (CSEDU 2015). We continue with a summary of the plug-in but we refer to the original publications for more details.

The concepts taught in programming courses are far from trivial and the teaching of programming languages has proven to be a challenge for both students as well as teachers [135, 30, 105, 136, 162, 187]. Research highlights the importance of mental models when learning how to program, where a mental model is defined as "*a mental representation of the components and the operating rules of the system*" [185]. An incomplete representation results in an incomplete understanding of how the computer works and will cause novice programmers to have difficulties in writing correct programs [180, 192]. The other way around, a more complete mental model results in more effective learning and practising of programming [45]. To help reinforce the mental model of students, visual aids are identified as an effective solution [180, 255]. More specifically, Mayer [185] states that a possible solution for providing an effective mental model is to use visuals and to show the user the changes in state—such as variable changes—while the program is executed. The teaching of programming based on only presenting information such as syntax and structure in a lecture is not sufficient as it is not immediately clear how states change and there is a lack of contribution to a student's mental model [134].

We acknowledge presentations may not be the best medium to teach programming and ideally other teaching methods such as interactive lab sessions should be applied. Nevertheless, the majority of programming courses are at least partially taught via lectures accompanied by slide decks. In fact, these slide decks often form a major part of the study material and to make matters worse, slideware such as PowerPoint is a particularly unsuitable medium for presenting source code. Given that slideware is actively used in programming courses regardless of the unsuitability of the medium, our plug-in aims to improve the effectiveness of presentation-based teaching of programming. In the context of presentation tools there is little to no related work. However, the scientific community has produced various stand-alone tools and visualisations to help the teaching of programming. For instance, while working towards

a potential solution we investigated tools such as the Bradman tool [255], the VIP tool [290], Jeliot 3 [194], the notional machine [31], jGRASP [58], RGraph [235] and UUhistle [257]. As a presenter that wishes to use slideware, they either have to work around the limitations of the presentation tools or they have to use one of the standalone applications mentioned above, switching between their presentation and the external application on demand. Besides not being beneficial for the flow of the presentation, most of the standalone tools only focus on a very specific aspect of programming (e.g. recursion or memory management) and are made for a specific programming language. This implies that a programming course may require more than one of these tools to illustrate all relevant concepts or that a tool for the programming language used in the course might not even exist. In conclusion, projects such as CodeWitz [159] show that there is a clear need for better teaching tools in computer science, but as shown earlier, even support for simply displaying source code is lacking in presentation tools.

In our related publications we argue various requirements for a presentation-based solution. For instance, it should support the automatic indentation and syntax highlighting of code to reduce the effort required from the presenter. The tool should also integrate nicely in the presentation and should not disrupt the flow if the presentation. As there is no consensus on what programming language to teach in introductory programming courses [113, 68], we also require support for multiple programming languages and visualisations. Most importantly, we aim to make use of the findings of earlier studies and incorporate proven techniques to help students reinforce their mental models, for instance by providing graphical representations of certain programming concepts [101, 284]. It should be possible to step through the provided code in the order that it would normally be executed, which is not necessarily a linear progression from the first line to the last line of the provided code. As the presenter simulates the execution of the program, changes in the program's state are visualised and animated. Related work shows that different visualisations can help with different programming concepts [63] (e.g. memory pointers or recursion), so we also provide support for an extensible pool of visualisations that can be used with any programming language. Finally, it is also important that these features can be used with minimal effort from the presenter's side.

To achieve the goals we set, we present the general idea behind the plug-in in Figure 7.6. If a supported programming language is used, the plug-in handles everything automatically and the presenter only needs to provide the relevant snippet of source code. Similar to the basic source code plug-in discussed earlier, the programming language is automatically determined using a Naive Bayes classifier at compile time. The plug-in supports multiple programming languages by bundling language-specific functionality in so-called `Language Modules`. Once the programming language is

detected the corresponding language module is invoked to process the source code. In order to visualise the execution of the code at runtime we process the code in a language-aware manner at compilation time to extract additional data related to the execution flow of the provided code. A language module achieves this by applying a language-appropriate debugger. A debugger is a tool that examines a running application and offers functionality to provide insights about the program flow and for finding unwanted behaviour in the form of bugs. Debuggers are hard to use and they make no real effort to reinforce a mental model [255], but their output, a so-called execution trace, can be turned into something more meaningful by our tool. Nevertheless, debuggers are standalone applications dedicated to one specific language only and different debuggers produce output in different formats. For this reason a language module translates the resulting executing log into a generic language-independent format that can be used by the plug-in's visualisation engine for visualisations. This way, the runtime plug-in does not need to be able to process any code, it merely needs to be able to visualise execution logs in the generic internal format.

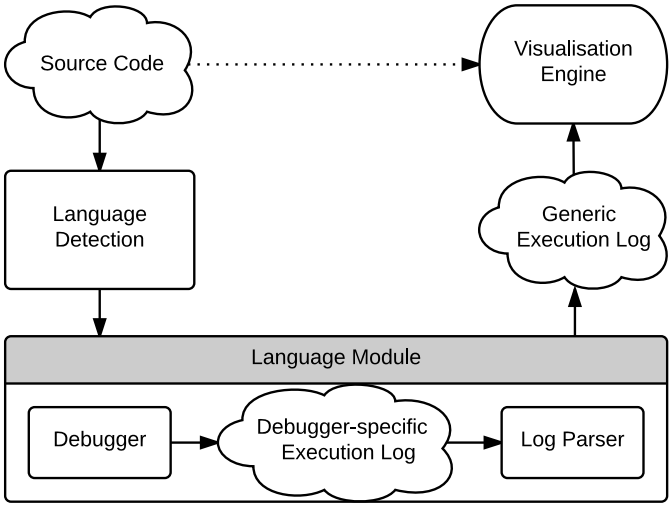


Figure 7.6: The process of turning source code into an interactive visualisation

At runtime the plug-in properly visualises the source code, allows the presenter to step through the code, visualise state changes (e.g. variables) and an optional visualisation can be selected to clarify the relevant programming concepts. The implementation uses Google's code-prettify[1] library for code formatting and syntax highlighting. The D3[2] visualisation library is used used for some of the optional visualisations. Our initial implementation comes with two language modes, one for C and one for Java. For

[1] `https://github.com/google/code-prettify`
[2] `https://d3js.org`

the creation of the execution traces, the C module uses the GDB[1] debugger while the Java version uses JDB[2], a debugger included with the Java Development Kit. Figure 7.7 shows a screenshot of the plug-in in action. In this example the plug-in is used to demonstrate the recursive implementation of the Fibonnaci function. The highlighted line indicates what line of code is being executed in the current step. The buttons can be used to go forwards or backwards in the execution and the presenter may also use the slider to jump to a particular point of interest. On the right-hand side the state changes for selected variables are shown. Given the topic of the code, a recursion tree is used as the optional visualisation which shows a history of the recursive function calls and subsequent backtracking process.

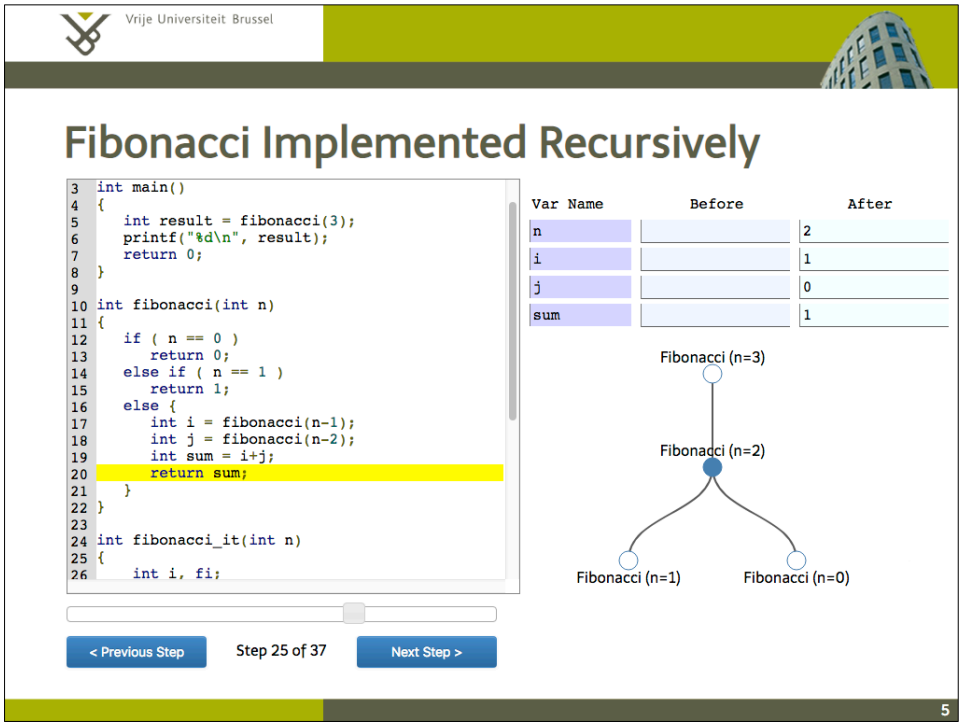


Figure 7.7: A MindXpres plug-in for interactive source code visualisation

To summarise, the plug-in aims to make the teaching of programming concepts via presentations more effective. To use the plug-in, the presenter just provides some source code and everything else is handled automatically by the plug-in. Language modules extract execution logs from the provided code and language modules can

[1] http://www.gnu.org/software/gdb/
[2] https://docs.oracle.com/javase/8/docs/technotes/tools/windows/jdb.html

easily be added to support additional programming languages. At runtime, the extracted execution log is processed by the plug-in to create a dynamic and interactive visualisation. Additional visualisations can be enabled and as the execution log is in a language-independent format these optional visualisations work can be reused for any programming language. The earlier example uses a recursion tree to help students understand how the function invokes itself multiple times. As another example, Figure 7.8 shows how the optional visualisation can be used to help explain a sorting algorithm as suggested by earlier studies on the subject [18, 94].

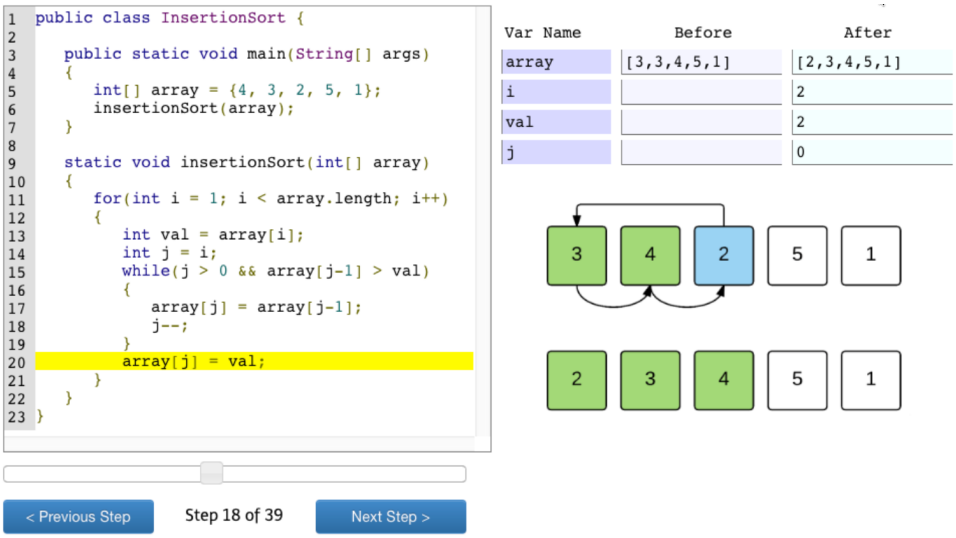


Figure 7.8: Visualising sorting algorithms, one of the many possible visualisations

7.6 Interactive and Narrative Data Visualisation

In Chapters 2 and 3 we discussed the benefits of dynamic data visualisation but we have also mentioned how existing presentation tools offer little to no support to create such visualisations. This plug-in solution for interactive and narrative data visualisation was created in collaboration with Yves Baeten, as part of his Master's thesis "*Intuitive Information Visualisation for the MindXpres Presentation Tool*" [19]. The plug-in and the research behind it has been published in two papers [222, 223] and was nominated for a *student best paper award* at the International Conference on Computer Supported Education (CSEDU 2016). We continue with a summary of the plug-in but we refer to the original publications for more details.

Storytelling has shown to be an effective approach for sharing insights gained by studying specific data sets [244]. Facts that have been tied together as part of a

story are easier to present as well as more memorable by the audience. Therefore, we have seen the rise of so-called narrative visualisations or visualisations that help us to tell stories with data [157]. For example, educational textbooks often contain various charts and diagrams that support a certain message that the text is trying to convey. These narrative visualisations have been adapted for recent media and are becoming more dynamic. For instance, news on the television might use animated graphs to show changes in oil prices or election results whereas online news articles might be complemented by user-controllable interactive visualisations.

Benefits of (Interactive) Data Visualisations

The main goal of such visualisations is to help users gain insight into a set of data that would be hard to interpret in its raw form. Abstract data representations can offer a high-level overview and help us to reinforce our mental model [177]. Such graphical representations make use of our highly developed ability to process the continuous stream of information-rich signals captured by our eyes [293]. Concepts such as shape, colour, size or distance are intuitive to us and the interpretation of some of these concepts comes naturally. Research in this domain led to Gestalt psychology, a research field that identified a series of laws helping us to understand these natural interpretations [83]. For instance, when comparing objects in a visualisation, it is clear that a larger object represents a larger quantity or higher importance. Similarly, objects that are spatially close to each other are likely to be more related than distant objects. The field of information visualisation tries to exploit these findings in order to facilitate knowledge transfer. Few [90] proposed a classification of eight messages that one might want to show using quantitative data, together with message-appropriate visualisation types. The messages include time series, rankings, part-to-whole, deviation, distribution, correlation, geospatial messages and nominal comparison.

The formation of a mental model can further be augmented by allowing the user to interact with the data [293]. The significance of interaction while processing information was illustrated in Gibson's cookie cutter experiment [102] and is often used as a classic example to prove the relevance of interaction in information visualisation. Gibson concluded that our brain performs better as active explorer, even if the act of exploring requires additional coordination and processing. Interaction techniques in information visualisation can be seen as the features that provide users with the ability to directly or indirectly manipulate and interpret representations. Note that this also includes menu interfaces that allow users to manipulate the representation and, for instance, switch to another chart type or sort a bar chart in descending order [302]. Furthermore, Dix and Ellis [70] emphasise two important principles in interacting with visualisations. The first principle "*same representation, changing parameters*" states

that users should be able to interactively change parameters of the presentation. The second principle "*same data, changing representation*" implies that a user should be able to switch between conceptually different data visualisations. Various representations can be appropriate for different types of data and each representation needs to be tuned for its purpose.

There are various academic studies about different interaction techniques such as zooming or filtering which resulted in the categorisation of frequently used techniques in information visualisation. One of the widely accepted classifications was independently proposed by both Siirtola [252] and Yi [302]. Even though the authors did not collaborate, the proposed interaction categories are very similar:

- *Select*: mark something as interesting
- *Explore*: show something else
- *Reconfigure*: show a different arrangement
- *Encode*: show a different representation
- *Abstract/Elaborate*: show more or less details
- *Filter*: show something conditionally
- *Connect*: show related items

Narrative Visualisations

Interactive visualisation techniques cover the exploration and analysis of data but there is also a need for presenting and communicating data effectively. As stated by Kosara, "*tying facts together into a story is one of the most effective ways of presenting them and making a point*" [157]. The main reason for using stories is the fact that they are known to be a popular way of conserving information and passing it on. Not only do narratives preserve and advertise information, they also act as an adhesive between facts to make them memorable [16]. Segel and Heer [244] further provide a classification of the different approaches and design techniques used in news media to visually tell stories. In the context of presentations, narrative visualisations are mainly author driven. The scenes and scenarios are linear and predefined by the presenter, messages and conclusions are explicitly mentioned and the audience has little to no influence on the story. This contrasts with reader-driven narratives found in other contexts where there is no prescribed ordering, the free interaction and exploration is central and possible interpretations are left to the reader. Segel and Heer state that ideally, visual narrative genres must balance a narrative intended by the author—with story discovery by the reader [244]. Kosara confirms that this also holds true for

collaborative settings where stories cannot only be used to support discussion and decision making, but also during the analysis process. Hence, stories can serve as a source for drawing conclusions, similar to the narrated history of an event [157]. Note that narrative visualisations can be manipulated to emphasise specific messages during free exploration. For instance, Hullman and Diakopoulos [129] identified a number of approaches and design techniques for prioritising particular interpretations in visualisations. These findings imply that narrative visualisations can be designed to deliver a predefined message without explicitly giving the message away.

Support in Presentation Tools

Even though the visualisation of information in graphs is an important feature of current presentation tools, existing presentation solutions clearly lack the interactive or narrative aspects discussed earlier in this section. PowerPoint makes it easy to visualise numbers stored in a spreadsheet and provides a lot of freedom in terms of chart types and styling options. Nevertheless, the final result of this process is always a static graph. Of course, as with any content in PowerPoint, it is possible to apply transitions (e.g. fade in or slide out) and motion path animation effects. These effects can either be applied to the complete chart or, depending on the type of chart, to smaller parts within the chart. By using these transitions and motion path animations as a workaround, authors can compose basic narratives by, for example, making parts of a pie chart appear one by one. However, this approach has several shortcomings. First of all, it requires a major authoring effort since animations have to be manually applied to the different parts in order to achieve the desired effect. Furthermore, things might get even more complicated when changes have to be made at a later stage. In order to switch to another chart type, it might further be necessary to define multiple versions of the graph with the corresponding transitions between them. Second, if we depend on these transition effects, the result consists of a predefined sequence of states and there is no way to deviate from this fixed path. While it can be desirable to predefine a path through the data, it might also be beneficiary to have the flexibility to show alternative unprepared variations when answering unexpected questions. Last but not least, it is important to note that a chart is rendered only when the underlying data or configuration is changed at authoring time, but from then on the chart has to be considered a collection of static images. This implies that any effects only operate on the graphical level but cannot do anything that would require the components of the chart to adapt between steps. We can make the bars of a bar chart appear one by one but it is impossible to apply modifiers to the information or configuration that defines the graph. For example, we cannot just switch to another chart type, change the scale of a graph or filter out specific values as a step in the animation. There are third-party plug-ins such

as oomfo[1] or think-cell[2] which add even more options for creating charts, but one has to be aware that these third-party plug-ins typically only add additional authoring and styling features for designing what will ultimately result in a static chart with the same limitations. So far we have only discussed charts in PowerPoint but we came to similar conclusions for alternative presentation tools such as Apple's Keynote or Prezi.

Support in External Tools

In terms of academic work, there are a number of tools based on the interactive visualisation principles discussed earlier. Notable examples are VICKI [64], Spotfire [4] and GGobi [266]. While these are promising tools founded on the principles of proven concepts, they also show a number of shortcomings which make them less suited for use in presentations. First of all, these solutions were built as standalone applications and their interfaces are not optimised for use during a presentation. The presenter has to leave the presentation and switch to another application which interrupts the flow. These tools also consist of multiple windows and have complex menus that do not translate well to the limited resolution offered by most projectors. In addition, significant interaction is needed to operate the tools, requiring the presenter to focus on the software and use the keyboard or mouse to go through a series of actions to switch between desired visualisations. It is evident that these solutions focus on the interactive exploration part, but the ability to use them as narrative visualisation tools is rather limited. Commercial solutions with similar restrictions include IBM's Many Eyes [289] and Tableau[3]. Note that the previously mentioned GGobi also provides an Application Programming Interface (API) that allows programmers to embed and pragmatically interact with visualisations. There are other development frameworks such as UC Berkeley's prefuse visualization toolkit[4] for the Java programming language or the popular D3[5] JavaScript library. While these frameworks offer a broad range of features for modern data visualisation, they are usually used for building standalone applications. More importantly, they require the programming of the desired visualisation which is not suitable for the majority of presenters.

Hans Rosling's 2006 TED talk entitled 'The Best Stats You've Ever Seen' [233] is an excellent example of the fact that it is possible to build a presentation around dynamic and interactive data visualisation. During his talk, Rosling made the point that there is so much data related to human development trends but it is difficult to educate people and transfer knowledge about current issues if we cannot present these statistics in

[1] `http://oomfo.com`
[2] `https://www.think-cell.com/en/product/`
[3] `https://www.tableau.com`
[4] `https://github.com/prefuse/Prefuse`
[5] `https://d3js.org`

an accessible way. For his presentation, he used a proprietary tool (now forming part of the Gapminder[1] suite) that allowed him to animate and visualise data over time, switch between chart types or highlight areas of interest and annotate them. The success of the talk can partly be attributed to Rosling's energetic personality and compelling arguments, but also his novel approach to presenting data gained a lot of attention [157] and has been explored in great detail. Robertson later showed that animated transitions can have a negative effect on a viewer's ability to follow trends [220], but because these animated transitions are entertaining and capture the attention, they work well in front of a live audience. While Rosling's 2006 TED talk was definitely a major step in the right direction, the Gapminder series of tools also has some shortcomings. First of all, once more they are standalone applications and require a presenter to switch between presentation and tool. More importantly, these tools were specifically built for educating people about certain topics related to human development. This implies that the data sets are fixed and the functionality and visualisations are tweaked for drawing conclusions from geographic and demographic data over time. Other tools have been built for specific use cases, including the MediaViz [67] platform for visualising data relevant to online media studies. Similarly, ArtVis [76] is a tool for exploring European art over time on a map-based visualisation. GeoTime [78] represents another geography-based visualisation tool focussing on creating a visual story out of geo-temporal events. While GeoTime is one of the few tools where the creation of a narrative out of a raw data set lies in its core, its use is limited due to the focus on geo-temporal data only.

The discussed related work highlights the added value of interactive and narrative visualisations even if we have to conclude that existing presentation tools do not offer the necessary support for applying such narrative visualisations in practice. There are some workarounds such as creating multiple static charts with manually-defined transitions between them, but often presenters are not willing to make this effort and rather opt for a less dynamic narrative. Alternatively, it is possible to use stand-alone tools which were not designed to be used in the context of live presentations and can therefore not easily be applied as tools for narrative visualisations.

Towards More Effective Visualisations in Presentations

As previously shown the use of interactive and narrative visualisations can be beneficial, but creation of such visualisations in presentation tools is often cumbersome and they are therefore not applied as often as they should be. To address this issue we present a new MindXpres for interactive and narrative data visualisations, making the presentation of raw data more effective and less cumbersome.

[1] `http://www.gapminder.org/downloads/`

Research in the field of information visualisation and narrative visualisation shows that the use of specific visualisation techniques can lead to improved knowledge transfer. However, as discussed earlier we see that existing presentation tools do not exploit these visualisation techniques to their full potential. Our goal is to close this gap and to apply lessons learned from interactive information visualisation as well as narrative visualisation in order to improve presentation-based knowledge transfer. Based on the presented related work and the shortcomings of existing presentation solutions discussed earlier, we derived a set of requirements for interactive information visualisation in presentation tools:

R1: Integration in presentation tools As slide decks are one of the most frequently used media for transferring knowledge in education and business settings, it is preferable to directly integrate interactive visualisations into a presentation rather than relying on third-party applications. If an interactive visualisation is not integrated into the presentation tool, the presenter is forced to switch between applications which takes time and interrupts the presentation flow.

R2: Focus on proven techniques and guidelines Popular presentation tools put their main focus on aesthetics and looks but the offered features are not always beneficial in terms of knowledge transfer. For instance, the ability to show three-dimensional bar charts or pie charts has been proven to cause longer interpretation times and may even be interpreted incorrectly [248, 91]. Similarly, Tufte [277] argues that most graphical bells and whistles (what he calls "chartjunk") increase the signal-to-noise ratio and dilute the message one wants to deliver. Presentation tools should not only create visually appealing visualisations but also support the presenter in creating visualisations that focus on strengthening the viewer's mental modal and transferring knowledge more efficiently. Therefore, a presentation tool should offer features based on the message that the presenter is trying to pass on, for instance based on Few's classifications introduced earlier [90]. Note that this is not only relevant for static visualisations but should also apply to the currently non-existent interactive features by, for example, basing ourselves on Siirtola's classification of relevant tasks for data exploration [252].

R3: Interactive visualisations as support for oral narratives When using interactive and dynamic visualisations as support for an oral narrative, it is desirable to be able to predefine a sequence of views for a given data set and to step through these views during the presentation. In addition to simple enter and exit animations offered by existing tools, it is important to be able to apply the two interaction principles by Dix and Ellis [70] introduced earlier. This implies that it should be possible to modify parameters in between the steps of a presentation (e.g. change the scale or apply a

filter on the data) and to change the data representation (e.g. by switching to another chart type). By allowing the presenter to define such a sequence of states, they can synchronise the visualisation state with the oral narrative at preparation time and ensure that limited interaction with the computer is needed during the presentation.

R4: Unscripted data exploration In addition to stepping through the predefined states of a visualisation, the presenter should also be able to change the representation or parameters at any time during a presentation. Segel and Heer [244] pointed out the importance of balancing the narrative intended by the author with story discovery by the reader. This also applies to certain presentation styles where questions or discussions with the audience can drive the presentation. Therefore, a presentation tool should also allow the presenter to interact with the visualisation during the presentation with the same set of interactions offered at authoring time. Since the resolution (screen real estate) and interaction is limited during a presentation, special care needs to be taken to offer the available interactions in a way that does not clutter the visualisation and can be controlled without intensive user input.

R5: Interactivity after the presentation As mentioned earlier, readers or audience members should not be excluded from the interaction. This does not only apply during a presentation but should be valid for a slide deck's entire lifetime. For example, in higher education slide decks are often offered as part of the study material. A student reviewing the slides at home should at least be able to play back the visualisation as it was defined by the presenter. Ideally, students should also have the option to freely navigate the data in order to clarify any doubts they may have and to strengthen their mental model by exploring the data set. Another use case is the inverted or flipped classroom setting where activities that are typically considered homework become central during class and the teacher merely guides the completion of these activities [33]. By offering students the interactive slide decks that were used in the pre-recorded lectures, they are not only able to replicate situations from the videos, but they also have a tool for further data exploration in order to come to their own conclusions.

Implementation

To summarise the functionality of the plug-in, it allows presenters to simply provide a dataset and to record a series of views on the data that need to be shown during the presentation. In other words, the presenter preprograms a sequence of visualisation states. In between the views the visualisation can be transformed and manipulated by using proven techniques such as those discussed in Section 7.6. For instance, the

representation can be changed (e.g. from a bar chart to a pie chart) or data can be filtered and partitioned in between views to match the presenter's narrative. The plug-in handles styling and visualisation automatically, so the presenter focusses on creating an outline for a visual story, but the actual details and handled automatically. We based ourselves on established guidelines such as those by Tufte [277] to automatically generate effective visualisations, and we base ourselves on classifications such as Few's [90] to support all chart and graph types a presenter might need to convey different messages. Supported visualisation types include (stacked) bar charts, pie charts, scatter plots, line charts and geographical maps. These choices are motivated in more detail in the original publications [222, 223]. The original publications also provide a detailed overview of the operations that can be applied to the visualisation in between views, and include for instance applying highlighting, changing visibility, filtering, selecting, grouping or changing visualisation parameters such as scales or legends.

The plug-in makes heavy use of the D3[1] visualisation library and also uses C3[2] which builds further on D3 and provides abstractions for some commonly used charts. Visualisations are predefined by the presenter in the MindXpres XML-based format and the exact vocabulary is detailed in the original publications. Data can be provided directly in the MindXpres format, or it can be provided as an external file in plain text, CSV, XML, HTML or TSV formats, or as an Excel spreadsheet. In the case of an Excel spreadsheet a compile-time trigger in the plug-in will extract the referenced data from the spreadsheet and bundle it with the presentation in the JSON format for easier runtime processing. We explained how the presenter can predefine a dynamic visualisation that changes over time, but all the provided operations can also be applied during the presentation to encourage free exploration and improvised changes. Figure 7.9 shows some of the interactions that can be performed during the presentation.

Use Case

We present a short scenario that demonstrates how one can use the plug-in to create a narrative visualisation. In contrast to existing presentation tools, a presenter only needs to create one visualisation with one data set for which they define a sequence of views that support their oral narrative. This not only requires much less time and effort than existing workarounds, but it also makes it easier to apply changes at a later point in time. In the presented scenario, the goal is to compare the tax and social burdens of salaried employees in the European Union. As a starting point, a JSON file that contains the relevant data for the year 2014 is used [231].

[1] `https://d3js.org`
[2] `https://c3js.org`

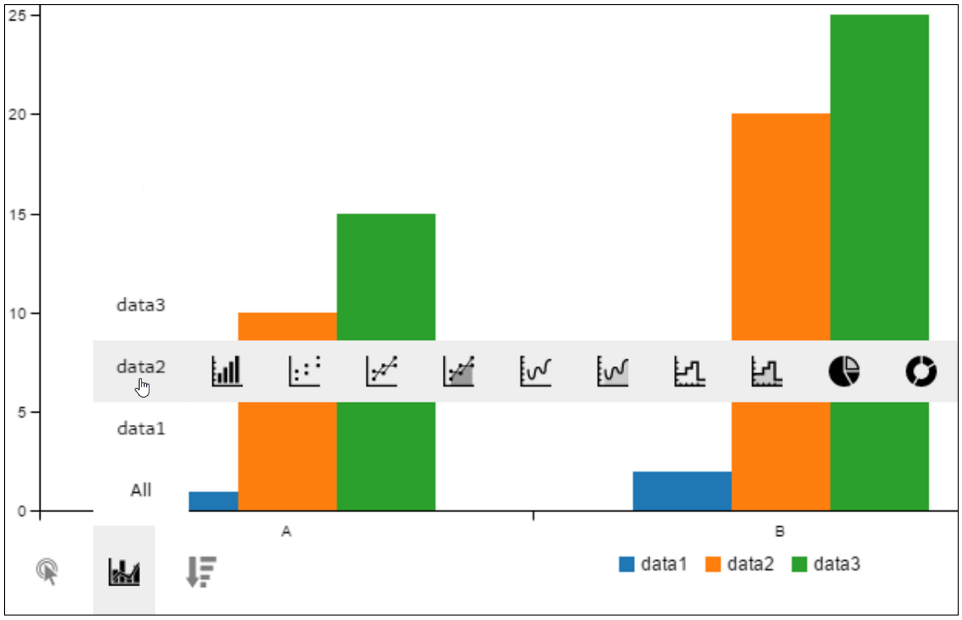
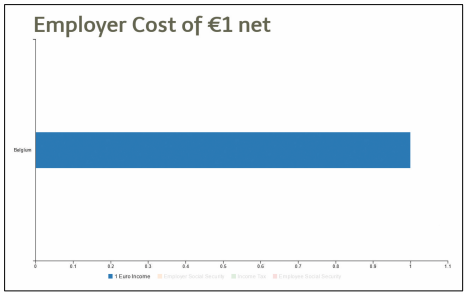


Figure 7.9: Ad-hoc interaction with the data visualisation during a presentation

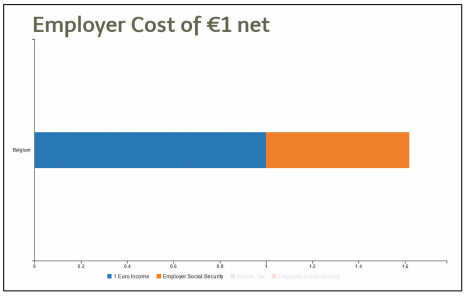
A small snippet of the JSON-based dataset is shown in Listing 7.2. A dynamic and interactive visualisation is then used to illustrate the extra money an employer has to pay in order that an employee will receive exactly one euro after taxes. Figure 7.10 shows some of the views that the presenter declared to help support his narrative. Assuming the presentation is going to be delivered to a Belgian audience, the presenter starts with an explanation of the tax situation in Belgium. The first views introduce taxes such as employer social security, income tax and employee social security. Then, the representation is changed to a pie chart to demonstrate how taxes make up more than half of the amount paid by the employer. Further views compare the situation with the EU average or with all individual European countries. Operations such as zooming and sorting are used to further clarify how much taxes differ within the EU. Finally, the data is overlaid on a map-based visualisation.

```
[
  ["Austria", 1, 0.5, 0.32, 0.29],
  ["Belgium", 1, 0.62, 0.5, 0.22],
  ["Bulgaria", 1, 0.22, 0.11, 0.16],
  ...
  ["Spain", 1, 0.39, 0.21, 0.08],
  ["Sweden", 1, 0.42, 0.33, 0],
  ["United Kingdom", 1, 0.14, 0.2, 0.12]
]
```

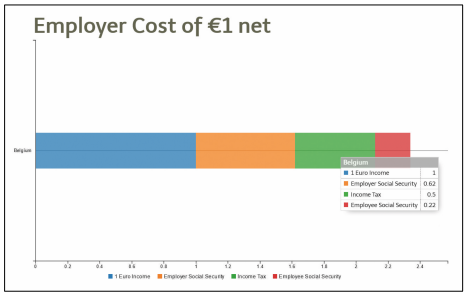
Listing 7.2: A snippet of the JSON data used in the scenario



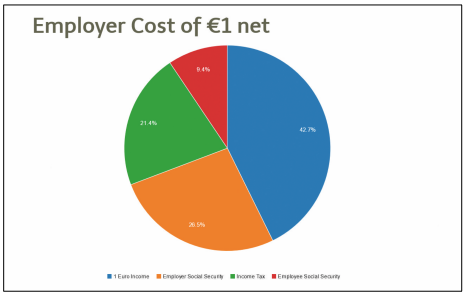
(a) One Euro received by employee



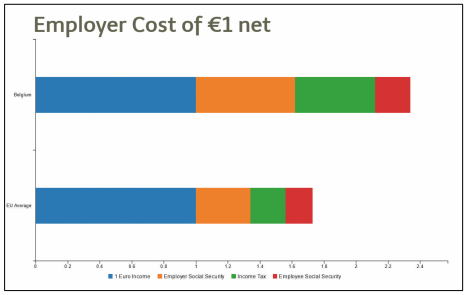
(b) Add employer social security



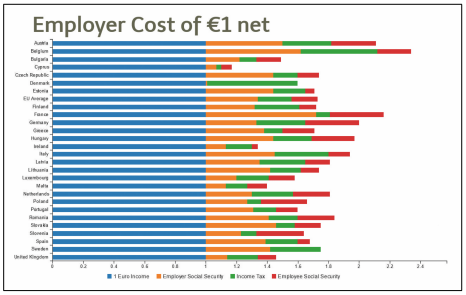
(c) Add income tax and employee security



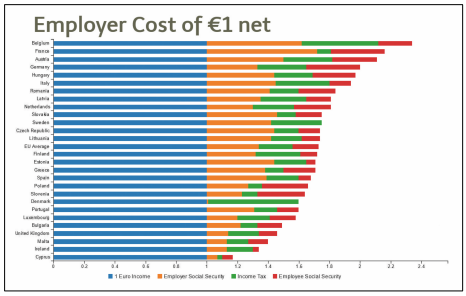
(d) Switch to pie chart



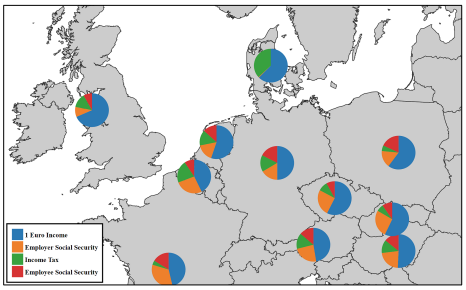
(e) Compare with EU average



(f) All countries (alphabetically)



(g) All countries (sorted by value)



(h) A map-based visualisation

Figure 7.10: Various graph-based phases of a visualisation for EU employer costs

7.7 Embedded Linux Environment

As another example of embedding interactive content directly in a presentation we present a plug-in for embedding a functional Linux environment in MindXpres. With the teaching of IT-related courses in mind, the plug-in allows presenters to demonstrate concepts related to programming or system administration directly from within the presentation. We make use of Fabrice Bellard's JSLinux[1], which uses WebAssembly to run lightweight virtual machines (VMs) directly in the browser. Since VMs are used, the plug-in has no dependencies to the operating system that runs the presentation and it is not required to run the presentation on a Unix-based host system. As an example of an interesting use case, lightweight VMs can be preconfigured and embedded by the presenter so that after the presentation students still have a working environment to perform a certain task or to replicate what was shown during the presentation. For instance, Figure 7.11 shows a scenario where the VM is setup to have a C compiler installed and already contains a small C program. This makes it easy for the presenter to demonstrate how the program can be compiled during the presentation or for students to replicate the example after the presentation. This plug-in was created as a proof of concept and additional work is needed for use in the real world. For instance, our prototype plug-in lacks methods to easily predefine VMs and parts of JSLinux are not open source resulting in some restrictions.

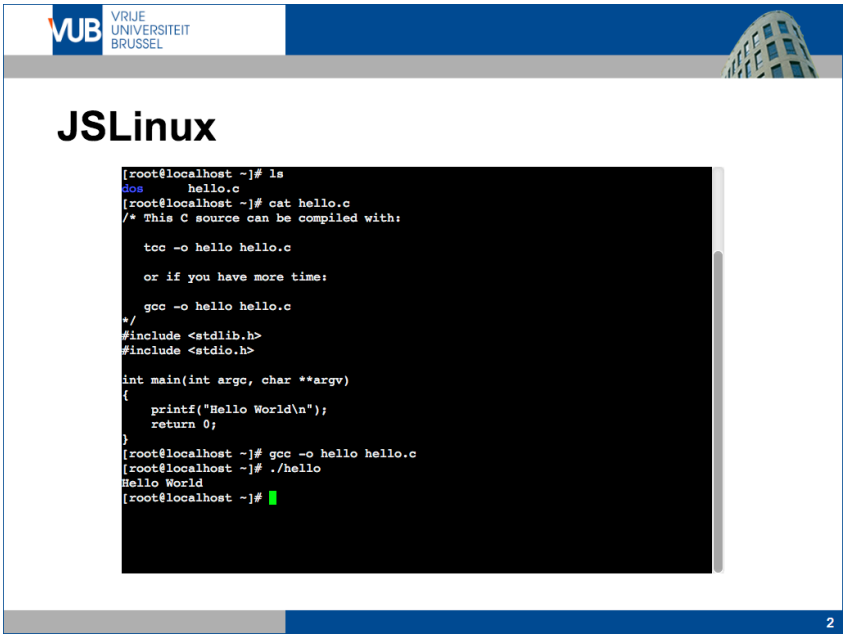


Figure 7.11: Embedding a preconfigured lightweight Linux VM in MindXpres

[1] https://bellard.org/jslinux/

7.8 Doom

It is often joked that the 1993 video game 'Doom' runs on practically any platform or hardware. As a running joke the community has ported the game to run on unusual platforms[1] such as oscilloscopes, fridges, calculators or watches. Intended as a joke, someone stated that MindXpres would not be complete without being able to run Doom. We present the following plug-in for embedding a playable version of Doom directly in a MindXpres presentation. Although the academic contribution of this plug-in is debatable, we include it as another example of how easy it is to include dynamic and interactive content in MindXpres. It also demonstrates how developers can make use of interesting existing libraries to quickly create novel use cases for MindXpres. The plug-in is implemented using a JavaScript version of the DOSBox emulator[2]. Listing 7.3 shows the entire `render` method of the Doom plug-in. The method could be implemented in a more elegant way, but we provide it like that as a minimal example of how MindXpres can easily be extended. Other than this method, the developer only needs edit the plug-in manifest to specify that the `<doom>` tag in the XML authoring language should be mapped to this plug-in.

```
1  class DoomPlugin extends MindXpresPlugin
2  {
3     render(tag, element)
4     {
5        let resourcePath = this.getPluginResourcePath("doom19s_deiced.zip");
6        $(element).prepend(
7           `<style type="text/css">
8              .dosbox-container { width: 640px; height: 400px; }
9            </style>
10           <script>
11             var dosbox = new Dosbox({
12               id: "dosbox",
13               onload: function (dosbox) {
14                 dosbox.run(${resourcePath}, "./DOOM.EXE");
15             }});
16           </script>
17           <div id="dosbox"></div>`);
18    }
19 }
```

Listing 7.3: The complete implementation of the Doom MindXpres plug-in

[1] https://www.reddit.com/r/itrunsdoom/
[2] https://js-dos.com

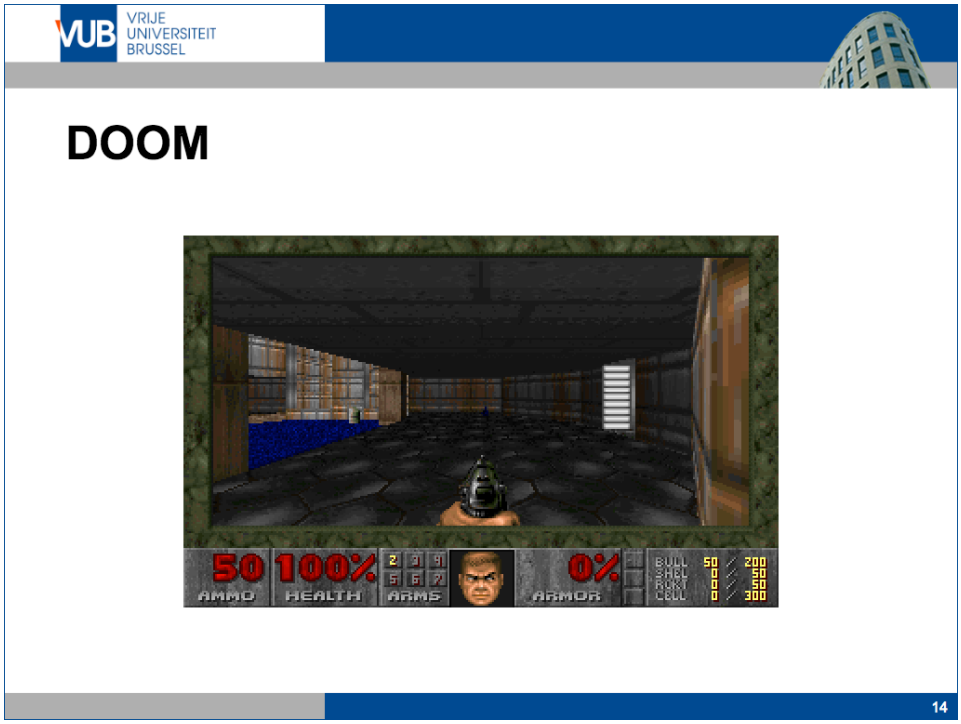


Figure 7.12: Embedding a playable version of Doom in MindXpres, based on DOSBox

Chapter 8

Solutions for Involving the Audience

In Chapters 2 and 3 we have shown the need for better involving the audience during a presentation. We have for instance discussed how active learning helps the audience to retain information. In Chapter 6 we have further motivated the need for connected presentations and we provide the technical foundation for such functionality. We have also shown how the design and implementation of the MindXpres platform allows presentations to form networks that allow plug-ins to communicate and additional hardware to be integrated for a variety of purposes. In this chapter we present plug-in solutions that build further upon these concepts and offer functionality related to involving the audience during a presentation.

The presented plug-in solutions in this chapter have been created in collaboration with Christophe Vermeylen, as part of his Masters's thesis "*Audience-driven Presentations based on the MindXpres Presentation Tool*" [288]. The plug-in and the research behind it have been published as a conference paper [230] and has received a *best paper award* at the International Conference on Advanced Learning Technologies (ICALT 2014). We continue with a summary of the plug-ins but we refer to the original publication for more details [230].

8.1 Polls and Quizzes

As a basic example of MindXpres' networking functionality we provide a plug-in for polls and quizzes. The idea is of course not new and there are many existing solutions for polls and quizzes. It is a common activity that comes with most clicker systems and

there are also other solutions such as Poll Everywhere[1] that allow audience members to use their personal mobile devices to participate in polls. However, although our implementation serves mainly as an example for plug-in developers, it does offer some benefits over similar tools. As explained in Chapter 6, different hardware can be combined and used in the same activity. For instance, some audience members could be using clicker devices while others might be using digital pens or smartphones. Furthermore, MindXpres was designed with the after-life of presentations in mind, and in this case the results of a poll or quiz can be recorded by the plug-in so that they are still available to users that view the presentation at a later stage. Last but not least, a MindXpres network can be formed locally with a hotspot without requiring a working Internet connection at the location of the presentation. Figure 8.1 shows a screenshot of the plug-in in action.

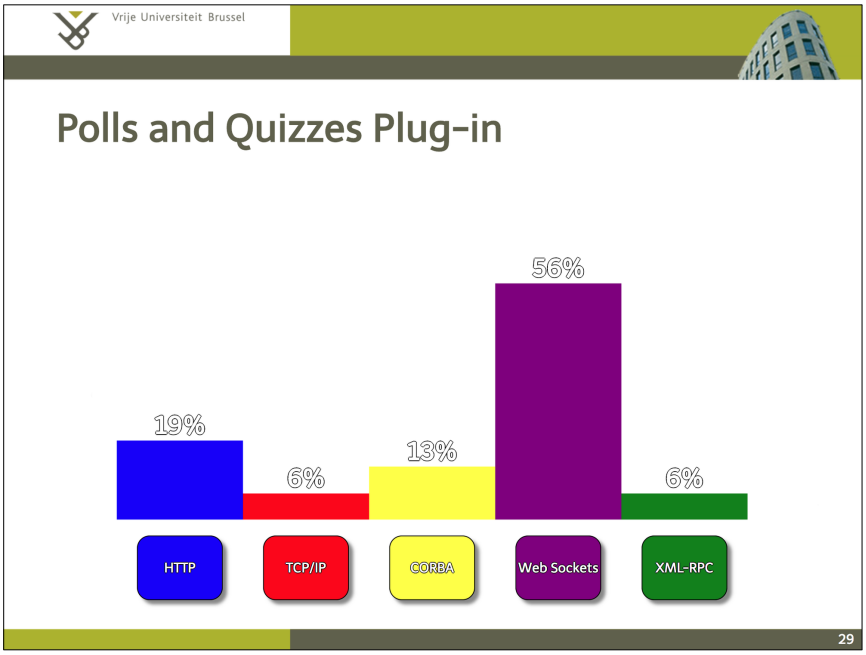


Figure 8.1: The polls and quizzes plug-in allows the audience to use a combination of smartphones, clickers and other devices for voting

As most plug-ins that require connectivity, the polls and quizzes plug-in makes use of the communication engine built into MindXpres. When an audience member selects an option in a poll, the poll plug-in on their device publishes the user's choice to the access point's publish-subscribe engine. From there, it is forwarded to the poll plug-in that is part of the presenter's MindXpres instance since it is subscribed

[1] `https://www.polleverywhere.com`

to receive such events from the access point. The poll plug-in that is part of the presenter's presentation can then update the count accordingly. Data adapters built into the access point allow votes that originate from different devices (e.g. clickers, smartphones or digital pens) to be converted into a more generic event before they are propagated in the network. This way, the poll plug-in does not need any logic to support different devices. Mediation is done automatically by the access point and the poll plug-in is notified with a generic event whenever a vote is made and does not need to care where the vote originated from.

Similar to most content plug-ins, polls are authored in the XML-based MindXpres format. The plug-in also offers basic functionality for managing the poll during the presentation. For instance, the questions can be hidden until the presenter opens the poll. The presenter can also close polls at any moment and the results can be recorded for later use.

8.2 Navigation Mirroring and Takeover

As a next plug-in with audience-driven functionality we present a plug-in for synchronising navigation across presentations. The plug-in allows audience members to always see the current view of the presentation on their own device. For instance, when the presenter moves to a next slide or region on a canvas, the presentation on the audience members' devices change to also display the new position in the presentation. Similarly, when the presenter interacts with content plug-ins such as the interactive source code visualisations shown earlier, devices in the audience are also synchronised so that their view is an exact copy of the presenter's presentation. The navigation mirroring plug-in can be seen in action in Figure 8.2.

Note that if the MindXpres access point has an Internet connection, others can also remotely follow a presentation in real-time without being at the same location. Audience members can choose if they want to have a synchronised view and they can always disable the synchronisation to break free and navigate on their own. As an additional feature, the plug-in allows audience members to take over navigation from the presenter. To use this feature, audience members must make a request from within their presentation. If the request is approved by the presenter, they will be able to navigate freely in the presentation and the view will be synchronised across all connected presentations, including the presenter's presentation up front. This is particularly useful when audience members have a specific question and want to revisit some relevant content in the presentation, for all to see. As many communication-based plug-ins, the implementation makes use of the access point's publish-subscribe mechanism.

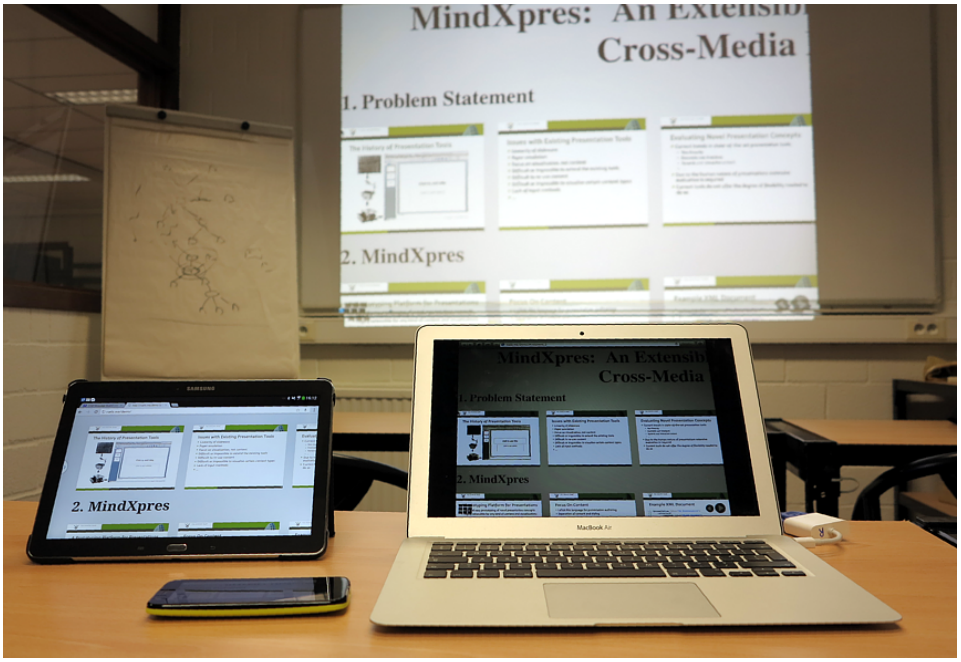


Figure 8.2: Synchronising navigation across different kinds of devices

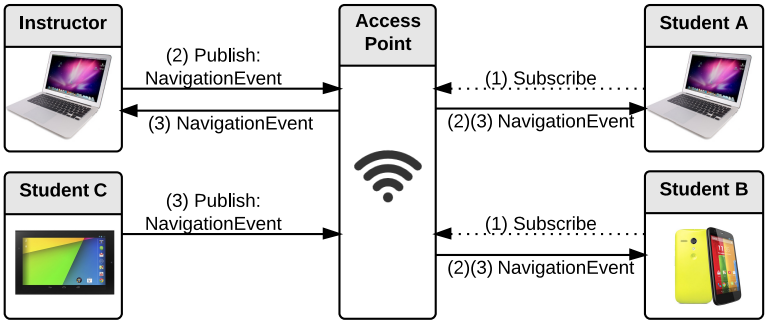


Figure 8.3: Synchronising views using the access point's publish-subscribe mechanism

Figure 8.3 illustrates how this synchronisation works for this particular plug-in. The illustration shows how the plug-in on the audience devices (e.g. Student A and Student B) first subscribes to the access point in order to receive events of the type `NavigationEvent` (Step (1)). Whenever the instructor's presentation detects a change in the view, it publishes a `NavigationEvent` containing the relevant data (Step (2)). The access point then forwards this event to devices that registered for this type of event, allowing them to update their view to match the presenter's

view (Step (2)). The illustration also shows the scenario where an audience device takes over navigation which causes other devices to synchronise their view (Step (3)). Note that the same mechanisms can be used to provide multiple users with the ability to navigate at the same time, for instance for collaborative presentations [60, 49]. We point out that some years after we published this idea, Microsoft Research has published and evaluated a similar system called Office Social [48, 49].

8.3 Digital Questions

Next, we show a plug-in that allows audience members to digitally ask questions. Some presenters or presentation styles try to avoid interruptions during the presentation and this plug-in allows audience members to ask their questions in digital form from their own devices. Audience members see each other's questions and can upvote or downvote questions. This way, duplicate questions are avoided but more importantly, it gives the presenter insights on which parts might need some further clarification. For instance, at a conference where the allocated time for questions is limited, the presenter may choose to only answer the top three questions. Figure 8.4 shows a screenshot of the question plug-in in action. Note that questions can also be recorded by the presenter for later processing.

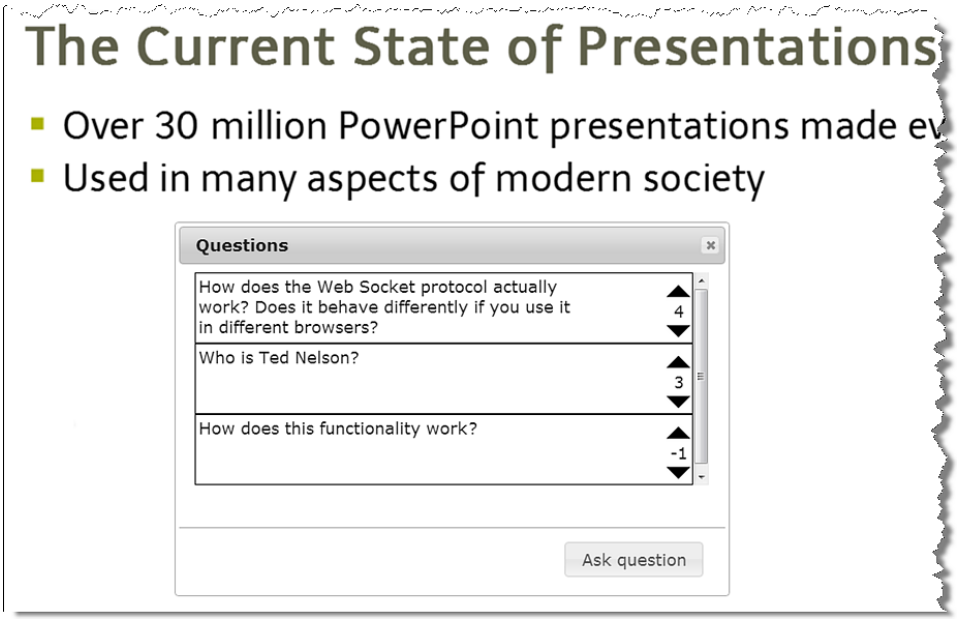


Figure 8.4: Allowing audience members to ask their questions digitally so that they can be discussed at an appropriate moment

8.4 Shareable Highlights and Notes

As a final plug-in for audience-driven functionality we present the notes plug-in for making shareable highlights and notes in a presentation. The plug-in allows audience members to highlight parts of the content and to attach small notes, similar to how some PDF viewers allow users to take notes. These annotations can be made during the presentation but also after the presentation if the presentation is hosted on MindXpres-enabled infrastructure. This is for instance useful for students who often use presentations as study material and might make notes to clarify certain things. Figure 8.5 shows a screenshot of the plug-in in action. More importantly, the plug-in allows notes and highlights to be shared between selected users. As stated by Lim [173], in recent times students have taken a more active role and they search and collect information, and share their findings. The presented plug-in for shareable notes allows students to attach and share any additional information they might find elsewhere.

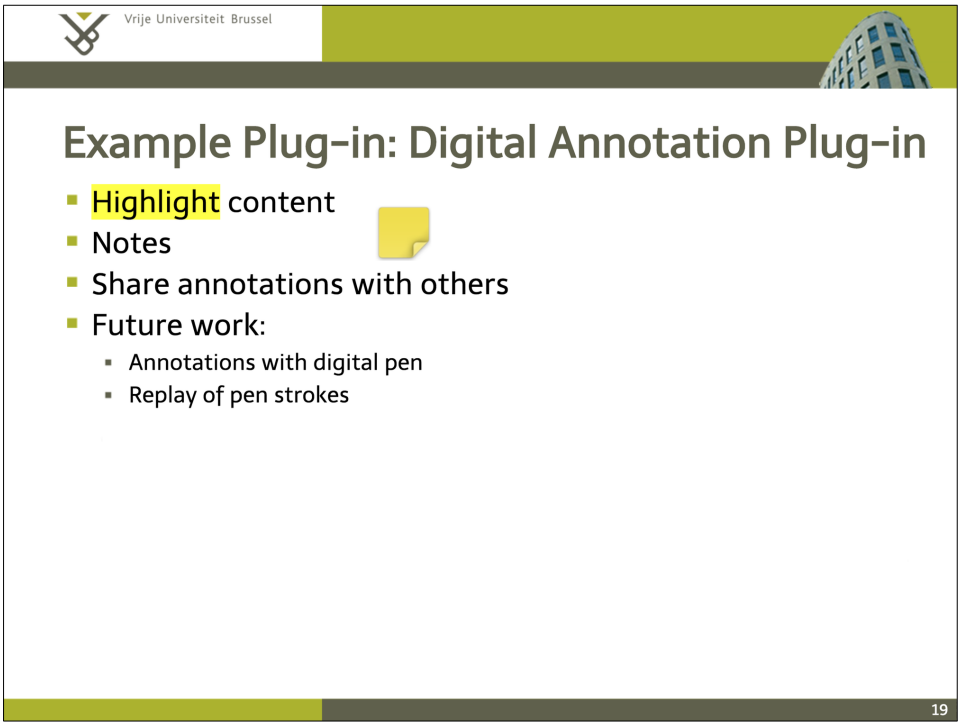


Figure 8.5: Shareable highlights and notes in a MindXpres presentation

Chapter 9

Solutions for Supporting the Presenter During the Presentation

In Chapters 2 and 3 we have shown the need for functionality that provides better support for the presenter during presentations. We have for instance highlighted issues related to the mobility of the presenter, the limited input modalities, the limited interactions with the presentation and the need for better presentation skills in general. In this chapter we provide some solutions that aim to help the presenter to interact with the presentation and to be more mobile. We further describe a modular feedback system that can provide feedback on the presenter's verbal communication, body language or presentation content.

9.1 Multimodal Presentation Interaction

As a first plug-in that aims to support the presenter we focus on how presenters interact with presentations and investigate the use of additional modalities for new types of interactions. This plug-in solution for multimodal interaction was created in collaboration with David Bos, as part of the Master's thesis "*Multimodal Interaction for the MindXpres Presentation Tool*" [39]. We refer to the original thesis for details, but a summary of the research is provided to motivate the purpose of the presented plug-in.

One common critique of existing presentation tools is that presenters are glued to their computer in order to navigate the presentation with their keyboard or mouse. These issues related to mobility have also surfaced in the observation and survey results as described in Chapter 3. There are of course tools that can help such as presenter remotes. However, these tools are usually limited to navigating forwards or backwards and offer little benefit in next-generation presentations which might not even be based on a sequence of slides. PaperPoint [251] is an existing solution based on physical paper and digital pens, and the tool provides the ability to navigate presentations in a non-linear way while keeping the presenter mobile. Our plug-in solution is the result of a further investigation where we examined some additional modalities for interacting with a presentation, without depending on special hardware.

Related work focusses mainly on the use of interactive whiteboards or smartboards, a well-known multimodal interface used in classrooms and office meetings [254]. These are mostly large, touch-sensitive boards which control a computer connected to a digital projector where the presenter can interact with the whiteboard by touch, using their fingers or by using a non-ink pen tool. SMART Technologies[1] is an example of a company that provides such products. However, smartboards are of course not present everywhere and can be expensive. In order to come to a hardware-independent solution we first investigated the benefits of these systems. Related work shows that users prefer using a smartboard to a computer because they found the keyboard and mouse difficult to operate [108]. It is also reported that simply using touch results in a more efficient presentation and more professional delivery of multimedia resources [271]. Teachers also appreciate the possibility to switch between different "pages" on the board for supporting a wide range of needs within a class spontaneously [168]. Compared to traditional whiteboards, also the option of having colour, sound and video is appreciated [59]. More interactive features such as annotating, drawing diagrams and labelling items are mentioned as benefits, especially if they can be saved and reused later [168]. Although it takes some time to prepare lessons using a smartboard, teachers reported that they believe that the planning time will be reduced thanks to the ability to save, share and reuse the materials [168]. Finally, a topic that has already been touched multiple times in this dissertation, the increased interactivity and involvement of the audience can encourage pupils to participate [168, 254, 189] but depends on how the tool is used [144].

As presentations are a communicative event, the visible content is not the only factor that affects the transfer of knowledge. Also the presenter's actions such as pointing to content and highlighting certain aspects have an impact and are most specific to this genre of communicative event [150]. Currently, a lot of presenters also use their computer's mouse or trackpad to perform these actions so we further investigated

[1] `https://education.smarttech.com`

the use of pointing and highlighting in presentations. The meaning of gestures can be reconstructed by identifying the particular gestural forms that carry meaning [143]. Most popular is pointing with the index finger, but a lot of technical instruments are also available to support pointing: sticks, pens and pencils as well as a computer mouse can be used to point in PowerPoint presentations. The most popular for common slideware is the use of the laser pointer. As noted by Knoblauch, the fact that pointing also includes pointing by technical aids means that the signification of pointing is not dependent on the sign itself as gesture studies would suggest [150]. Pointing in this case is directly related to other aspects of communication: speech and visuals. It is this relationship which makes certain passages of a presentation decipherable: the presenter can point at an illustration to make his talk decipherable. Pointing movements can also be understood as interpretations of what the speaker is saying and what they are hinting at [150]. It is safe to say that pointing does much more than just refer to something given. Its the interplay between pointing, speech and visuals that give meaning. Taking one of these out of the equation leads to a reduction in communicative performance of the presentation as a whole. Highlighting text has the same contextual meaning as pointing: it signifies the highlighted text and lays of a path for the reader to determine more easily what is important. This highlighting has proven to have a positive influence on retention if the highlighted part is significant. However, highlighting insignificant and unrelated parts of a text can have the opposite effect [118].

Based on these findings we focussed on creating a solution that does not depend on expensive hardware, and enables more natural navigation, annotation, highlighting as well as spontaneous content creation. The system enables presenters to use a laser pointer, their voice as well as their body for this additional functionality, allowing them to be more mobile. Our solution is based on the relatively inexpensive Microsoft Kinect

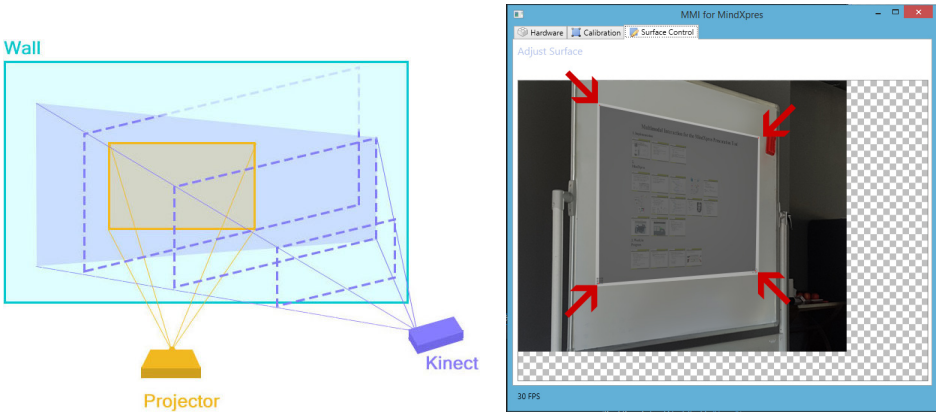


Figure 9.1: Mapping the projection area to the Kinect's coordinate space [39]

and we make use of the RGB camera, depth camera and microphone for detecting multimodal interactions. As the presentation's projection and the camera might not originate from the same source, a calibration wizard (Figure 9.1) is built into the tool for mapping the camera's field of view to the projection area. This is necessary to accurately mapping a laser point or gesture to a specific area in the presentation content. After the tool has been calibrated the following functionality is offered.

Navigation: By using a laser pointer, the presenter is able to navigate the presentation from any position in the room. By pointing the laser at the projection area and performing a specific gesture, a corresponding navigational action is executed. For instance, as shown in Figure 9.2 it is possible to go forwards or backwards in the navigational path by swiping left or right accordingly. However, in order to support other paradigms than simple slide sequences, additional gestures allow the presentation to zoom out by swiping up, or to zoom in by briefly clicking. The idea of using the laser pointer as an input device is not new [148, 214, 291, 5] but we included the laser pointer as an input modality as it provides a good solution for the issues we try to address in this specific context.

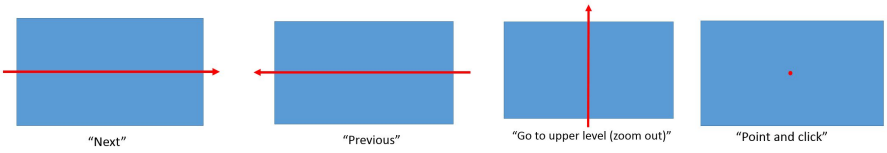


Figure 9.2: Implemented interactions for a laser pointer [39]

Automatic Highlights: As mentioned before, the highlighting of content can be beneficial but the highlights should be relevant and not be overused. As a next feature the presenter is able to predefine areas of the content that should be highlighted at the right time. During the presentation, data from the colour camera, depth camera and microphone are combined to determine if the highlight should be triggered. More concretely, if the presenter is pointing to specific content (with a laser pointer, stick or by hand) and if the presenter's narrative matches the keywords in the content, the highlight is automatically shown.

Touch-based Interaction with the Projection Surface: We have previously discussed the affordances of using touch-based input for this use case. However, related hardware such as a smartboard is not always present and can be expensive. Our solution uses the Kinect's depth camera to make any projection surface touch-enabled. If the depth data shows that the presenter is touching the projection surface with their fingers or with a pen-like object, this interaction is relayed to MindXpres as a touch event, allowing the presenter to click on content or draw on top of the presenta-

tion. The combination of laser pointing and touch, as provided by our system, has been shown to enable more interactive and fluid presentations in Wizard-of-Oz user studies [47].

Spontaneous Content Creation: As a final feature we aim to support spontaneous content creation, similar to how a traditional whiteboard can be used to freely make additional drawings to help explain unforeseen questions. Two additional hand gestures are provided to invoke this functionality. Waving the hand side to side (Figure 9.3a) will bring up an empty canvas which the presenter can use for free-style drawing using previously discussed modalities. Waving the hand top to bottom (Figure 9.3b) will clear the current canvas, and also works as the gesture to remove annotations made on top of presentation content.

(a) Bring up a blank canvas (b) Removing annotations

Figure 9.3: Using hand gestures to interact with the presentation [39]

9.2 TabletPoint: A Pen and Tablet-based Presenter Tool

As a next solution to support the presenter during the presentation, we investigated the use of a pen and tablet-based presenter view. This solution was developed in collaboration with Chunxiang Yan, as part of the Master's thesis "*TabletPoint: A Pen and Tablet-based Presentation Solution*" [299]. We refer to the original thesis for details, but a summary of the research is provided to motivate the purpose of the presented tool. In contrast to existing work, TabletPoint not only provides improved mobility, overview and navigation, but also addresses some shortcomings related to annotations.

The goal of the plug-in was to provide a portable presenter view that combines the affordances of pen and paper with the mobility and interactivity of a tablet. PaperPoint [251] is a related tool that uses physical paper and digital pens to offer the affordances of pen and paper for interacting with presentations. However, a limitation of physical paper is that it can only be used as an input modality and it cannot be updated to provide visual feedback. Given the interactive and dynamic aspects of a

MindXpres presentation, visualising changes becomes especially important. For this reason our solution is based on the Galaxy Note, a portable tablet with a special stylus for precise pen-based interactions. As a starting point, TabletPoint offers a presenter view (similar to the PowerPoint presenter view) on the portable device. The tablet provides an overview of the content and enables non-linear navigation directly from the tablet, two features motivated by studies such as [219]. Note that Microsoft already allows presenters to use mobile devices as a remote via Office Remote[1], so the idea is not new. However, functionality is limited to basic navigation and our goal was to take this much further. As a next step, we investigated the benefits of digital ink in education and applied our findings to the tool. Related work on digital ink lectures [3, 12] as well as on the use of tablets for this purpose [9, 297] highlight the benefits of providing such tools. In previous sections we have also mentioned how touch-based interfaces are sometimes preferred in lectures as users find the keyboard and mouse difficult to operate [108]. Most interesting are the studies on *how* presenters use digital ink, providing us with some particularly interesting findings. A study on digital ink in lecture presentations [12] from 2004 identifies three main purposes for digital ink:

Attentional Ink: Attentional ink makes up more than 50% of all strokes and are made with the goal of drawing attention to a specific part of the content. For instance, the presenter might circle a keyword or draw an arrow to draw the attention to content that is important to the narrative at that time, similar to how hand gestures are sometimes used. However, the study points out that ink is often persistent while the importance of the highlight is ephemeral and losing meaning as the narrative continues.

Diagrammatic Ink: The second largest use of digital ink is for drawing diagrams during the presentation. A follow-up study on diagrammatic ink in lectures [11] further provides some interesting insights. First of all, diagrams are often composed in different phases that match the presenter's narrative. In other words, the presenter often starts with a simple drawing and extends the drawing in different phases while explaining. However, when the final diagram is revisited later the temporal information and context is lost and diagrams become difficult to interpret. Second, the study also states that presenters often make use of the same base diagram. For instance, in the case of charts the same axes are redrawn for each chart, or in the case of a database course a table with fixed columns might be drawn again and again.

Textual Ink: As a third category, the study shows that writing full text is the least common use of digital ink and is was also investigated in more detail in a follow-up study [10].

[1] `http://research.microsoft.com/en-us/projects/officeremote/`

Our tablet-based tool allows presenters to make annotations with their fingers or with a pen-like stylus. Basic annotation functionality is provided such as different colours and an eraser. Figure 9.4 shows a screenshot of the tablet, showing how the presenter can navigate and annotate slides.

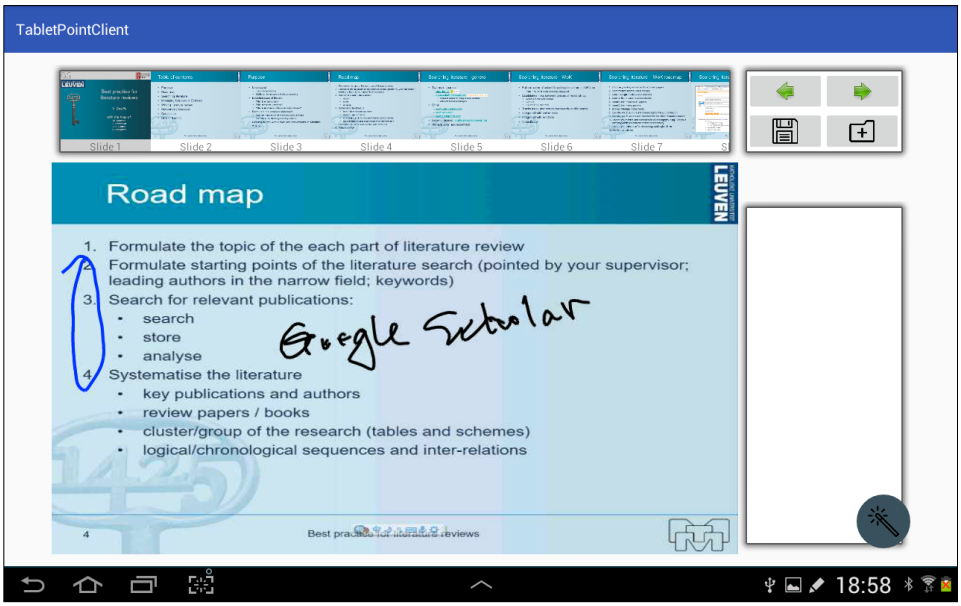


Figure 9.4: Tablet-based interface for navigating and annotating slides [299]

However, the findings presented earlier have resulted in some additional functionality meant to support the presenter as well as the transfer of knowledge. First of all, the tool offers a special type of "ink" for attentional marks. These strokes can be used to circle content or draw arrows, but automatically fade out and disappear after a preconfigured time. This way, these attentional marks serve their purpose of temporarily drawing attention to content but do not clutter up the presentation when the narrative moves on. Next, the tool allows the recording and playback of annotations. It allows students to play back attentional marks later, but more importantly it allows the playback of the incremental drawing of diagrams, as simply showing the final diagram may be difficult to process [11]. To better support the creation of diagrams, we also provide tools for quickly drawing common shapes such as rectangles, circles and lines. Based on the finding that a presenter's diagrams sometimes share a common base, our tool allows the creation of reusable "stamps". These stamps are basically predefined annotations that can be placed anywhere in the presentation and should be used to predefine frequently used diagram bases. As an example, Figure 9.5 shows how the presenter predefined a small grid that is reused multiple times as a template for drawing some Chinese characters. This functionality

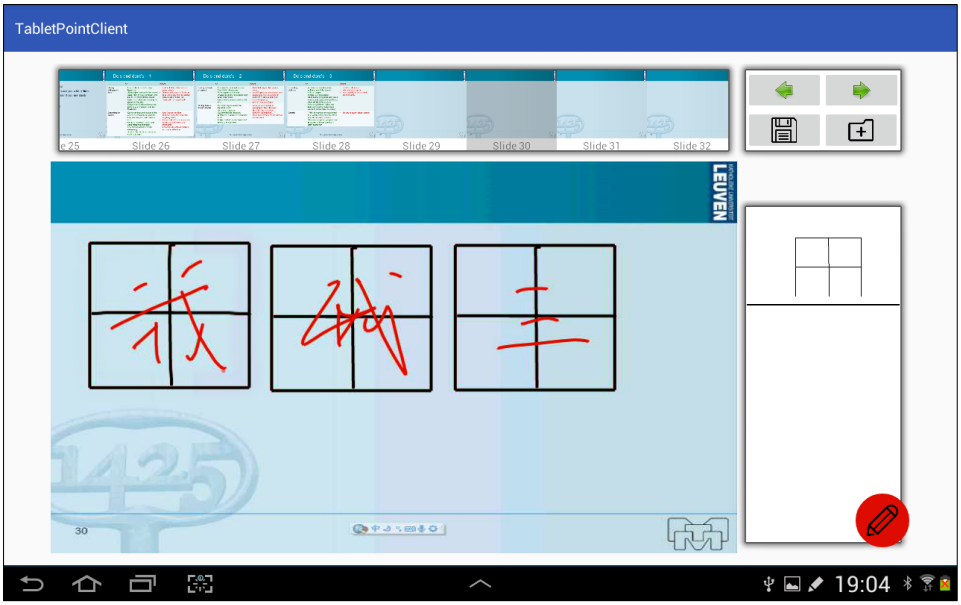


Figure 9.5: Reusing a grid stamp as a visual guide for different Chinese characters [299]

also helps teacher to prepare and plan some of the annotations in order to save time during the lecture, a feature appreciated by teachers [168].

We performed a small evaluation of the system's usability using the System Usability Scale (SUS). The SUS questionnaire as well as some details on the interpretation of the resulting scores is provided in Appendix C.1. For this evaluation we used 5 participants, all students in the field of engineering. To start, we provided a demo of the tool and showcased two different scenarios. The participants were then asked to reproduce these scenarios on their own using our solution, and were also given some time to freely operate the tool. Participants were then asked to fill in the SUS questionnaire, as well as one open question allowing them to share any thoughts or concerns they might have. The results of the questionnaire are shown in Figure 9.6 and result in a mean SUS score of 75. Note that for odd questions a higher score is better and for even questions a lower score is better. Using Bangor's adjective rating [21], this score places the usability of the tool between "Good" and "Excellent".

The scores for Q8[1] have the most variance but can be clarified by the responses obtained from the open feedback question. Some participants found the system cumbersome because the tablet used for the experiment was rather large and heavy, and needs to be held in one hand if the other hand is to be used to draw on the

[1]Q8: "I found the system very cumbersome to use"

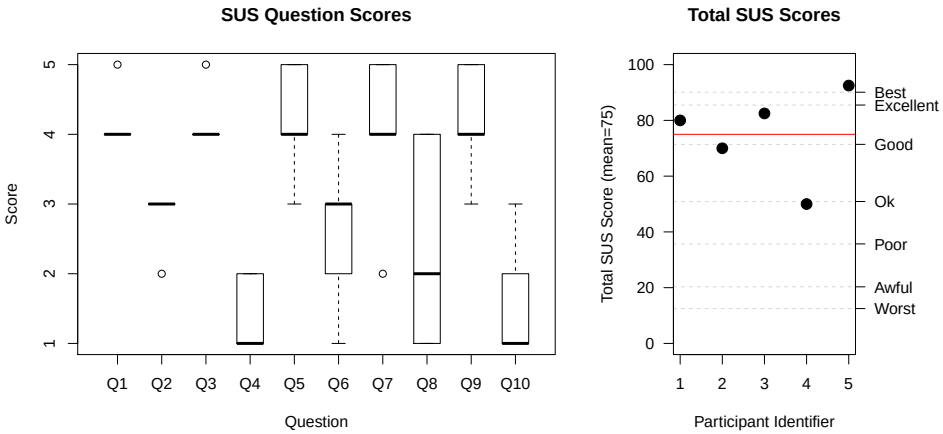


Figure 9.6: Results of the System Usability Scale questionnaire

tablet with a stylus. The remaining open feedback mostly concerned the lag between interactions on the tablet and when these actions become visible to the audience in the presentation's projection. For instance, strokes made with the stylus are instantly visible on the tablet but can take up to a few seconds to appear on the projected presentation content. We see this as a valid issue for this use case. However, the evaluation was performed on a prototype and we believe the delay can be made unnoticeable in future iterations. Finally, one participant also brought up the idea of adding a voice interface for controlling PowerPoint. For instance, the system could react to voice commands such as "Next Slide", "Previous Slide" and "Go to Slide 5" by using the tablet's microphone for input.

9.3 Presentation Feedback System

It is said that no one is born as a bad or a good presenter, and presentation skills, like many other skills, improve with practice and experience [161]. We present a solution for providing a presenter with feedback on their presentation skills, allowing them to improve. This solution was developed in collaboration with Mohamed Zahir, as part of the Master's thesis "*An Extensible Platform for Real-Time Feedback in Presentation Training*" [304]. We refer to the original thesis for details, but a summary of the research is provided to motivate the purpose of the presented tool.

When examining the literature on presentation skills and public speaking in general, the criteria for good oral communication are based on verbal as well as non-verbal communication aspects. Verbal aspects that result in better communication [261, 232]

are related to speech and to the voice, and include factors such as filler words, speed, pitch and volume [242, 206, 161]. Non-verbal aspects relate to body language and include for instance the use of gestures, eye contact with the audience and posture [156, 206]. We refer to the original thesis for a closer examination of the factors that influence how speakers are perceived. Instead, we focus on describing a tool that can automatically provide feedback on these aspects of oral communication. The idea of a tool for automatic presentation feedback is not new. Presentation Sensei [161] is one of the earlier systems in this category and uses a camera and microphone to monitor the presenter and to provide feedback. The tool focusses on monitoring the speech rate, speech pitch, use of fillers (e.g. "uhm" or "er"), the posture relative to the audience and the timing of the presentation. During the presentation small icons are shown as feedback (e.g. if the presenter speaks too fast) and after the presentation a report can be viewed with the measurements of these focus points over time. As one of the first in this category of tools, Presentation Sensei was an interesting novel contribution. However, the system suffers from some limitations. The system is very rigid and can only be applied in specific scenarios. For instance, the authors are Japanese and the rules for determining whether a presenter speaks too slow or too fast are based on the average speaking speed of the Japanese language (7.6 mora/sec[1]). The thresholds for all the other monitored criteria are similarly hard-coded and the tool cannot be extended to support additional presentation aspects that should be monitored. Given that the system was developed in 2007, it also suffers from some technical limitations and the automatic detection of the monitored presentation aspects are far from accurate when comparing results with a human evaluator.

We argue the need for a more modular system for presentation feedback. First of all, it should be possible to load different rules to take different scenarios and use cases into account. Rules might not only be different depending on the language but also based on the culture [32, 261, 232] or presentation setting. For instance, optimal talking speed and pitch are different for each language, eye contact might be interpreted differently depending on the culture. Different types of presentations such as lectures, sales pitches or TED talks also require different presentation styles which might have different requirements in terms of speech rate or body posture. Ideally, it should also be possible to extend the hardware used by the system and to extend the presentation aspects that can be monitored by the tool for feedback. Finally, whereas Presentation Sensei and similar systems focus on providing feedback on the presenter's performance, we also aim to include the presentation content itself for feedback.

[1]Mora is a unit of sound used in phonology that determines syllable weight. A syllable can be composed of multiple mora.

	Supports Presentation Training	Supports Live Presentations	Real-time Feedback	Post-presentation Feedback	Extensible Monitoring	Extensible Input Modalities	Extensible Output Modalities	Extensible Thresholds
Presentation Sensei [161]	✔	✔	✔	✔				
Presentation Trainer [242]	✔		✔			✔		✔
RHEMA [268]	✔	✔	✔					
PreSense [170]	✔			✔				
Cicero [25]	✔			✔				
Logue [61]	✔	✔	✔	✔				
Kopf et al. [156]	✔			✔				
Nguyen et al. [205, 204]	✔	✔	✔					
Gan et al. [96]		✔		✔		✔		
Wortwein et al. [298]	✔			✔				

Table 9.1: Related solutions for presentation feedback

In the original thesis we discuss the related work in detail, but to motivate our new feedback system we summarise the characteristics of existing presentation feedback systems in Table 9.1.

As shown in Table 9.1, different tools have different characteristics. Some focus on pre-presentation training while other focus on providing feedback during real presentations. Some tools provide feedback in real-time during the presentation while others create reports that can be analysed after the presentation. However, our biggest concern is that like many presentation tools, also feedback tools make certain assumptions about a presentation that cannot be changed. As previously mentioned, it is very difficult if not impossible to define a set of presentation rules that work for all languages, cultures and scenarios. However, the presented tools make such assumptions and cannot easily be changed. Furthermore, we see a lot of additional factors that would be interesting to monitor in such a feedback system, but existing systems can also not be extended in terms of the input hardware or the aspects that are monitored.

To address existing shortcomings and to meet our previously mentioned requirements, we designed and implemented a more flexible framework for presentation feedback. An overview of the architecture is shown in Figure 9.7. First of all, the framework makes

it easy to plug in a variety of hardware for capturing information from the physical world (e.g. microphones, colour cameras or depth cameras such as the Kinect). These devices provide data streams that can then be interpreted by detection plug-ins. The streams from the input modalities can be efficiently used by multiple detection plug-ins without overhead. For instance, the audio stream provided by the microphone modality can be used by a detection plug-in for filler words, but a detection plug-in that monitors speech volume would also operate on the same stream without overhead. Similarly, a video stream can be used by a variety of plug-ins to determine posture or orientation. In order to prevent the shortcomings of existing tools and to avoid hardcoding thresholds in the decision making process (e.g. "the presenter should speak between X and Y words per minute"), we make use of a rule engine. Rules for the decision making process are loaded into the rule engine and can easily be replaced to accommodate different languages, cultures and presentation scenarios. When detection plug-ins derive certain information from the input modalities these are published as facts into the rule engine (e.g. "the presenter is currently speaking at 22 words per minute"). By using an existing efficient rule matching engine, rules are automatically evaluated with the currently available facts and if a match is made, the application controller layer is notified and passed the relevant metadata. The logic in the controller can then relay feedback to the presenter via one of the many possible output modalities. This might for instance be in the form of a discrete pop-up but can also be a bracelet that provides haptic feedback as investigated by Presentation Trainer [242]. To the best of our knowledge, we were the first to use a rule engine for interchangeable rules and efficient rule matching, but a similar approach was independently applied by Presentation Trainer in the same year. The data streams as well as the facts derived by the detection plug-ins are all logged in order to also generate reports for later analysis. By also logging the data streams, we can for example show video or audio fragments of the moments where issues were detected. Finally, we would like to point out that the presentation tool itself can also serve as an input modality. This not only allows us to monitor, for instance, the time spent on each slide, but also lets detection plug-ins analyse the content itself. Such a plug-in might for instance analyse each slide and use interchangeable sets of content guidelines for different use cases to avoid major issues such as having too much text on a slide or using colour combinations that might negatively affect viewers. Some guidelines are also universally applicable, such as guidelines for making optimal use of the capabilities of the human mind [98, 158].

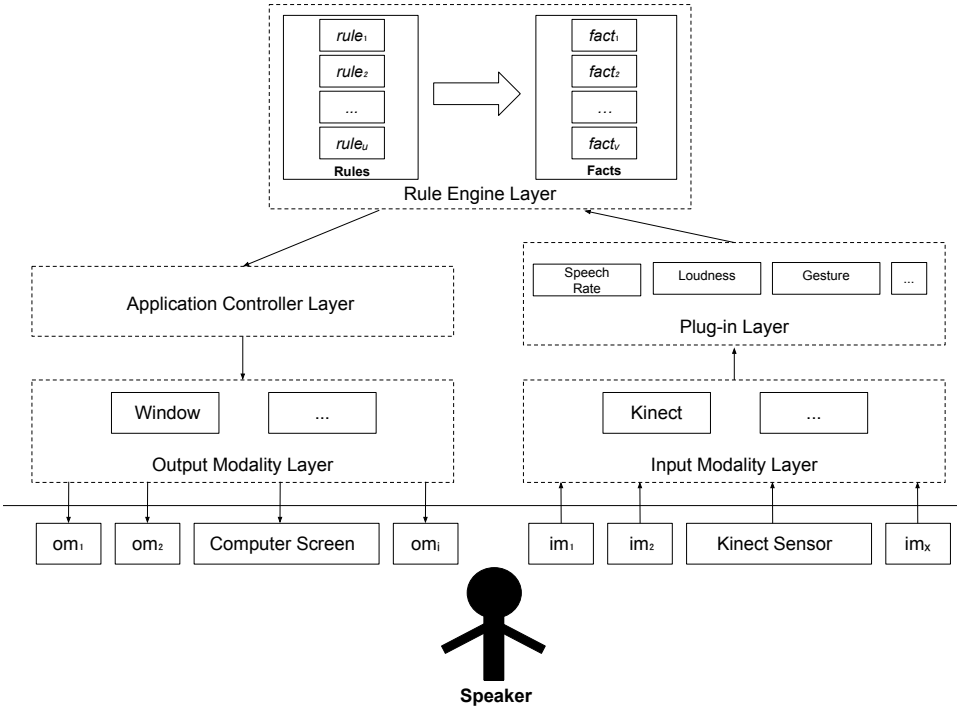


Figure 9.7: The architecture of our modular presentation feedback system [304]

9.4 Augmented Reality Presenter View

As a follow-up on the earlier tablet-based presenter view and the feedback system, we investigated the use of augmented reality for a new presenter view. This solution was developed in collaboration with Kwinten Pardon, as part of the Master's thesis "*An Extensible Presentation Framework for Real-time Data Acquisition, Interaction and Augmented Reality Visualisations*" [210]. We refer to the original thesis for details, but a summary of the research is provided to motivate the purpose of the presented tool.

One of the major critiques on the tablet-based presenter tool detailed in Section 9.2 was that the tablet was quite heavy, requires two hands to operate and was cumbersome to use in general. Furthermore, our extensible system for providing live feedback during a presentation (Section 9.3) is very powerful but discretely delivering a wide range of feedback to the presenter without disrupting the narrative remains a challenge. As an alternative to pop-up warnings, Presentation Trainer [242] uses a vibrating bracelet for haptic feedback which is discrete and relatively non-disruptive, but limited in the amount of messages that it can relay to the presenter. Additionally,

the modalities for interacting with a presentation as a presenter (e.g. mouse, gestures or mobile device) are not very discrete either and might disrupt the focus of the audience. In order to address these shortcomings we investigate the use of an augmented reality presenter view. Although existing hardware such as the Microsoft HoloLens[1] is expensive and bulky, we operate under the assumption that the technology will become more discrete over time and will eventually look like regular glasses or even contact lenses. The biggest advantage of this approach is that it allows the presenter to see information that is invisible to the audience without having to turn away and lose eye contact. This would not only allow real-time presentation feedback to be shown to the presenter, but it could also offer an interactive interface for navigation and interaction with the content. Note that the content shown to the presenter does not have to match what the audience sees, and it could also be used as a presenter view to show notes or even act as a kind of teleprompter [14].

The system was not developed with a fixed set of end-user functionality in mind, but rather as a generic extensible framework that would enable interesting new use cases in the future. The idea is that the framework would facilitate the development of modular "widgets" that can be shown in the presenter's view to compose configurable heads-up displays for presentations. As with a lot of the solutions presented in this dissertation, the framework is extensible on multiple levels. First of all, there is a layer of plug-ins dedicated to capturing data from arbitrary sources, and also sending data back. Feedback from a presentation feedback system or currently relevant content from the presentation tool might be captured, or a navigation event might be sent back to the presentation tool if certain actions are performed. Captured data is published to a local publish-subscribe engine. The UI widgets can subscribe to specific data or events, allowing them to update their visual state accordingly. For instance, if new feedback is published to the publish-subscribe engine, the widget that is responsible for visualising this feedback is notified so that it can show the appropriate warning. Finally, another plug-in layer is responsible for providing different interaction modalities to the presenter. For instance, a plug-in for speech recognition uses the HoloLens's microphone to capture speech commands and to publish these to the publish-subscribe engine. Similarly, another plug-in allows the presenter to discretely use the HoloLens's remote control for a variety of interactions with the UI widgets. The framework was built for the Microsoft HoloLens but also this could be changed in the future. Concretely, the presenter UI was developed as a widget system based on web technologies, making it easier to port the interface to other devices. Figure 9.8 shows the view of a presenter using the system and various widgets can be seen as an overlay of the real world. As a proof-of-concept basic functionality such as a teleprompter, presenter notes, feedback pop-ups and presentation mirroring was

[1] `https://www.microsoft.com/en-us/hololens`

implemented, with support for gestures, the HoloLens remote and speech commands as interaction modalities. However, we see some more interesting use cases for the future. As an example, the colour camera and depth sensors built into the HoloLens could be used to scan faces in the audience and to detect emotional states [20] such as boredom or excitement for feedback. The extensibility of the framework would also support the use of brain-computer interfaces to measure audience engagement, as explored by Hassib et al. [120].

Figure 9.8: A presenter view based on augmented reality [210]

We also performed a small evaluation of the framework. As the goal of the project was to create an extensible framework for this type of use cases, we evaluated the usability of the framework for developers. For this we used the Computer System Usability Questionnaire (CSUQ). The CSUQ questionnaire as well as some details on the interpretation of the resulting scores are provided in Appendix C.2. Given that we were evaluating a software development framework, participants were required to have some development experience. There were six participants (two females, four males), each having 2 to 12 years of development experience. As a first step the participants were shown a short presentation which explains the goals and purpose of the framework. Next, each participant was explained the technical details of the framework (e.g. how the core works and how the various components can be extended). As a next step, the participants were asked to come up with a basic idea for a widget. They were then given the opportunity to explore the framework and investigate how their idea would be implemented. During this phase, the participants were free to ask questions or discuss their ideas with us to gain further insights on how to achieve their goals. Finally, participants were asked to score the 19 statements of the CSUQ questionnaire on a 7-point Likert scale. The questionnaire also allows participants to list some positive as well as some negative aspects of the framework, allowing them to provide open feedback. The mean scores for the questions as well as the total overall and subscale scores are shown in Figures 9.9 and 9.10.

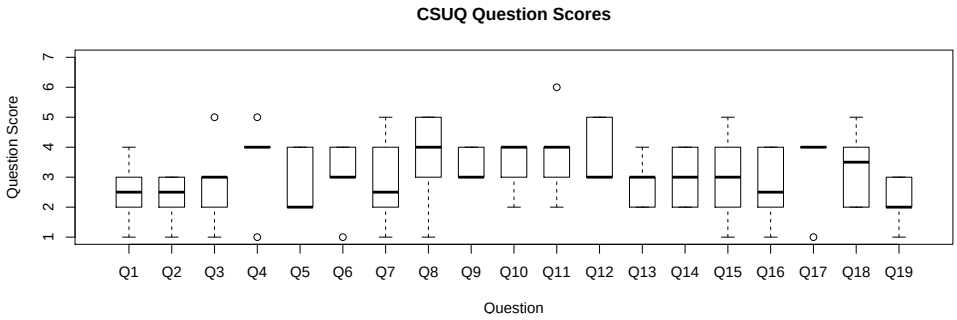


Figure 9.9: CSUQ Scores (lower is better)

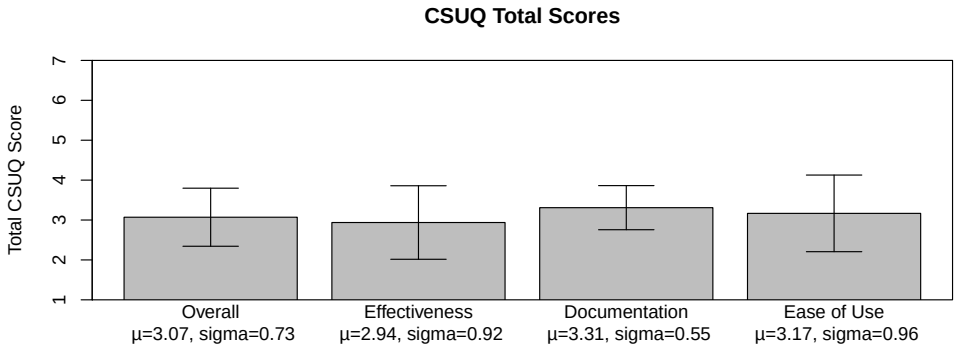


Figure 9.10: Total CSUQ Scores (lower is better)

In the case of the CSUQ scores, lower is better. The overall score shows that the framework receives an overall positive evaluation but also indicates that there is still room for improvement. Particularly the questions related to *information and documentation* show lower scores. However, considering that the framework was a proof of concept, little effort had been made in that area and a lower score was expected. One participant stated that the framework requires a lot of technical knowledge which might cause issues for non-technical users. In terms of the positive and negative aspects participants were asked to identify, most participants praised the modularity of the framework as well as the ease of learning and use of the framework. Negative feedback mentions the lack of documentation and that the framework can only be used by programmers (i.e. there is no easy to use GUI for creating widgets).

9.5 Virtual Presentation Assistant

As a final tool for supporting the presenter during the presentation, we present a virtual assistant for presenters. This solution was developed in collaboration with Kushal Soni, as part of the Master's thesis "*An Extensible Virtual Assistant for the PowerPoint Presentation Tool*" [256]. We refer to the original thesis for details, but a summary of the research is provided to motivate the purpose of the presented tool.

Virtual assistants are being embedded in more and more devices such as smartphones, smartwatches, speakers and cars. Examples of such assistants include Microsoft's Cortana[1], Apple's Siri[2], Amazon's Alexa[3] or the Google Assistant[4]. These assistants provide a human-like interface to a variety of actions such as playing music, querying for specific information or for interacting with other systems such as smart homes. The goal of this tool was to create a virtual assistant specifically for presentations, providing the presenter with quick access to ad-hoc functionality. As will be shown later, such an assistant is particularly useful for dealing with unforeseen circumstances where some relevant content is required but was not, or could not be, embedded in the presentation beforehand.

In contrast to DynamicDuo [275], the goal was not to create a virtual co-presenter that is visible to the audience. Instead, the system would live in the background and would be closer to Siri or Google Assistant. In order to be able to interface the assistant with our other solutions, also here extensibility was a major aspect in the design. An overview of the architecture is shown in Figure 9.11. First of all, we aim to provide a wide range of modalities to interact with the assistant. For this reason a plug-in system is provided for the input modalities allowing for instance speech, gestures or a graphical user interface to be used to interact with the assistant in a way that is appropriate for the situation. Internally an extensible vocabulary is used to map a user's commands to specific action plug-ins. This way, the supported actions can easily be extended together with the speech or text keywords that will trigger the action. As a proof of concept some basic input modalities, vocabulary and action plug-ins were implemented. For instance, a navigation plug-in allows the presenter to ask the assistant to jump to a specific part of the presentation. Other plug-ins allow the assistant to bring up movies, pictures, maps or weather forecasts on command, based on provided user criteria. As part of future work we intend to make better use of the MindXpres content metadata and semantics to, for instance, enable powerful semantic querying in natural language (e.g. "show me the related diagram from last week's presentation").

[1] `https://www.microsoft.com/en-us/cortana`
[2] `https://www.apple.com/siri/`
[3] `https://www.amazon.com/b?node=17934671011`
[4] `https://assistant.google.com`

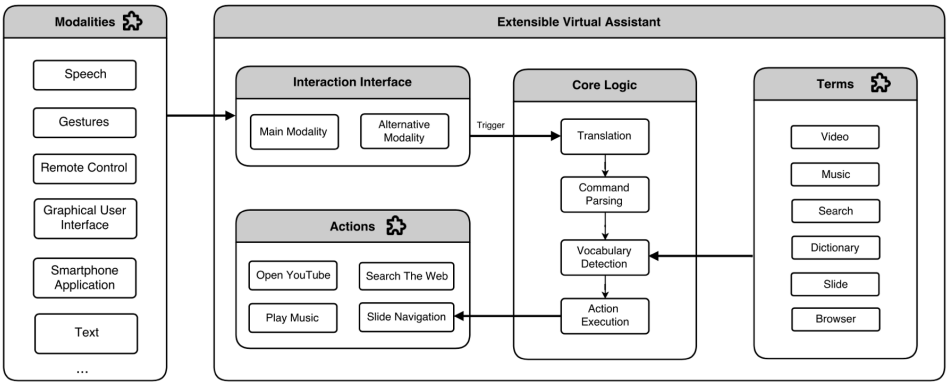


Figure 9.11: The modular architecture of our virtual presentation assistant [256]

We performed a small evaluation of the assistant's usability using the System Usability Scale (SUS). The SUS questionnaire as well as some details on the interpretation of the resulting scores are provided in Appendix C.1. For this evaluation there were 5 participants with varying backgrounds. All participants had at least some basic experience with giving presentations. After a short explanation, each participant was allowed to try out the system by using different modalities and actions. Afterwards they were asked to fill in the SUS questionnaire as well as some open questions where feedback could be provided. Figure 9.12 shows the scores of individual questions as well as an overview of the total SUS scores. Note that for odd questions a higher score is better and for even questions a lower score is better. With an average total SUS score of 82, the usability of the system can be graded between "Good" and "Excellent" using Bangor's adjective rating [21]. Noteworthy are the very good scores for Q10[1] indicating that the system is easy to use. However, the score for Q9[2] indicates that some users are not perfectly confident in using the system. Feedback from the open questions revealed that due to the flexibility of modalities such as speech, it is unclear which terms or keywords should be used to achieve the desired action. If the relevant terms and keywords and not known beforehand, multiple attempts might have to be made which can lead to embarrassing situations in front of an audience. The participants were also asked to mention their favourite modality and all of them preferred speech. One participant specifically stated that using gestures would seem strange in front of an audience. We also observed how accurately the system would react to the user's intentions. Although speech is the preferred modality, it is also the least accurate. We identified multiple sources of inaccuracy. First of all, the speech recognition engine we used (Google's Speech API[3]) often misunderstood the

[1] Q10: "I needed to learn a lot of things before I could get going with this system."
[2] Q9: "I felt very confident using the system."
[3] `https://cloud.google.com/speech/`

user. We believe that this is partially because most participants were not native English speakers. Second, as mentioned before users are sometimes uncertain how to formulate their request, or formulate their request in an ambiguous way. For instance, the query "show me a rabbit" did not trigger an action because the user did not use keywords such as "image" or "picture". Even if the system had recognised the query as a command, the query is still ambiguous because it is unclear if the user wanted to see a video or an image, so the system would not know which action to invoke. These issues are related to the fact that our implementation is a prototype with a basic custom intent interpreter. Applying existing and more robust solutions for virtual assistants and natural language processing can address the mentioned issues. For instance, by using a framework for natural language processing the tool might better understand the user intent when synonyms are used or an existing API (e.g. LUIS[1] or Wit.ai[2]) can be used to better extract the user's intent from a query. Given that the focus of the experiment was to evaluate the idea of a virtual assistant in presentations (and not our very basic intent interpreter prototype), the scores and feedback indicate that all in all, the participants are enthusiastic about the system.

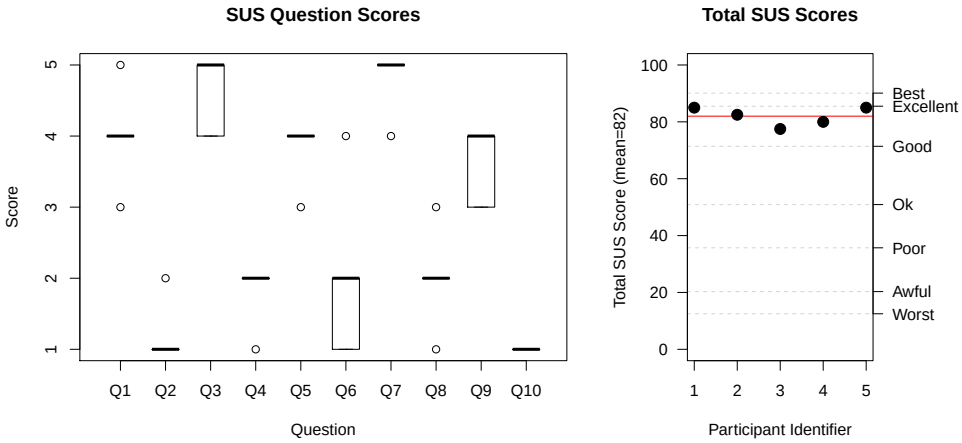


Figure 9.12: Results of the System Usability Scale questionnaire

[1] https://www.luis.ai/home
[2] https://wit.ai

Chapter 10

Solutions for Authoring, Reuse and Collaboration

As a final category of solutions, we present some solutions related to authoring, reuse and collaboration in this chapter. In Chapters 2 and 3 we have shown how presentation software could better support the user to create good content with less effort. The MindXpres platform itself was already a first step towards addressing these shortcomings. The data-driven approach of the presentation engine's visualisation plug-ins allow presenters to focus on the content and plug-ins can automatically take care of the layout and visual details if desired. Furthermore, the foundation for improved content reuse and collaboration is provided by the content model and resulting information system. In this chapter we present some additional solutions that further aid the presenter during the authoring process.

10.1 Guided Data Visualisation

A typical presentation authoring tool has many different buttons and toolbars and a lot is dedicated to aesthetic details such as colours and fonts. Similarly, these tools offer the ability to create charts and a lot of the provided functionality revolves around styling these charts. However, as brought up in Chapter 2, it is in the hands of the user to apply these correctly and users receive little to no support in creating *good* content. In this section we focus on a tool for helping presenters create effective charts and graphs. The tool follows the wizard design pattern [273] and guides the user towards an effective visualisation for the provided data and message that needs to be conveyed. This data visualisation wizard for MindXpres was created in collaboration with Jasper

Debie, as part of the Master's thesis "*Guided Data Visualisation for the MindXpres Presentation Tool*" [66]. We refer to the original thesis for details, but a summary of the research is provided to motivate the purpose of the presented solution.

In the last ten years the area of information visualisation has witnessed a high increase in its popularity and is becoming an increasingly necessary tool to think, explore and comprehend complex data [239]. Visualisation not only makes the patterns, outliers and trends visible and understandable, but also helps us think by using the close relationship between vision and cognition [279]. However, it is in the hands of the user to design the visualisation so that it is easily understandable and that it conveys the intended message effectively. After studying related literature our initial idea of automatically generating visualisations for MindXpres turned out to be infeasible. The reason is that the visualisation depends on the message the presenter wants to convey. For instance, a comparison between data points requires a very different visualisation than showing the evolution of the data points over time. However, this message cannot be automatically be derived from the data alone and therefore the process will always require some input from the user. Spence [258] similarly describes how a combination of technical tools and human insight is required for creating good data visualisations. For this reason, our solution is a tool that automates as much as possible but still requires a minimum of input from the user.

In order to guide the user to a good data visualisation the tool is based on existing guidelines from experts and researchers in the field of information visualisation. We compiled guidelines from work that, amongst others, include William Cleveland [55, 54], Edward Tufte [277], Stephen Few [88, 90] and Michael Friendly [93]. For instance, Cleveland [54] provides insight into which types of visualisations suit specific structures in the data (e.g. univariate, bivariate, trivariate or hypervariate data). Few [90] details eight types of quantitative messages that users might want to convey (time series, rankings, part-to-whole, deviation, distribution, correlation, geospatial messages and nominal comparison) and maps them to specific visualisation types. As vision is one of the senses that is the closest connected with cognition [89] we also gathered guidelines for optimising visual perception [293]. This includes guidelines on the use of space, colour, backgrounds, ticks, labels and legends from a variety of sources. The goal is not only to enlighten and inform the reader [155] but we also took guidelines into account for avoiding misleading and deceptive visualisations [128, 55, 277]. We only mention the most important sources and guideline categories here to illustrate the general idea, but we refer to the original thesis for a deeper analysis of the related work on charts of graphs.

The general idea behind the guided visualisation tool is illustrated in Figure 10.1. As a first step, raw data in the JSON format is loaded into the system. The tool then attempts to derive as much as possible from the data. For instance, an estimate is

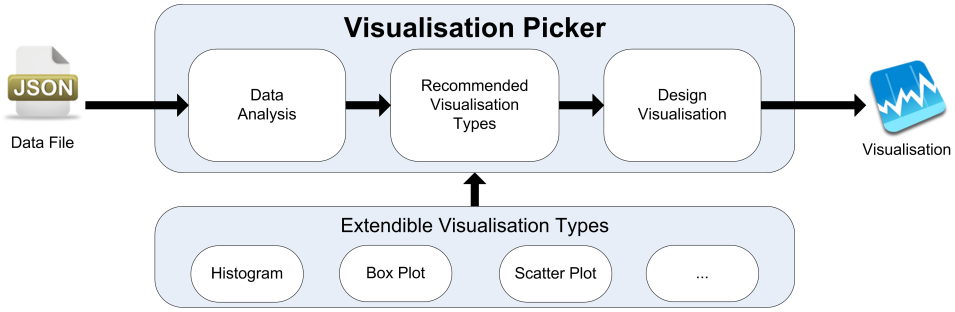


Figure 10.1: The process of recommending a suitable visualisation for raw data [66]

made for the amount of variables and the type of the variable is derived (e.g. numeric or text). The findings of the analysis process are shown to user (Figure 10.2) allowing them to verify the findings and correct them if needed. Next, the user is also required to provide some additional information such as the message that is important for this specific visualisation (e.g. comparison or evolution over time). The types of visualisations (e.g. bar chart, pie chart or scatter plot) are extensible and implemented as plug-ins. Each plug-in also exports a set of criteria that state in which situations the plug-in should be considered as an option. For instance, a plug-in might state that it is relevant if the data is bivariate and if one of the variables is numeric. Each criterion should also include a weight allowing the plug-in to state to what degree it is relevant if the criterion is met. This not only allows developers to easily add new visualisations and rules but it also provides a solution for guidelines that contradict each other as they can live side by side in different plug-ins.

After the user's data has been analysed and missing information has been provided, the system matches this metadata with the weighted criteria of all the plug-ins to create a ranking. This way, the visualisations that the system deems as most effective for the data and message are shown higher in the ranking but the user is still able to choose from all the recommendations. One shortcoming of the system is that we trust plug-ins to provide honest matching criteria but the recommendation system can easily be manipulated by creating a plug-in that states that it is the optimal choice for all possible data structures, data types or messages. Finally, when the user selects one of the recommended visualisations, matching code is generated in the MindXpres format for usage in a MindXpres presentation.

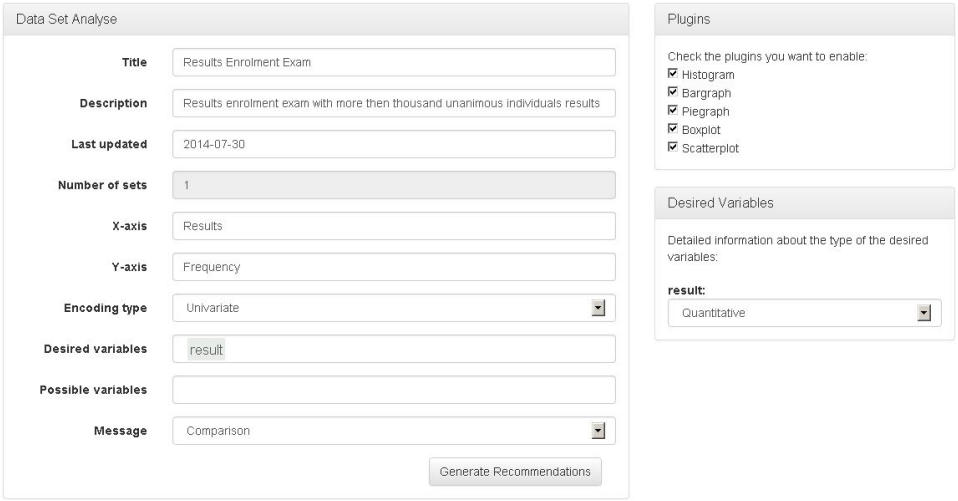


Figure 10.2: The interface of the guided visualisation tool showing derived as well as user-provided data characteristics [66]

10.2 PowerPoint Import and Automatic Content Layout

The goal of this next plug-in is to help presenters with reusing content created in PowerPoint and to automatically apply an appropriate layout for imported content. This plug-in solution for importing legacy content and automatic layouts was created in collaboration with Joris Vandermeersch, as part of the Master's thesis "*Content Migration and Layout for the MindXpres Presentation Tool*" [283]. We refer to the original thesis for details, but a summary of the research is provided to motivate the purpose of the presented solution.

With PowerPoint being the dominant solution for presentations knowledge workers and businesses have produced large quantities of PowerPoint content in the last decades. We have previously discussed the importance of reusing content and outlined reuse and collaboration within the MindXpres-specific information system presented in Chapter 5. However, given the large quantity of existing PowerPoint content we also deem it important to provide a way to bring PowerPoint content to the MindXpres platform. In future stages of the MindXpres project users might also be more willing to try MindXpres if they do not have to recreate all their content from scratch. As a first goal, we intended to create a tool for porting existing PowerPoint content to the MindXpres platform. However, importing PowerPoint content introduces a new interesting problem. MindXpres takes a data-driven approach and when content is defined in the LaTeX-like declarative language its layout and visualisation is handled mostly automatically. On the other hand, PowerPoint content has fixed dimensions

and positions that might not translate well to the presentation styles used in a MindXpres presentation. As a secondary goal we aimed to improve MindXpres with truly automatic layouts instead of the existing template-based approach. This way, PowerPoint content can be imported and automatically laid out in a way that matches the selected MindXpres presentation style. However, presenters might not want their original PowerPoint layouts to radically change after importing to MindXpres. For this reason we designed the algorithm to optionally accept some layout constraints. The import tool can derive layout constraints from the original PowerPoint content such as the relative positioning of the elements or relative sizes. If provided, the automatic layout algorithm takes these constraints into account to generate a layout that is similar, but that still blends seamlessly into any presentation style and makes optimal use of the available space.

The thesis investigated various methods and approaches for automatically laying out content [131] with MindXpres in mind as the area of application. This includes for instance graph layout algorithms [26], bin packing algorithms [124], responsive design, grid systems [198, 130] and layout features of software UI frameworks. More promising is the use of constraint-based methods and constraint resolvers [305, 38, 110, 127, 151, 126, 295, 200] in the context of layouts and graphical interfaces. Another interesting approach is the use of machine learning to automatically generate constraints for the previously mentioned constraint solvers [310].

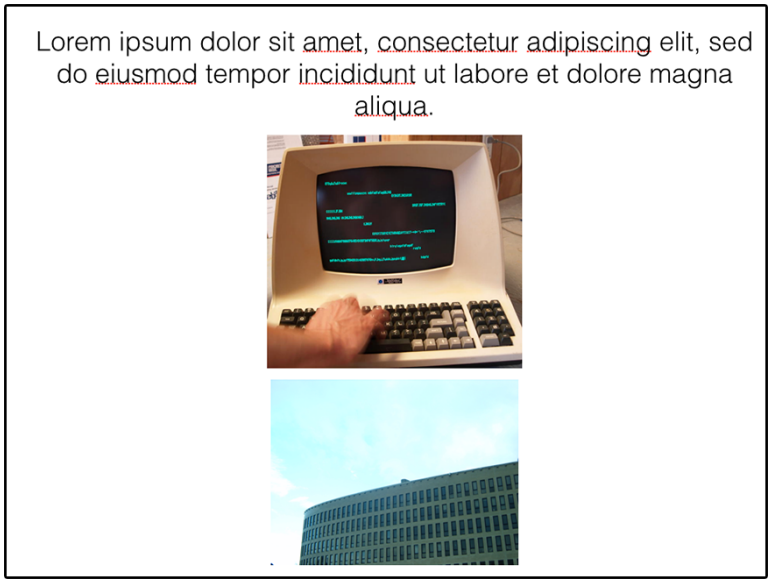


Figure 10.3: PowerPoint content as created in the PowerPoint editor [283]

It should be noted that the main contribution of the thesis was a study of the literature and an exploration of the approaches that might apply to MindXpres. A basic proof of concept was built but it is not based on any of the more advanced systems that were explored. The import tool uses the Apache POI[1] library for parsing PowerPoint presentation documents and is able to extract images, text, bullets and animations. As mentioned before the system can optionally also derive basic constraints such as relative positioning of elements or relative sizes. These constraints can later be used by the automatic layout algorithm to reproduce a layout that is similar but adapted for MindXpres. The automatic layout feature is implemented as a container plug-in for MindXpres (see Section 6.2.2) that lays out content within. The layout algorithm applies a mix of existing guidelines extracted from related work (e.g. golden ratio, symmetry or centering [237]), predefined constraints and constraints derived from the original PowerPoint content if provided. Figure 10.3 shows some PowerPoint content which is then imported and automatically laid out in Figure 10.4. For the sake of completeness we would like to add that Microsoft Research later published a paper on the subject of automatic layouts in general (not for presentations per se) [300]. However, the paper was published after the completion of this import-related project and as such we did not take these additional findings into account.

Figure 10.4: Automatically applying a layout to imported PowerPoint content [283]

[1] https://poi.apache.org

10.3 Graphical Authoring Tools

Early on in the research project we already investigated graphical authoring tools to supplement the XML-based MindXpres authoring language. As a first step in this initial investigation we compared the functionality of existing authoring tools such as PowerPoint, Keynote and Prezi. The shortcomings associated with the existing authoring tools have already been discussed multiple times throughout this dissertation. These include, among other issues, a lack of extensibility, a limited number of supported content types, the large amount of time and effort that has to be spent on visual details and the reuse of content. On the one hand the MindXpres platform addresses these issues in the previously presented fundamental and technical foundations. On the other hand the design choices in the foundations require us to also reinvent the graphical authoring process. A good example is that existing authoring tools only support a limited set of content types. This allows them to hardcode editing support for these types and to provide content-specific editing methods for each content type. Given the limited set of content types, it is also clear which content types are compatible with each other and how they might interact. In contrast, in the MindXpres platform almost everything is a plug-in and there is no limit to the supported content types. It might even be possible that there are multiple plug-ins available for the same content type. As a result, it is a major challenge to create a user-friendly editor that takes every potential plug-in into account. Propagating technical abstractions to the end-user would cause usability issues. However, the alternative of making assumptions about the supported content types in the editor might introduce some restrictions for the user, which is exactly what we have been trying to avoid throughout this dissertation. One of the outcomes of this initial investigation is that the creation of a graphical editor is more challenging than expected. A significant research effort is required to bring the more complex functionality of the MindXpres platform to end-users. This includes the development of architectural and UI abstractions that can span all potential content plug-ins and also requires the investigation of appropriate metaphors to bring their complex functionality to end-users. Given the limited time available we focussed on the conceptual and technical foundations, and although we could not fully address the challenges of creating a graphical editor for the MindXpres platform, some basic editors have been created to demonstrate how some functionality might be offered to end-users.

10.3.1 Graphical Template Authoring Tool

Given the challenges of creating a complete authoring tool for the MindXpres platform, we limited the scope of the problem and first focussed on creating a graphical tool for creating themes and templates for MindXpres. It should be noted that this initial investigation was performed before most of the other MindXpres components

were created. This authoring solution for creating MindXpres templates and themes was created in collaboration with Jeroen Heymans, as part of the Master's thesis "*A WYSIWYG Template Authoring Solution for the MindXpres Presentation Tool*" [121]. We refer to the original thesis for details, but a summary of the research is provided to motivate the purpose of the presented solution.

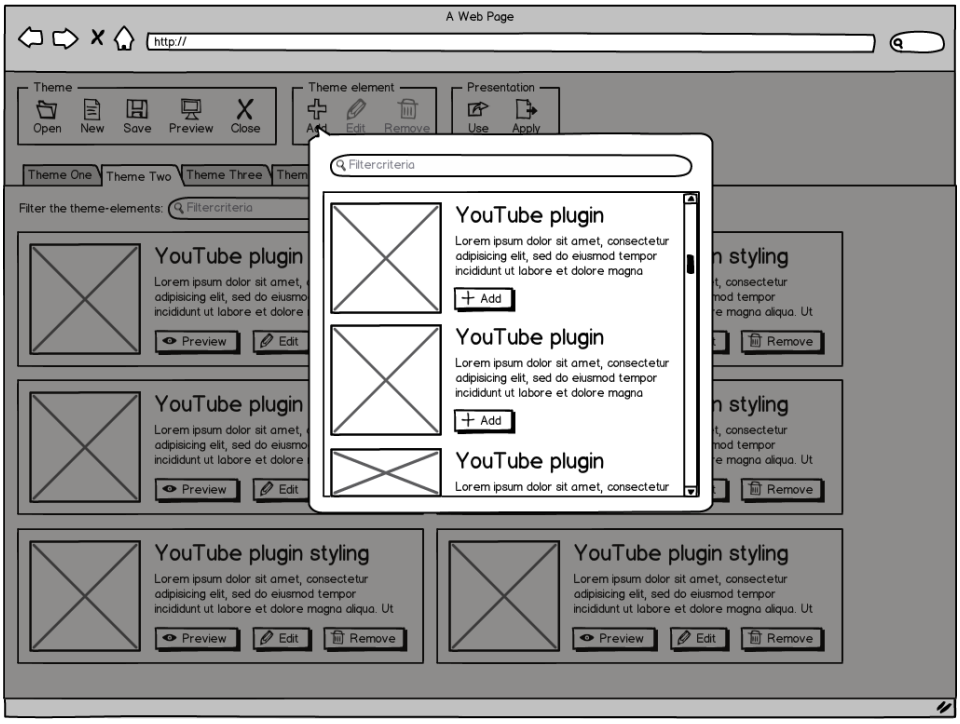


Figure 10.5: One of the many mock-ups of the envisioned theme authoring process [121]

After a comparison of existing authoring tools, we focussed on investigating how the design of the MindXpres platform would affect the development of themes and templates. The MindXpres presentation engine can handle layouts automatically, but it also supports themes so that users can change aspects such as backgrounds, fonts and colours if desired. These themes can override the default settings provided by the various content plug-ins. Themes can override settings presentation-wide (e.g. all text), for specific plug-ins (e.g. only titles of slides) or for specific instances of content (e.g. one specific piece of text). We developed an appropriate software architecture that is compatible with the presentation engine's plug-ins, we developed mock-ups of the authoring process. One of these mock-ups is shown in Figure 10.5 and shows how visual details can be changed for individual plug-ins.

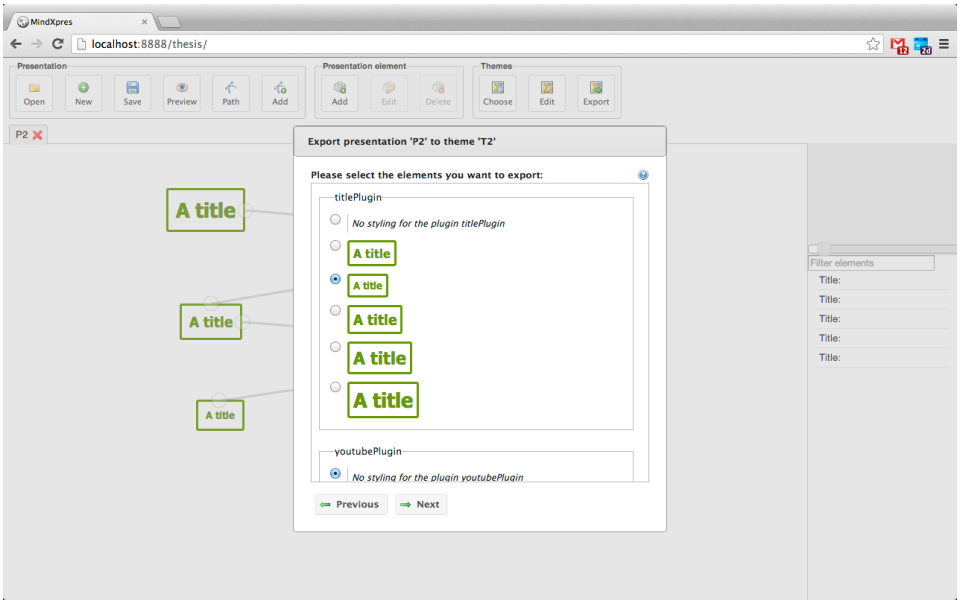


Figure 10.6: A graphical tool for defining themes and templates for MindXpres presentations [121]

As a next step we also created a proof-of-concept implemented. The tool was implemented using web technologies such as HTML5, CSS3 and JavaScript. Themes are exported to CSS, on which MindXpres themes are based (see Chapter 6). Figure 10.6 shows how a user can use the editor to style a canvas-based presentation. Note that the editor does not make any assumptions about the available plug-ins or presentation style. This way, the theme editor is not limited to canvas-based presentations but it works for styling any of the available container or content plug-ins. Once themes are created with this editor, they can be reused by other users in the MindXpres authoring language. In other words, this tool is only needed to create custom themes and once created, they can be reused by others without needing the theme authoring tool.

10.3.2 Graphical Presentation Authoring Tool

As mentioned at the start of this section, this dissertation focusses mainly on the conceptual and technical foundations for improved presentation tools, and a proper graphical authoring tool remains a major challenge. Nevertheless, we created a basic graphical presentation editor to demonstrate how some of the new low-level functionality might be offered to users. However, it should be noted that the graphical editor is relatively basic and it was not feasible to offer all the functionality of the information system and to investigate the appropriate metaphors and interactions for

end-users within the available time. The chosen approach is one of many possibilities and our design choices for the editor remain to be evaluated. The limitations of the graphical editor will be further discussed at the end of this section as well as in Section 11.2 where we discuss some future work.

The graphical presentation authoring tool was implemented using client-side web technologies such as HTML5, CSS3 and JavaScript. We used libraries such as jQuery-UI[1], Bootstrap[2], jsTree[3], Handlebars[4], Split.js[5], Font Awesome[6] and jQuery tagEditor[7] to facilitate the development of the UI. The editor communicates directly with the information system presented in Chapter 5. An application-specific microservice was added to the server wrapper in order to provide some high-level abstractions for the MindXpres model. This way, less REST API calls need to be made by the client and the API methods provided by the microservice are more specific than the generic methods for working with entities and links. Currently, the MindXpres runtime engine does not interact with the information system directly. To export a presentation, the editor translates the presentation structure into the XML-based MindXpres document format so that it can be compiled into a portable presentation bundle by the MindXpres compiler as described in Chapter 6. In the future we intend to remove the need for the document format and to allow the MindXpres presentation engine to directly fetch content from the information system. Due to time constraints we instead made use of the existing XML-based functionality of the compiler and presentation engine.

Figure 10.7 shows a screenshot of the graphical authoring tool. In the left panel the user's presentations and the associated content is shown as a tree-view with collapsible nodes. By using context menus, nodes can be added or removed. The nodes represent the content, container and layout subtypes defined in the content model presented in Chapter 5. The figure shows how content such as text, bullet lists or images can be aggregated into multi-level presentation structures composed of slides and sections. We also made some additions to the plug-in architecture of the MindXpres presentation engine to support the existing visualisation plug-ins in the graphical editor. First of all, each plug-in now also has to specify the specific entities (as defined in the data model) that it can visualise. This way, a mapping is created between the entities of the content model and the visualisation plug-ins that can visualise them. Second, each plug-in specifies a list of parameters and options that can be modified by the user in the graphical editor. For instance, the slide plug-in allows the user to set the slide title, theme and toggle the automatic layout of

[1] https://jqueryui.com
[2] https://getbootstrap.com
[3] https://www.jstree.com
[4] https://handlebarsjs.com
[5] https://split.js.org
[6] https://fontawesome.com
[7] https://github.com/Pixabay/jQuery-tagEditor

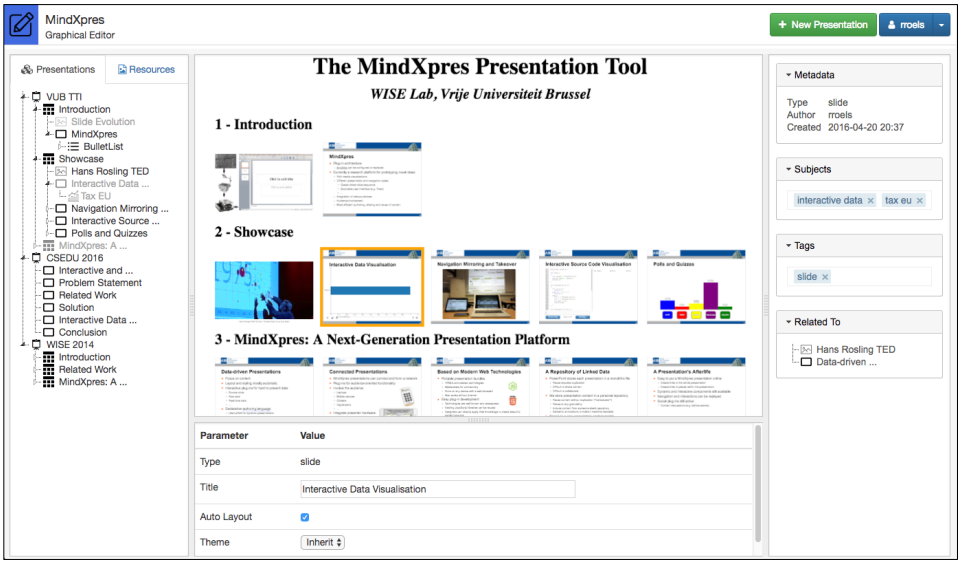


Figure 10.7: A basic graphical editor for MindXpres presentations

content for every slide instance. Every plug-in exports a different set of parameters and settings that is relevant for that specific visualisation plug-in. At run-time, the plug-in takes these settings into account while visualising the content.

By dragging and dropping nodes, parts of the presentation structure can be moved within the presentation and also between different presentations. Nodes can be reused by duplication or by transclusion. In the screenshot, the light grey nodes are nodes that are transcluded from other presentations. Concretely, the top presentation "VUB TTI" reuses an image, a slide and a complete section containing multiple slides from the other presentations shown lower in the tree. If these nodes would be changed in the original presentations, these changes are reflected in the top presentation. In the upper middle panel a real-time preview of the presentation is shown. Currently, a slide is selected and the lower middle shows the parameters and settings that can be applied to the selected slide. The right panel is used to display some metadata and to make some semantic annotations supported by the data model. The screenshot shows how the selected slide is associated with some subjects, some semantic tags and with some related content. By using jQuery tagEditor the tags are auto-completed but users can also apply new subjects and tags.

In summary, the graphical editor allows users to author content using the tree metaphor. Nodes and branches can easily be reused by duplication or by transclusion. Since a node can contain child nodes, larger structures can be reused. The editor further provides some features related to semantic annotations. The editor operates directly

on the presentation data stored in the information system which provides the starting point for collaborative authoring. However, the current implementation of the information system is not yet capable of broadcasting changes to other connected clients. In other words, the foundation for collaborative authoring is provided but this was not fully enabled in the graphical editor and changes made by others are only seen when the whole presentation is reloaded. Other features such as custom navigational paths and user-friendly versioning are not offered in the editor yet. In general, further research is required to bring the advanced functionality of our conceptual and technical foundations to the graphical editor in a user-friendly way.

Part V

Discussion and Future Work

Chapter 11

Conclusions and Future Work

As demonstrated by the variety of topics that have been discussed as part of this dissertation, attempting to improve presentation solutions is a problem that spans many domains. While working towards addressing existing shortcomings and user needs, we have touched subjects such as document engineering, hypermedia, information systems, information visualisation, cognitive psychology, pedagogy and human-computer interaction. This is also reflected by the relatively large amount of references in this dissertation (see bibliography), as many of our research artefacts had different sets of related work, sometimes from different research domains. We start this chapter by providing concrete answers for the research questions we defined in Chapter 1. Answering the research questions also gives us the opportunity to simultaneously summarise our achievements. We end the chapter with future work and a more critical discussion where we detail some of the limitations of our contributions.

11.1 Summary

We start by revisiting the research questions presented in Chapter 1. By concretely answering these questions, we discuss our findings and simultaneously provide a summary of our work.

RQ 1: How is presentation software used and what are the benefits or downsides?

We mostly based ourselves on existing studies to answer this question. However, our programmatic analysis of PowerPoint documents, our observations and the results of our survey also made some contributions towards answering this research question.

An important finding is that the use of presentation software does not automatically result in improved *learning performance* (e.g. exam grades). However, there is evidence that presentation software can improve *learning effectiveness*. Presentation software can help the audience to process information faster with less effort from their side. The combination of auditory and visual information helps the audience to process information and appeals to a broader range of learning styles than for instance a textbook. Presentation software further helps by providing structure to content, aiding the audience in seeing the big picture even if the details are not understood.

A second important finding is that the power of presentation software is in the hands of the user. Presentation software can be applied beneficially if presenters have the knowledge and put in the effort to create good presentations. However, presentation software does not provide help for creating good presentations. If presenters lack the knowledge or feel obliged to apply the large quantity of clipart or animations pushed forward by tools such PowerPoint, the benefit of presentation software can easily be lost or there can even be a negative impact.

To us, a third important finding is that there is no "ideal" presentation style or slide design. Related work, our observations as well as our programmatic analysis show that presentation software is used differently depending on the context. For instance, one presentation style might work very well for a sales pitch, but would not work well for lectures. Even within the use case of lectures, studies show that slide design differs depending on the field or even culture. This helps to explain why presentation guidelines from experts sometimes conflict with each other. Farkas [86] states that a lot of strongly voiced opinions against a particular presentation style originate from people who only have experience with a particular use case with different requirements. Having experienced such criticism ourselves, we found this statement particularly interesting and are inclined to agree on this opinion.

Many other findings are discussed in Chapters 2 and 3, but together with the findings mentioned above, they lead towards a similar conclusion. Presentation software is an important tool in education and business, but it is up to the user to apply them correctly. However, as shown, "correctly" depends on the context and there are no guidelines that work for every case. The authoring view of PowerPoint has many buttons and widgets related to aesthetics but not many of them will help users to design their presentation for the benefit of knowledge transfer. This conclusion further indicates that there is room for improvement in presentation software and has affected some of the design choices in our own solutions.

RQ 2: What are the shortcomings and unmet user needs in existing presentation tools?

Based on existing literature, a survey, observations and a programmatic analysis of more than 12 000 PowerPoint documents, we believe that we have extracted the most important shortcomings and unmet user needs. A first shortcoming is that there are many interesting findings from pedagogy, cognitive psychology and information visualisation that could be beneficial to presentations, but that cannot be applied in existing presentation software. Examples include non-linear navigation, dynamic and interactive data visualisation and pedagogical methods that make the audience a part of the presentation. Although some related work has touched these subjects, solutions are often restricted by the limitations of presentation software such as PowerPoint. A next shortcoming is that some content is simply difficult to integrate in a presentation when using existing presentation software. We previously discussed source code and complex raw data as examples of such content. Existing presentation software often offers only basic content types such as text, images and video, but for anything more complex presenters have to use external software or use tedious and time-consuming workarounds. Another shortcoming that we encountered in literature as well as in the survey is related to the reuse, sharing and collaborative authoring of content. Currently, reusing content implies that it is copied and pasted, which creates duplicate content that is difficult to keep synchronised or collaborate on. Literature also indicates a need for versioning, content discovery, semantic annotations and for keeping track of ownership and reuse permissions. As another point of potential improvement we saw a clear need for better presentation skills and for better presentation content. The findings related to RQ 1 have already indicated that these issues exist, but the findings for RQ 2 show that support on this matter is also highly desired by companies and presenters. However, although companies want better presentations, they also state that they do not want employees to spend too much time on creating presentation content. Ideally, presentation software should help users to create better content with less effort.

RQ 3: How do we need to extend, modify or rebuild presentation software and the underlying technologies to allow researchers and developers to address existing shortcomings and user needs?

We have shown that it is very difficult, if not impossible, to address the shortcomings mentioned in RQ 2 directly in the existing presentation software. As a simple example, PowerPoint enforces the idea that a presentation is always a linear sequence of slides with relatively static content. PowerPoint's view on presentations is not only enforced in the graphical authoring tool but the same restrictions are applied all the way down to the PowerPoint document format. As a result, it is impossible to support different presentation styles or introduce radically new forms of interaction

and content visualisations. There are of course tools such as Prezi that take alternative approaches, but these are similarly restrictive and merely enforce a different view on presentations. As another example, many of the shortcomings related to reuse, sharing and collaboration originate from the fact that PowerPoint presentations are stored as monolithic files. Although it is arguably possible to work around this issue with various layers of middleware and content servers, it would in the end still be a workaround for shortcomings that could have been avoided in the first place. For this reason we deemed it necessary to rebuild the conceptual and technical foundations before improved presentation solutions could be provided.

In order to avoid any technical or ideological restrictions in our foundations, it was important to not make any assumptions on the concept of a presentation. In Chapters 2 and 3 we show that presentations come in different forms and actually need to be adaptable to suit a wide range of use cases. As a way of unifying the shortcomings obtained from RQ 2 with the existing presentation software and state-of-the-art academic work, we first defined a conceptual framework (see Chapter 4). The conceptual framework incorporates the functionality of academic next generation presentation solutions and aims to be generic enough to also support ideas that have yet to come (to some degree). Features of the conceptual framework include support for complex navigational structures, arbitrary content structures and aggregation levels (not only slides), content reuse, collaboration, semantics and context awareness. In other words, our conceptual framework can be seen as a model that can represent existing as well as next generation presentation concepts, defining a much more generic and up-to-date construct for the concept of a presentation. The concepts defined in the conceptual framework were then later propagated to other layers such as the information layer or the visualisation layer, helping us to avoid restrictive design choices.

As a next step of rebuilding the conceptual and technical foundations, we approached the way presentation content is represented and stored. Current ways of storing content make it difficult to address the shortcomings related to reuse, sharing and collaboration, and can often not be extended to support the new concepts we defined in our conceptual framework. We based ourselves on hypermedia concepts, and in Chapter 5 we presented a new content model based on the Resource-Selector-Link (RSL) hypermedia metamodel. The design of the model was driven by requirements such as extensibility, flexibility, fundamental support for reuse, semantics, user access management, versioning and other concepts mentioned as part of the conceptual framework. We have implemented the content model as a functional information system by creating a new model-driven hypermedia link service. As this service can turn any RSL-based model into a functioning information system, it also serves as a reusable foundation for any hypermedia-driven application. By populating our infor-

mation system with a variety of complex randomly generated presentations, we were able to simulate scenarios that involved context adaptation and reuse, verifying that the content model is indeed viable and that it meets our requirements.

In addition to creating a suitable method for storing and managing next generation presentations, we also had to create a new presentation engine that could visualise next generation presentations. In Chapter 6 we presented the requirements, architecture and implementation of the MindXpres presentation engine. Except for a set of core abstractions and supporting APIs, almost everything is implemented as plug-ins, down to the visualisation of simple content types such as text. This allows almost any part of the engine to be modified, replaced or extended, supporting all kinds of presentation types and content visualisations. As such, parts of the visualisation engine can be swapped out on demand to deal with classic slide-based presentations, Prezi-like canvas presentations or any other styles that can be represented in the underlying information system. MindXpres also supports hyperlinks as an additional method for non-linear navigation. In this dissertation we have also provided some example content plug-ins that help presenters to visualise content in interactive and effective ways. A brief summary of proof-of-concept plug-in solutions is provided below as part of RQ 4. It is however worth mentioning that we have taken a data-driven approach to presentation visualisation and we encourage plug-ins to do the same. Concretely, we believe the visualisation engine should handle details such as aesthetics and layout automatically as much as possible. Many of our plug-ins allow the presenter to simply provide the data (e.g. source code or raw data) and an appropriate visualisation based on state-of-the-art findings is automatically created with minimal effort from the presenter's side. We similarly provide a LaTeX-like declarative authoring language based on XML, allowing presenters to focus on content instead of visualisation details. Another feature worth noting is that MindXpres instances can connect into networks, allowing plug-ins for audience-driven functionality or allowing extra hardware to be integrated such as clicker systems or mobile presenter views.

To summarise the answer to RQ 3, in order to fully support solutions that address the shortcomings and requirements presented in Chapters 2 and 3, we had to rebuild the conceptual and technical foundations for presentation tools. This has resulted in a new conceptual framework, content model, information system and visualisation engine. For all of these artefacts flexibility and extensibility was a primary requirement in order to avoid restrictions that might prevent the implementation of radically new presentation solutions.

RQ 4: Given our new conceptual and technical foundation that is free of some of the restrictions seen in other presentation tools, what are the innovative solutions that can be provided for the documented shortcomings and user needs?

Before we discuss individual plug-in solutions, we would like to point out that the MindXpres presentation platform, a combination of artefacts that came out of RQ 3, already addresses some shortcomings on its own. We have already discussed how many of the issues related to reuse, sharing and collaboration are dealt with directly in the information layer, but also the design of the presentation engine addresses some shortcomings. By minimising the restrictions we impose on plug-in developers, we make it much easier to bring findings from domains such as pedagogy, information visualisation and cognitive psychology to presentation solutions. The presentation engine supports (among other styles) zoomable canvas presentations which provide a better overview on the content and make better use of our natural ability for spatial reasoning. The navigational links described in the content model manifest themselves as a tool for non-linear navigation, and hyperlinks in the visualisation layer allow complex navigational paths within and across presentations. The presentation engine's ability to form networks allows us to better integrate the audience in a presentation and enables the integration of a variety of hardware, for instance allowing the presenter to be more mobile. By using web technologies, MindXpres presentations do not have any major software dependencies and run on any platform with a modern web browser. This design choice also ensures that MindXpres presentations always look the same on every platform. In contrast, a related study, as well as our own survey, confirm that PowerPoint users have issues with the fact that presentations are incompatible or look different between different versions of PowerPoint [188]. MindXpres presentations are easy to share after the presentation session, with all interactive functionality intact and with the option to play back the original navigational path and annotations.

In addition to the inherent benefits of the MindXpres platform, we demonstrate the platform's flexibility with a wide range of additional proof-of-concept plug-in solutions. The plug-ins show how some of the shortcomings and unmet user needs that were identified as part of RQ 3 might be addressed. One group of plug-in solutions presented in Chapter 7 focusses on presenting content that is difficult or time-consuming to present with existing presentation software. As an example, we provide plug-ins for visualising source code in a dynamic and interactive way, helping students to strengthen their mental model. From the presenter's perspective, using the plug-in is as simple as specifying the source code that needs to be visualised, with no further effort required. Similarly, we provide a plug-in for creating dynamic data visualisations that change over time in support of the presenter's narrative. Other examples include support for mathematical expressions via MathML or LaTeX, an advanced

video player and an embedded Linux environment for teachers. A next group of plug-ins presented in Chapter 8 focusses on involving the audience. We provide plug-ins for the well-known quizzes and polls, but also provide some novel solutions such as digital question asking, note sharing, mirroring the presentation on mobile devices in the audience and allowing audience members to take over navigation. The next category of solutions presented in Chapter 9 focusses on helping presenters to improve their oral performance. For this reason some of our solutions aim to make interactions with the presentation software more natural. One solution focusses on providing interactions by means of a laser pointer, touch and gestures. Another solution provides a tablet-based presenter view giving the presenter a better overview, encouraging non-linear navigation, allowing annotations with specific support for attentional ink and in general allows presenters to be more mobile. We also investigated the use of augmented reality for a presenter view, further enhancing the mobility of the presenter and making interactions much more discrete. Next to these interaction interfaces we also focussed on a tool for providing presentation feedback. As the rules for a "good" presentation, for instance, depend on the language, culture or presentation setting, we use a rule engine allowing different thresholds to be set depending on the context. A plug-in architecture allows different hardware to be used to generate a variety of performance metrics in an extensible way. Finally, the feedback system's integration with the MindXpres platform gives the system access to the content allowing feedback plug-ins to also provide feedback on the content itself. A final solution in this category investigated the use of a virtual assistant during presentations. In Chapter 10 we have presented a set of solutions that focus on helping the presenter to create content. One plug-in solution is a tool for helping users select a visualisation style that works well for the provided data, and is based on a compilation of existing guidelines. Another plug-in allows us to import existing PowerPoint content into the MindXpres platform. In order to seamlessly blend imported content into a MindXpres presentation, a basic constraint-based content layout mechanism is provided. Last but not least, we presented a graphical authoring tool for MindXpres themes as well as a graphical authoring tool for MindXpres presentations.

11.2 Discussion and Future Work

At the start of this chapter we have mentioned how our research trajectory has led us to a wide range of solutions that span a variety of research domains. We believe that our work would have made a much smaller impact if we had only focussed on a smaller research problem such as the underlying information system for presentation solutions. However, given the limited time frame of this research project, we had to prioritise certain artefacts over others and had to make some strategic choices as to where to best spend our resources. Throughout this dissertation we have frequently

stated that we provide an improved conceptual and technical *foundation*. In order to allow us, or others, to create novel presentation solutions for existing shortcomings, we prioritised the completeness of this foundation. To illustrate, we made sure that our conceptual model spans all existing presentation concepts and is abstract enough to be somewhat futureproof. We have implemented the resulting content model as a functioning information system and we simulated all of its functionality with complex randomly generated presentations. The wide range of proof-of-concept plug-in solutions further helped us to verify the presentation engine, and the potential of MindXpres as the foundation for next generation presentation solutions. In a way, the MindXpres platform can be compared to the Eclipse[1] platform. In essence, Eclipse is an extensible plug-in based framework that can be used to build IDEs for different programming languages. For instance, Google built the earlier versions of Android Studio[2] on top of the Eclipse platform. Although the goals of Eclipse are of course very different, the MindXpres platform is comparable in the sense that it serves as a shared framework that allows researchers and developers to create new presentation solutions and to tailor the end result to suit their needs. Similar to how Eclipse is used to build IDEs for specific programming languages, MindXpres offers a common platform for building dedicated presentation tools for specific use cases. To illustrate, one could use MindXpres as a framework for creating presentation software specifically for teaching students how to program, or specifically for primary schools.

Because of our choice to prioritise an enabling foundation over concrete end user solutions, the most important limitations of this dissertation are related to end user usability. We have presented a basic graphical authoring tool for MindXpres presentations, but bringing the full potential of our conceptual and technical foundations to end users is a major challenge that requires a significant amount of additional research. As motivated in Section 10.3, a more complete graphical editor requires the development of architectural as well as UI abstractions that work for all potential content plug-ins, and we need to investigate appropriate metaphors to bring their complex functionality to end users. As seen, a lot of our contributions are enablers for more solutions in the future, but many of the possibilities remain unexplored. For instance, the semantic metadata in our content model and resulting information system could be used for a variety of purposes. In the future, as part of a graphical authoring tool, we would like to explore semantic querying, content recommender systems and the (semi-)automatic assembly of presentations. We designed the semantic layer to be interoperable with existing ontologies, so we also intend to create mappings to existing learning material standards such as SCORM. Such a mapping would not only allow MindXpres to integrated existing material from learning material repositories, but this would also allow other systems to index and use MindXpres content with-

[1] `https://www.eclipse.org`
[2] `https://developer.android.com/studio`

out modifications on their side. Although some of these subjects have already been touched by related work, our foundation paves the way for improvements and solutions that are better integrated in the presentation software. The interoperability of the semantic metadata would also allow content from existing learning packages to be integrated in presentations, which is definitely worth investigating.

We mentioned how our information system for presentation software is complete from a research perspective, but when discussing its implementation in Section 5.7 we mentioned some possible future technical enhancements. For instance, we discussed how versioning could be further optimised by using delta compression. In the future we might also consider a dedicated syntax for defining contextual rules directly as part of the model definitions. We have also explained how we currently use the facilities offered by ObjectDB to support a basic form of schema evolution, but in the future this should be generalised to support other persistence backends and should also support evolution on a finer granularity than complete entities.

As a next point of discussion we would like to mention our research on existing shortcomings and unmet user needs (Chapter 3). Some of our findings have not yet been published, even though we see them as valuable contributions in the related research domains. However, we believe that it would be beneficial to increase the number of survey participants first, to further strengthen the results. As discussed in the related chapter, the current number of participants prevented us from demonstrating statistical significance for some of our findings. Concerning the results of the programmatic analysis of PowerPoint documents, we see the resulting dataset and database with extracted information as a contribution that will provide further utility after this dissertation. For this dissertation we have only focussed on processing the information that was the most relevant. However, the database contains much more unprocessed metadata that is likely to lead to interesting findings that have not yet been published before. As part of future work, we intend to further analyse the data we have, which might lead to another publication. One shortcoming of our programmatic analysis is that the use case of the presentations is unknown and cannot be automatically derived. As an example, being able to classify presentations as a lecture or sales pitch would lead to new insights. One might hypothesise that the use of images or the amount of slides is significantly different depending on the use case. Similarly, it is unknown what the images represent and classifying the images within the presentations might further provide new findings. We manually processed 254 presentations and all the images within so that we could measure how often certain content such as charts or diagrams were included as images. However, in the future this could be done on a larger scale with a more complete classification. Nevertheless, even in its current state we see the dataset and database as a contribution that will further help us and others in the future.

We have shown earlier that the ability to connect MindXpres presentations and peripheral hardware into networks allows for some new interesting use cases. We presented some implemented use cases where multiple devices are used as presenter views, as input for the same presentation, or where the presentation output is mirrored across different devices. However, we have not yet fully explored the possibility of using multiple displays or projection areas to show different parts of the same presentation. This idea is explored by MultiPresenter [164], but the flexibility of the MindXpres platform might allow us to make the use of multiple displays more effective, or MindXpres might be used to reduce the complexity and effort required to operate multiple displays simultaneously. This is especially important considering that some studies show that classrooms are transitioning from one central display to multiple screens [274].

Another limitation is that we have not been able to evaluate the effect of our solutions on knowledge transfer. While some of our solutions incorporate proven principles from other research domains, we have yet to demonstrate that certain MindXpres plug-ins contribute to more effective learning. We did however perform technical or usability evaluations on many of our research artefacts.

In general, as an enabling platform MindXpres paves the way for a new wave of creative solutions. By providing easy to use abstractions for visualisations, interactivity and connectivity we allow developers to bring presentation-based content delivery to a new level. We see that there are many more scenarios, especially in teaching, that could benefit from the development of dedicated content plug-ins. The possibilities are endless but some examples that come to mind is a plug-in for visualising SQL queries in lectures, or a plug-in for visualising and playing back sheet music in support of music education. Innovation is not limited to new content plug-ins but as shown by our own plug-in solutions, the platform also encourages the development of new types of presenter views and interaction modalities for the presenter. We have also laid the foundation for a new type of feedback system that can be extended to support all kinds hardware and metrics. Our approach allows others to customise the feedback system for presentation skills that are important for their use cases, and not only use it with metrics and thresholds that have been preprogrammed by others.

As part of this discussion we would also like to highlight how MindXpres enables co-creation between different parties, on different levels. First of all, the functionality of PowerPoint is determined by Microsoft and parties such as teachers or students have to work with what Microsoft provides them. In contrast, MindXpres can easily be extended and modified, and different parties can help shape the presentation software by contributing plug-in solutions. The functionality of the MindXpres platform is therefore not determined by us, but the different stakeholders can co-create presentation software that meets the evolving requirements of all the involved parties. Next, the platform's focus on improved content reuse, sharing and collaboration helps

different parties to also co-create presentation content that might consist of input from various sources. Finally, we also point out that the MindXpres engine's communication component provides support for co-creation during presentations. Similar to some other audience-driven solutions we have shown earlier, the platform allows the creation of plug-in activities that capture and combine input from various parties (e.g. students and teachers) for the purpose of co-creation.

In order to further highlight some of the strengths and weaknesses of our contributions we discuss some of the themes that were defined by Myers [199] for evaluating software tools. We define two groups of end users:

- Researchers and developers: end users that use the MindXpres platform to build new presentation solutions.
- Presenters: end users that use the MindXpres platform and its plug-in solutions to author and deliver presentations.

In the context of this dissertation, researchers and developers are the main target audience for our conceptual and technical foundations. However, the users of the solutions built on top of our foundations are also important for the following discussion, and we will refer to this group of end users as "presenters".

Threshold: how difficult it is to learn how to use the system?

As mentioned before, we focussed on providing the conceptual and technical foundations that enable the development of new solutions. Although we discussed some proof-of-concept solutions and graphical editors in this dissertation, further research is required to bring the advanced functionality of the platform to presenters, for instance via graphical authoring tools. A minority of our plug-in solutions were evaluated for usability, but the platform was not evaluated as a whole, and therefore it is not possible to make a statement about the threshold for presenters that would use MindXpres for authoring and delivering presentations.

However, we can discuss the threshold from the perspective of the researchers and developers that use the platform to build their own solutions. If a new solution is built by simply combining existing plug-ins, the threshold is of course very low. If a plug-in has to be extended or created, then the threshold is higher and some programming skills are required. However, our choice of using web technologies helps to keep the threshold as low as possible. JavaScript is relatively easy to get started with, to get immediate results, and the existing (free) programming tools allow for easy debugging. The technical foundations of the platform abstracts a lot of the technical complexity away from the plug-ins so that developers only need to worry about implementing the desired functionality. Furthermore, developers can make use of the many great existing JavaScript libraries to help them achieve their goals. In

some cases the development of a MindXpres plug-in is a simple as writing a few lines of "glue code" to integrate an existing library that performs all the work. Finally, the use of CSS in plug-ins and themes allows people such as web designers to apply their skills and tools to MindXpres without necessarily requiring knowledge of JavaScript.

Ceiling: what is the upper limit of what can be done with the system?

The MindXpres platform was designed to be flexible and extensible and ultimately the functionality is determined by its plug-ins and extensions. For this reason, from the perspective of the presenters, the ceiling is determined by the available plug-ins and authoring tools.

From the perspective of researchers and developers, any missing functionality can easily be added by developing new plug-ins. For this reason, the ceiling is not determined by the functionality of the platform itself, but it is mostly determined by the limitations of the underlying technologies. Our foundations were designed to avoid most technical or ideological restrictions, but we see our web-based presentation engine as the most likely source of potential limitations. However, given the capabilities of modern web technologies, the ceiling is still relatively high. New HTML5 APIs, new JavaScript language features and improved web browser engines enabled web applications where the performance and functionality is similar to native desktop applications. For instance, HTML's event system allows us to capture user interactions, Canvas and WebGL APIs enable hardware-accelerated 2D and 3D rendering, MediaStreams allow us to capture and process audio and video, and WebSockets enable full-duplex communication over TCP. Also note that all this functionality is available on any platform with a modern web browser, including mobile devices. As demonstrated by the embedded Linux plug-in (Section 7.7) or the Doom plug-in (Section 7.8), web technologies can be used to bring rich applications to the MindXpres platform. However, *only* web technologies can be used, and we believe that this is the limitation that will determine the ceiling for developers. Currently, there is no practical way of embedding native desktop applications in MindXpres. A lot of features and functionality of existing desktop applications can be replicated using web technologies, but sometimes it is unfeasible to reimplement an entire application in web technologies.

Path of Least Resistance: how does the tool push users towards doing the right things and away from doing the wrong things?

As discussed in Chapters 2 and 3, tools such as PowerPoint offer a lot of functionality related to visual details such as layout, fonts, colours and animations, but these tools do not help the user to apply them in a beneficial way. It is up to the presenter to reap the benefits of presentation software, but it is also very easy to apply the offered functionality with negative consequences. We remedy this issue by taking a data-driven approach in the foundations as well as in some of our proof-of-concept

solutions. One example is the declarative authoring language (Section 6.3.1). Similar to how LaTeX is used, the user focusses on defining the content and the platform handles layout and other visual details automatically. As another example, some of our plug-in solutions simply take raw data as input and automatically turn them into dynamic and interactive visualisations with no further input from the user. Our interactive source code plug-in (Section 7.4) takes source code as input, and the plug-in then automatically detects the programming language, applies formatting and syntax highlighting, and automatically generates some visualisations that show how the code is executed over time. In the same way, our plug-in for interactive and narrative data visualisation (Section 7.6) only requires the presenter to provide a dataset and to define a series of "views", and the plug-in handles the actually visualisation and interactive components automatically. As a benefit, these plug-ins allow presenters to create rich interactive content with almost no effort, and in our case the plug-ins apply findings and guidelines from other academic work to automatically provide visualisations with cognitive benefits.

However, there are of course some downsides to this approach. In order to ensure a proper visualisation, the data-driven approach takes away some of the presenters' freedom. To still provide some flexibility, many plug-ins take some configuration parameters and plug-ins can always be modified if needed. As another potential downside; if a data-driven approach is taken, the plug-in developer is responsible for making good design choices for the resulting visualisation. We based ourselves on existing research for our plug-in solutions, but personal preferences or different use cases might cause conflicting opinions. We refer back to the popular LaTeX typesetting engine, where a similar approach is successfully taken, to illustrate that the benefits of a data-driven approach outweigh the downsides for the average use case.

From the developer perspective, the platform provides the necessary abstractions to hide a lot of the technical complexity and to encourage developers to follow the principles on which MindXpres is built. However, one of our main goals was to not enforce technical or ideological restrictions, and it is always possible to ignore the provided abstractions and to take a different approach. The developers are free to do as they see fit, and given the use of a flexible programming language such as JavaScript, it would be almost impossible to enforce certain practices even if this was desired.

Predictability: how predictable are the "automatic" features of the system?

The core foundations do not offer automatic functionality that might result in unexpected results, so this theme is mostly relevant for the presenters. The previously discussed data-driven approach is perhaps the most important factor that can result

in potentially unexpected results. For instance, MindXpres can handle content layout automatically, but the suggested layout might not be exactly what the presenter envisioned and some manual tweaking may be required. As of now, the available plug-ins are mostly developed by us but in the future one might imagine an "app store" for MindXpres plug-ins where developers can publish their creations. However, this introduces new problems that other app stores also face. It would be difficult, if not impossible to ensure a high quality for plug-ins, and plug-ins from third parties might give undesirable results. Finally, we also point out transclusion as a source of unexpected changes. If content is included from another source by transclusion, any changes that are made to the transcluded content are also reflected in your presentation. Sometimes this is desired, but in other cases the content might change so much that it does not suit your presentation anymore. To address this issue we provided versioning for content and associations, and presenters can "lock on" to a specific version of the content so that changes will not affect the presentation in an unexpected way.

Moving Targets: how does the system keep up with the continuously changing tasks that it supports?

From the technical perspective, we deliberately chose HTML5 and related technologies to also make the system somewhat future-proof on the technical level. When we designed the presentation engine, HTML5 was just in the draft phase and support in web browsers was mediocre at best. However, eventually web browsers caught up and as a "living standard" we are guaranteed that the standard will not be replaced any time soon, and that any updates to the standard will not break existing code. Other technical dependencies include Java, for instance for our information system, which is also an industry standard that will remain relevant for a long time.

The MindXpres platform was designed from the ground up to be flexible and future-proof up to some degree. We make no assumptions about what a presentation should be, also not in the conceptual framework or content model, making the platform usable for presentation concepts that have yet to be born. As the concept of a presentation continues to evolve, our design choices ensure that the MindXpres platform can evolve alongside new user needs, by only extending the components that were designed to be extended. However, some modelling or programming skills are of course required to extend the content model or to implement new plug-ins.

11.3 Final Conclusions

We started this dissertation with identifying shortcomings and unmet user needs in presentation software. Next, we have shown that the discovered shortcomings could not easily be addressed in existing presentation solutions and their related technologies. For this reason we had to rebuild the conceptual and technical foundations in order to properly support next generation presentation concepts. To demonstrate the flexibility of the resulting MindXpres presentation platform, we discussed some proof-of-concept solutions for the identified shortcomings, and we go beyond what is possible in existing presentation software. The MindXpres platform not only allowed us to develop new solutions, but as a research platform MindXpres also enables other researchers and developers to easily prototype and evaluate novel presentation concepts. For this reason we hope that our research artefacts further contribute to a new generation of presentation solutions, which might ultimately lead to improved knowledge transfer.

Part VI

Appendices

Appendix A

Related Student Contributions

The following related Bachelor's and Master's theses were supervised by Reinout Roels and have contributed to the work presented in this dissertation.

2012 - 2013

- Kevin van Gyseghem, 2013. "*Dynamic Graph Visualisations for the MindXpres Presentation Tool*". Bachelor's Thesis. Vrije Universiteit Brussel, 2013.
- Jeroen Heymans, 2013. "*A WYSIWYG Template Authoring Solution for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2013.
- Christophe Vermeylen, 2013. "*Audience-driven Presentations based on the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2013.

2013 - 2014

- Paul Meștereagă, 2014. "*Intelligent Source Code Visualisation for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2014.
- Jasper Debie, 2014. "*Guided Data Visualisation for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2014.

2014 - 2015

- Yves Baeten, 2015. "*Intuitive Information Visualisation for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2015.

- Joris Vandermeersch, 2015. "*Content Migration and Layout for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2015.
- David Bos, 2015. "*Multimodal Interaction for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2015.
- Chunxian Yan, 2015. "*TabletPoint: A Pen and Tablet-based Presentation Solution*". Master's Thesis. Vrije Universiteit Brussel, 2015.

2015 - 2016

- Jonas Vereyken, 2016. "*Investigating Graph-based Storage Backends for Link Services*". Master's Thesis. Vrije Universiteit Brussel, 2016.
- Mohamed Zahir, 2016. "*An Extensible Platform for Real-Time Feedback in Presentation Training*". Master's Thesis. Vrije Universiteit Brussel, 2016.

2016 - 2017

- Kushal Soni, 2017. "*An Extensible Virtual Assistant for the PowerPoint Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2017.

2017 - 2018

- Kwinten Pardon, 2018. "*An Extensible Presentation Framework for Real-time Data Acquisition, Interaction and Augmented Reality Visualisations*". Master's Thesis. Vrije Universiteit Brussel, 2018.

Appendix B

Database Schema for Programmatic Analysis Results

Presentations	
Column Name	Description
presentationID	Unique identifier for every presentation
fileName	Current file name of the presentation in the data set
originalFileName	Original file name before it was renamed
fileSize	Size of the presentation file in bytes
hash	MD5 hash of the presentation file
lang	Language of the presentation content
slideCount	Total amount of slides (including hidden slides)
hiddenSlideCount	Amount of hidden slides
editTime	Total edit time (in milliseconds)
format	Slide format (e.g. "4:3")
applicationName	PowerPoint application name
applicationVersion	PowerPoint version number
author	Author name
applicationTemplate	Name of the currently applied presentation template
category	Topic of the presentation
createdTime	Creation time (UNIX timestamp)
lastModifiedTime	Last modified time (UNIX timestamp)
description	Presentation description
keywords	Keywords
lastSavedBy	Name of the last editor
subject	Presentation subject
title	Presentation Title
pixelColors	Histogram of all slide pixels

Slides	
Column Name	Description
slideID	Unique identifier for every slide
presentationID	Foreign key pointing to the containing presentation
slideNumber	Number of the slide within its presentation
isHidden	Binary value indicating if the slide is hidden
hasNote	Binary value indicating if the slide has presenter notes
notes	Presenter notes (if any)
transitionType	Transition effect
advanceAfterTime	Time interval before the slide advances
advanceOnClick	Binary value indicating if the slide advances on clicks
whiteSpace	Percentage of empty (non-content) space in the slide
imageCount	Amount of images
audioCount	Amount of audio fragments
videoCount	Amount of video fragments
chartCount	Amount of charts
diagramCount	Amount of diagrams
formulaCount	Amount of mathematical formulas
hyperlinkInternalCount	Amount of hyperlinks to internal content
hyperlinkExternalCount	Amount of hyperlinks to external content
charCount	Amount of characters on the slide
charCountNoWhitespace	Amount of characters without whitespace (e.g. spaces)
wordCount	Amount of words on the slide

Text	
Column Name	Description
textID	Unique identifier for every text fragment
presentationID	Foreign key pointing to the containing presentation
slideID	Foreign key pointing to the containing slide
textNumber	Number of the text within its slide
text	The text fragment itself
bulletType	The type of bullet (if any)
bulletDepth	The depth of the bullet (if any)
charCount	Amount of characters in the text fragment
charCountNoWhitespace	Amount of characters without whitespace (e.g. spaces)
wordCount	Amount of words in the text fragment
animation	Binary value indicating if this text is animated

Images	
Column Name	Description
imageID	Unique identifier for every image
presentationID	Foreign key pointing to the containing presentation
slideID	Foreign key pointing to the containing slide
imageNumber	Number of the image within its slide
hash	MD5 hash of the image data
imageName	Original name of the image
encoding	Encoding of the image (e.g. "JPG")
fileSize	Size of the image (in bytes)
width	Width (in pixels) of the unscaled image
height	Height (in pixels) of the unscaled image
animation	Binary value indicating if this image is animated

Appendix C

Questionnaires

C.1 System Usability Scale (SUS)

The System Usability Scale (SUS) [42] was introduced in 1996 and has since proven to be a reliable usability scale for the assessment of a system's usability. The scale has been shown to work well with relatively small sample sizes and studies show that in 50% of the cases the SUS score from a sample size of 5 was within 6 points of the true SUS score[1].

Questionnaire

Participants are asked to score the following 10 statements on a 5-point Likert Scale (e.g. Strongly Disagree - Strongly Agree):

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.

[1] `https://measuringu.com/publications/`

8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

Scoring and Interpretation

The use of a Likert Scale provides each question with a score from 1 to 5. For the odd-numbered questions a higher score is better and for the even-numbered questions a lower score is better. To calculate the total score, the scores for odd questions are decreased by one point, and scores for even questions are flipped by subtracting them from 5. All scores are then added and multiplied by 2.5 to obtain a total SUS score between 0 and 100.

To interpret these scores, they should not be seen as percentages of usability but as a metric that provides a way of comparing the usability between systems. For this reason they should be considered in terms of their percentile ranking. For instance, by comparing the scores of thousands of systems Jeff Sauro determined that a SUS score of 68 corresponds to the 50% percentile and can therefore be considered as a threshold for a system being below or above average[1]. Later work has similarly used large collections of SUS results to provide an adjective rating for the SUS scale [21]. Note that the SUS scores below represent the mean score for each adjective, and not thresholds.

SUS Score	Adjective
12.5	Worst Imaginable
20.3	Awful
35.7	Poor
50.9	Ok
71.4	Good
85.5	Excellent
90.9	Best Imaginable

[1] https://measuringu.com/interpret-sus-score/

C.2 Computer System Usability Questionnaire (CSUQ)

The Computer System Usability Questionnaire (CSUQ)[169] was first introduced in 1995 and is used to evaluate the usability of computer systems.

Questionnaire

Participants are asked to score the following 19 statements on a 7-point Likert Scale. However, note that the scale should range from Strongly Agree to Strongly Disagree, so a lower score is always better.

1. Overall, I am satisfied with how easy it is to use this system.
2. It was simple to use this system.
3. I can effectively complete my work using this system.
4. I am able to complete my work quickly using this system.
5. I am able to efficiently complete my work using this system.
6. I feel comfortable using this system.
7. It was easy to learn to use this system.
8. I believe I became productive quickly using this system.
9. The system gives error messages that clearly tell me how to fix problems.
10. Whenever I make a mistake using the system, I recover easily and quickly.
11. The information (such as online help, on-screen messages, and other documentation) provided with this system is clear.
12. It is easy to find the information I needed.
13. The information provided for the system is easy to understand.
14. The information is effective in helping me complete the tasks and scenarios.
15. The organization of information on the system screens is clear.
16. The interface of this system is pleasant.
17. I like using the interface of this system.
18. This system has all the functions and capabilities I expect it to have.
19. Overall, I am satisfied with this system.

Participants are also asked to list the most *negative* aspects of the system, as well as the most *positive* aspects.

Scoring and Interpretation

The CSUQ produces four scores (an overall score and three subscales) that are calculated as following.

1. Overall Score ("OVERALL"): the average of items 1 - 19
2. Effectiveness Score ("INTERQUAL"): the average of items 1 - 8
3. Information and Documentation Score ("INFOQUAL"): the average of items 9 - 15
4. Ease of Use Score ("SYSUSE"): the average of items 16 - 18

It does not matter if the final scores are calculated for each participant separately and then averaged, or if the average is immediately calculated across the question scores of all participants. This is because all participants are guaranteed to answer the same amount of questions and thus the average of averages does not need to be weighted, giving the same result.

C.3 Custom Survey

The following survey was used to gain insight into how people use presentation software and the results helped us to identify some shortcomings and unmet user needs. The results of this survey are discussed in Chapter 3. The survey was hosted on an online LimeSurvey[1] instance. The questions below are presented without any visual markup, but the LimeSurvey platform visualises them in an attractive way and even helps the user to fill in the questions correctly.

1. What is your age?

   ○ 0 - 12
   ○ 13 - 18
   ○ 19 - 29
   ○ 30 - 39
   ○ 40 - 49
   ○ 50 - 64
   ○ 65+

2. What is your gender?

   ○ Male ○ Female ○ Other: ☐

3. What is your computer skill level?

   ○ Poor ○ Fair ○ Good ○ Very Good ○ Excellent

4. In which sector do you work?

   ○ Student
   ○ Primary/Secondary Education
   ○ Academia
   ○ Finance and Insurance
   ○ Information Technology (IT)
   ○ Healthcare
   ○ Transportation
   ○ Public Administration
   ○ Publishing
   ○ Manufacturing
   ○ Wholesale and Retail Trade
   ○ Leisure and Hospitality
   ○ Other: ☐

[1] `https://www.limesurvey.org`

5. How often do you create presentations?

- ○ Multiple times a week
- ○ Once a week
- ○ Once a month
- ○ A few times a year
- ○ Once a year
- ○ Less than once a year
- ○ Never

6. How often do you give a presentation?

- ○ Multiple times a week
- ○ Once a week
- ○ Once a month
- ○ A few times a year
- ○ Once a year
- ○ Less than once a year
- ○ Never

7. On average, how long do your presentations take (in minutes)?

8. On average, how many slides do you present during the time you specified in the previous question?

9. On average, how much time do you spend per slide (in minutes) when creating a presentation (excluding practise)?

10. When you create a presentation:

- Some part of your time goes towards creating content (e.g. text, charts, images or diagrams)
- Some part of your time is spent on finetuning the content's visual details such as layout, animations, fonts or colors

  If we assume that these two activities together take up 100% of your time when creating a presentation, what percentage of your time is spent on visual details (layout, animations, fonts, colors, ...)?

11. What are your most important reasons for giving presentations?

- ☐ Academic knowledge transfer (e.g. teaching, conferences or project presentations in an academic setting; either as teacher or as student)
- ☐ Corporate knowledge transfer (e.g. meetings, presenting results or employee training within a company)
- ☐ Sales pitch (presenting with the goal of selling a product)
- ☐ Entertainment (e.g. holiday picture slideshows)
- ☐ Other: ____

12. If you use presentations for academic knowledge transfer, please list all the resources that you provide or reference to help explain the topic.

For instance, in the case of a lecture the list of relevant material might include your PowerPoint slides, a third-party textbook, handouts or video footage.

Please order your list so that the most important resources are at the top. Make sure to include your presentation (i.e. slides) in the list too. Not all fields have to be used.

13. What is your prefered software for creating your presentations?

- ○ PowerPoint
- ○ Keynote
- ○ Google Slides
- ○ Prezi
- ○ Beamer (LaTeX)
- ○ None
- ○ Other: ____

14. Please specify any additional software you use while giving your presentations.

Examples: a PDF viewer, a web browser, video playback software, a programming IDE, or software for data visualisation (e.g. Tableau).

15. What type of hardware setup(s) do you use while giving your presentation?

Examples:

- a Dell laptop running Windows 7 connected to a projector
- a MacBook connected to a large television display

16. Please specify any additional tools you use while giving your presentations. Examples include whiteboards, paper, handouts, pamphlets, laser pointers, digital pens, tablets, smartphones or DVD players.

17. Do you ever use any tools (software or hardware) to involve the audience during your presentation (e.g. real-time voting or quizes via clickers)? If so, please specify which tools you use.

18. What kind of content do you use in your presentations?

- ☐ Text
- ☐ Images
- ☐ Tables
- ☐ Charts
- ☐ Diagrams
- ☐ Audio
- ☐ Video
- ☐ Mathematical formulas
- ☐ Web links to external material
- ☐ Source code
- ☐ Geographical Maps
- ☐ Other(s):

19. Do you sometimes experience issues when integrating any of the above content into your presentations? Please explain.

20. Do you use any other software to create or pre-process content before integrating it into your presentation tool? Examples include image editors, diagram/chart creation tools or programming environments. If so, please briefly mention what you do there.

21. How often do you include content from an existing presentation that you (or your team) has created?

○ Always ○ Often ○ Sometimes ○ Rarely ○ Never

22. How often do you include content from external resources (e.g. from the Web or from other people's presentations)

○ Always ○ Often ○ Sometimes ○ Rarely ○ Never

23. Please list the most used external sources (e.g. Wikipedia, Google Images, SlideShare, ...)

24. Which of these statements related to presentation themes applies the most to you?

☐ I use simple plain blank slides without a theme
☐ I pick one of the default themes that appeals to me
☐ I change an existing theme or create a new one that suits my needs
☐ I use the school's or company's theme
☐ Other:

25. How often do you make use of animations while giving your presentations. This includes for instance showing bullets one by one or slide transitions.

○ Always ○ Often ○ Sometimes ○ Rarely ○ Never

26. Why do you use animations?

Examples:

- to capture attention
- to help the audience process the information

27. Do you make use of your presentation software's presenter view?

- ○ Yes
- ○ No
- ○ My tool does not have a presenter view
- ○ I don't know what a presenter view is

28. Which feature(s) of the presenter view do you use?

- ☐ Overview of current slide / notes
- ☐ Slide navigation
- ☐ Time management
- ☐ Other: ☐

29. Why don't you use the presenter view?

☐ No Need ☐ Other: ☐

30. Do you think anything important is missing in the presenter view? If so, please specify.

☐

31. How do you keep track of time while giving your presentations?

- ☐ I don't keep track of time
- ☐ System clock on the computer/tablet which runs your presentation
- ☐ Presenter view
- ☐ Wristwatch
- ☐ Smartphone
- ☐ Other: ☐

32. How do you navigate your content while giving your presentations (e.g. switch between slides)?

- ☐ Keyboard
- ☐ Mouse / trackpad
- ☐ Presentation remote / pointer
- ☐ Smartphone / smartwatch
- ☐ The presentation software automatically shows the next piece of content after a predefined amount of time
- ☐ Other: ☐

33. How often do you show slides out of order (not in the predefined order) while giving your presentations?

○ Always ○ Often ○ Sometimes ○ Rarely ○ Never

34. Do you sometimes show the same content multiple times while giving a presentation?

Examples:

- you include the same image or table on multiple slides
- you go back to a previously visited slide during the presentation

○ Yes ○ No

35. How do you show the same content multiple times while giving a presentation?

☐ I include the same content multiple times in the presentation at different locations
☐ I include the same content in the presentation multiple times at different locations, but sometimes with some modifications
☐ I navigate back (or forward) to show specific content multiple times
☐ Other: [ ]

36. How often do you practise a presentation before giving your presentation to a real audience?

○ Always ○ Often ○ Sometimes ○ Rarely ○ Never

37. How do you practise your presentation?

☐ Read through the content
☐ Talk out loud
☐ Film yourself
☐ Present to your family/friends/colleagues as a test
☐ Other: [ ]

38. How often do you collaborate with other people while creating a presentation?

○ Always ○ Often ○ Sometimes ○ Rarely ○ Never

39. How do you collaborate while creating a presentation?

☐ Everyone creates their part separately and at the end someone merges everything together
☐ We pass the file around (e.g. via email/ftp/im/dropbox) and take turns to work on the document
☐ We use software that allows multiple people to work on the presentation at the same time (e.g. Google Slides)
☐ We synchronise our contributions via software such as SharePoint or SVN/GIT
☐ Other: [ ]

40. How often do you make your presentation content available for the audience after the presentation?

○ Always ○ Often ○ Sometimes ○ Rarely ○ Never

41. How do you make presentation content available after the oral presentation?

☐ Manually sent via email, or instant messaging
☐ School/company platform (e.g. Canvas) or server (e.g. via FTP)
☐ Shared via generic online storage platform such as Dropbox, Google Drive, Microsoft OneDrive, ...
☐ Specialised online presentation platforms such as SlideShare or SlideRocket
☐ Other: [ ]

42. How important would you rate the presentation content as reference material for later use?

For example, for students that use the slides as study material, or for training new employees.

○ Very Important
○ Fairly Important
○ Important
○ Slightly Important
○ Not important

43. Please clarify the score you gave in the previous question (the importance of your presentation content as reference material).

For example, is your presentation not important because there is a textbook? Or is it very important because there is no other material?

[ ]

44. Is the presentation content used for purposes other than the presentation itself? For instance as a report or as study material?

○ No ○ Yes, purpose: [ ]

45. What do you like most about the presentation software of your choice?

[ ]

46. What do you think is lacking in presentation software?

47. Do you have other feedback or comments?

Three €50 vouchers for Amazon will be given away to randomly chosen participants. Do you want to participate in the raffle?

If so, please provide a valid email address so that we can contact you if you have been selected.

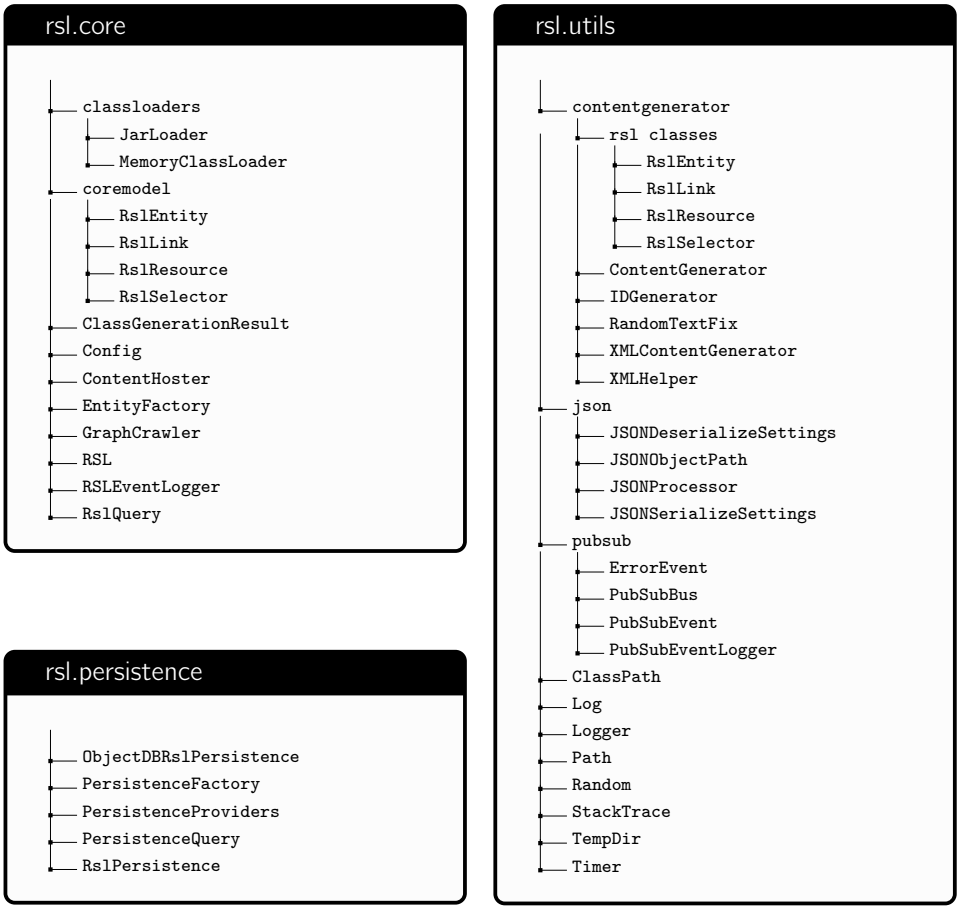
○ No ○ Yes, email:

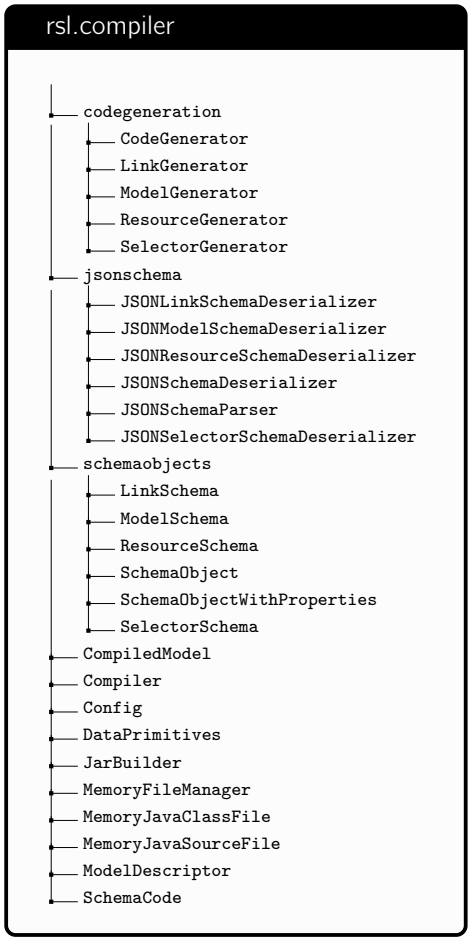
Can we contact you in case we have any follow-up questions related to your answers? If so, please provide a valid email address.

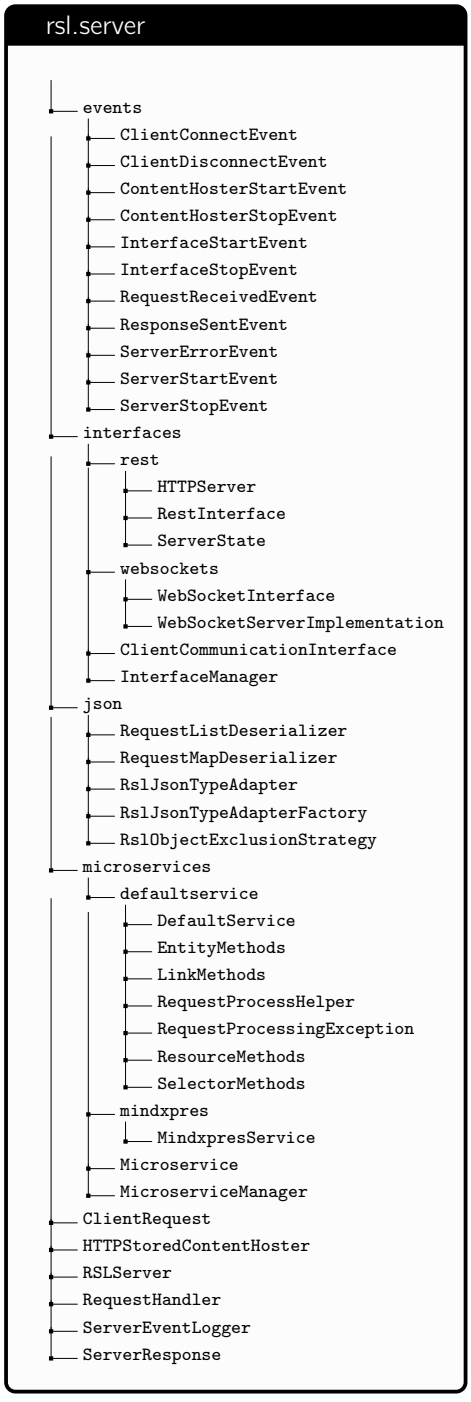
○ No ○ Yes, email:

Appendix D

RSL Implementation: Java Classes




rsl.compiler
codegeneration
CodeGenerator
LinkGenerator
ModelGenerator
ResourceGenerator
SelectorGenerator
jsonschema
JSONLinkSchemaDeserializer
JSONModelSchemaDeserializer
JSONResourceSchemaDeserializer
JSONSchemaDeserializer
JSONSchemaParser
JSONSelectorSchemaDeserializer
schemaobjects
LinkSchema
ModelSchema
ResourceSchema
SchemaObject
SchemaObjectWithProperties
SelectorSchema
CompiledModel
Compiler
Config
DataPrimitives
JarBuilder
MemoryFileManager
MemoryJavaClassFile
MemoryJavaSourceFile
ModelDescriptor
SchemaCode


rsl.server
events
ClientConnectEvent
ClientDisconnectEvent
ContentHosterStartEvent
ContentHosterStopEvent
InterfaceStartEvent
InterfaceStopEvent
RequestReceivedEvent
ResponseSentEvent
ServerErrorEvent
ServerStartEvent
ServerStopEvent
interfaces
rest
HTTPServer
RestInterface
ServerState
websockets
WebSocketInterface
WebSocketServerImplementation
ClientCommunicationInterface
InterfaceManager
json
RequestListDeserializer
RequestMapDeserializer
RslJsonTypeAdapter
RslJsonTypeAdapterFactory
RslObjectExclusionStrategy
microservices
defaultservice
DefaultService
EntityMethods
LinkMethods
RequestProcessHelper
RequestProcessingException
ResourceMethods
SelectorMethods
mindxpres
MindxpresService
Microservice
MicroserviceManager
ClientRequest
HTTPStoredContentHoster
RSLServer
RequestHandler
ServerEventLogger
ServerResponse

Appendix E

Used Tools and Technologies

As this dissertation is provided in pursuit of a PhD degree in Science, we feel that it is also important to demonstrate a certain level of technical proficiency. In this appendix we provide a list of the tools and technologies that were used to support this dissertation and its research artefacts.

This document was created in LaTeX. Statistical analysis was performed with R[1] and the related plots were scripted in RStudio[2] so that they could easily be reproduced if changes had to be made. We used Visual Paradigm[3] to create UML diagrams for the RSL hypermedia metamodel and our content model. Most of the other diagrams and figures were created with draw.io[4]. This dissertation as well as all research artefacts were managed under version control in SVN[5] or Git[6] repositories.

For gathering the data for our programmatic analysis of more than 12 000 PowerPoint documents we created a web crawler in Java 8. This allowed us to create a complete list of all presentations on a web-based presentation hosting service before randomly sampling them and downloading the selected PowerPoint files. For the analysis itself we created a small framework in Java 8. In order to interpret the PowerPoint documents we used Aspose Slides[7], and we used an SQLite[8] database to store extracted metadata. In order to later extract data and aggregate results across multiple tables we queried the database using some relatively complex SQL queries.

[1] https://www.r-project.org
[2] https://www.rstudio.com
[3] https://www.visual-paradigm.com
[4] https://www.draw.io
[5] https://subversion.apache.org
[6] https://git-scm.com
[7] https://products.aspose.com/slides/java
[8] https://www.sqlite.org

The implementation of our information system for next generation presentations was created in Java 8. We used Gradle[1] as a build tool, allowing us to easily manage dependencies and to allowing us to build the system on any platform. We also made heavy use of custom Gradle tasks, for instance for setting up test scenarios (e.g. databases) for our unit tests. The project involved various other libraries, the most notable being Google Guava[2], JavaPoet[3] and the ObjectDB[4] object database. As this list of technologies hardly reflects the complexity of the system we refer to Chapter 5 for more implementation details on our model-driven hypermedia link service.

Our presentation engine was created in web technologies such as HTML5, CSS3, JavaScript and WebSockets. The MindXpres compiler is based on the Node.js[5] platform and we make heavy use of the associated NPM package manager for dependency management as well as for custom build scripts. One notable dependency is Webpack[6] which helps the compiler to bundle plug-ins and other content into a portable presentation bundle. Individual MindXpres plug-ins use a variety of existing libraries to achieve their goals, but one that is used quite often is the D3[7] library for data visualisation.

Finally, the research performed as part of this dissertation has also provided us with some additional hardware experience. For instance, various plug-in solutions make use of embedded systems such as the Raspberry Pi or the Arduino platform. At various occasions we made use of the Microsoft Kinect and the related SDKs, for instance for detecting gestures. We have also experimented with using the Myo armband for interacting with presentations. Last but not least, we have written a device driver from scratch for an as of yet unreleased digital pen based on Anoto[8] technology.

[1] `https://gradle.org`
[2] `https://github.com/google/guava`
[3] `https://github.com/square/javapoet`
[4] `https://www.objectdb.com`
[5] `https://nodejs.org/en/`
[6] `https://webpack.js.org`
[7] `https://d3js.org`
[8] `http://www.anoto.com`

Part VII

Bibliography

References

[1] "*1484.12.1 IEEE LTSC Draft Standard for Learning Object Metadata*". http://grouper.ieee.org/groups/ltsc/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf. Accessed: 2018-03-27.

[2] "*Navy Integrated Learning Environment (Navy ILE) Introduction*". http://www.netc.navy.mil/ile/. Accessed: 2018-03-27.

[3] G. D. Abowd, 1999. "*Classroom 2000: An Experiment with the Instrumentation of a Living Educational Environment*". IBM Systems Journal, 38(4):508–530, 1999. https://doi.org/10.1147/sj.384.0508.

[4] C. Ahlberg, 1996. "*Spotfire: An Information Exploration Environment*". ACM SIGMOD Record, 25(4):25–29, 1996. https://doi.org/10.1145/245882.245893.

[5] B. A. Ahlborn, D. Thompson, O. Kreylos, and O. G. Staadt, 2005. "*A Practical System for Laser Pointer Interaction on Large Displays*". Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST 2005), pages 1–4, Monterey, California, USA. ACM. https://doi.org/10.1145/1101616.1101637.

[6] E. G. Aiken, G. S. Thomas, and W. A. Shennum, 1975. "*Memory for a Lecture: Effects of Notes, Lecture Rate, and Informational Density*". Journal of Educational Psychology, 67(3):439, 1975. https://doi.org/10.1037/h0076613.

[7] O. Akcay, M. H. Dalgin, and S. Bhatnagar, 2011. "*Perception of Color in Product Choice among College Students: A Cross-National Analysis of USA, India, China and Turkey*". International Journal of Business and Social Science, 2(21), 2011.

[8] M. Alley and K. A. Neeley, 2005. "*Rethinking the Design of Presentation Slides: A Case for Sentence Headlines and Visual Evidence*". Technical Communication, 52(4):417–426, 2005.

[9] R. Anderson, R. Anderson, B. Simon, S. A. Wolfman, T. VanDeGrift, and K. Yasuhara, 2004. "*Experiences With a Tablet PC Based Lecture Presentation System in Computer Science Courses*". ACM SIGCSE Bulletin, 36(1):56–60, 2004. https://doi.org/10.1145/971300.971323.

[10] R. Anderson, C. Hoyer, C. Prince, J. Su, F. Videon, and S. Wolfman, 2004. "*Speech, Ink, and Slides: The Interaction of Content Channels*". Proceedings of the 12th Annual ACM International Conference on Multimedia (Multimedia 2004), pages 796–803, New York, New York, USA. ACM. https://doi.org/10.1145/1027527.1027713.

[11] R. Anderson, C. Hoyer, C. Prince, J. Su, F. Videon, S. Wolfman, and R. Anderson, 2004. "*Understanding Diagrammatic Ink in Lecture*". Proceedings of the AAAI Fall Symposium Workshop on Making Pen-Based Interaction Intelligent and Natural (2004), Arlington, Virginia, USA. AAAI. ISBN 978-1-57735-217-4.

[12] R. J. Anderson, C. Hoyer, S. A. Wolfman, and R. Anderson, 2004. "*A Study of Digital Ink in Lecture Presentation*". Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2004), pages 567–574, Vienna, Austria. ACM. https://doi.org/10.1145/985692.985764.

[13] J. M. Apperson, E. L. Laws, and J. A. Scepansky, 2006. "*The Impact of Presentation Graphics on Students' Experience in the Classroom*". Computers & Education, 47(1):116–126, 2006. https://doi.org/10.1016/j.compedu.2004.09.003.

[14] R. Asadi, H. Trinh, H. J. Fell, and T. W. Bickmore, 2017. "*IntelliPrompter: Speech-based Dynamic Note Display Interface for Oral Presentations*". Proceedings of the 19th ACM International Conference on Multimodal Interaction (ICMI 2017), pages 172–180, Glasgow, UK. ACM. https://doi.org/10.1145/3136755.3136818.

[15] W. Atkins-Sayre, S. Hopkins, S. Mohundro, and W. Sayre, 1998. "*Rewards and Liabilities of Presentation Software as an Ancillary Tool: Prison or Paradise?*" Proceedings of the Annual Meeting of the National Communication Association, New York, New York, USA.

[16] M. Austin, 2010. Useful Fictions: Evolution, Anxiety, and the Origins of Literature. University of Nebraska Press, 2010. ISBN 978-0803230262.

[17] B. Avsar, D. E. Aliabadi, E. E. Aliabadi, and R. Yousefnezhad, 2016. "*Academic Presenter: A New Storytelling Presentation Software for Academic Purposes*". arXiv:1607.06979, https://arxiv.org/abs/1607.06979, 2016.

[18] R. Baecker, 1998. "*Sorting Out Sorting: A Case Study of Software Visualization for Teaching Computer Science*". Software visualization: Programming as a multimedia experience, 1:369–381, 1998. https://doi.org/10.1145/1597849.1384319.

[19] Y. Baeten, 2015. "*Intuitive Information Visualisation for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2015.

[20] T. Baltrušaitis, P. Robinson, and L.-P. Morency, 2016. "*Openface: An Open Source Facial Behavior Analysis Toolkit*". Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV 2016), pages 1–10, Lake Placid, New York, USA. IEEE. https://doi.org/10.1109/WACV.2016.7477553.

[21] A. Bangor, P. Kortum, and J. Miller, 2009. "*Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale*". Journal of Usability Studies, 4(3):114–123, 2009.

[22] P. M. Barnaghi and S. A. Kareem, 2006. "*A Flexible Architecture for Semantic Annotation and Automated Multimedia Presentation Generation*". SEMPS, 228, 2006.

[23] C. Barritt, D. Lewis, and W. Wieseler, 1999. "*Cisco Systems Reusable Information Object Strategy*". Definition, Creation Overview, and Guidelines Version 3.0, 1999.

[24] R. A. Bartsch and K. M. Cobern, 2003. "*Effectiveness of PowerPoint Presentations in Lectures*". Computers & Education, 41(1):77–86, 2003. https://doi.org/10.1016/S0360-1315(03)00027-7.

[25] L. Batrinca, G. Stratou, A. Shapiro, L.-P. Morency, and S. Scherer, 2013. "*Cicero - Towards a Multimodal Virtual. Audience Platform for Public Speaking Training*". International Workshop on Intelligent Virtual Agents, pages 116–128, Springer. https://doi.org/10.1007/978-3-642-40415-3_10.

[26] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis, 1998. Graph Drawing: Algorithms for the Visualization of Graphs. Pearson, 1998. ISBN 978-0133016154.

[27] B. B. Bederson, 2011. "*The Promise of Zoomable User Interfaces*". Behaviour & Information Technology, 30(6):853–866, 2011. https://doi.org/10.1145/1865841.1865844.

[28] B. B. Bederson and J. D. Hollan, 1994. "*Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics*". Proceedings of the 7th annual ACM symposium on User Interface Software and Technology (UIST 1994), pages 17–26, Marina del Rey, California, USA. ACM. https://doi.org/10.1145/192426.192435.

[29] S. D. Beets and P. G. Lobingier, 2001. "*Cyber Dimensions: Pedagogical Techniques: Student Performance and Preferences*". Journal of Education for business, 76(4):231–235, 2001. https://doi.org/10.1080/08832320109601316.

[30] J. Bennedsen and M. E. Caspersen, 2007. "*Failure Rates in Introductory Programming*". ACM SIGCSE Bulletin, 39(2):32–36, 2007. https://doi.org/10.1145/1272848.1272879.

[31] M. Berry and M. Kölling, 2013. "*The Design and Implementation of a Notional Machine for Teaching Introductory Programming*". Proceedings of the 8th Workshop in Primary and Secondary Computing Education (WiPSCE 2013), pages 25–28, Aarhus, Denmark. ACM. https://doi.org/10.1145/2532748.2532765.

[32] F. Biadsy, A. Rosenberg, R. Carlson, J. Hirschberg, and E. Strangert, 2008. "*A Cross-Cultural Comparison of American, Palestinian, and Swedish Perception of Charismatic Speech*". Speech Prosody 2008, Campinas, Brazil. https://doi.org/10.7916/D8M3342B.

[33] J. L. Bishop and M. A. Verleger, 2013. "*The Flipped Classroom: A Survey of the Research*". ASEE National Conference Proceedings, Vol. 30, No. 9, pages 1–18, Atlanta, Georgia, USA. ASEE.

[34] W. Blokzijl and B. Andeweg, 2007. "*The Effect of Text Slides Compared to Visualizations on Learning and Appreciation in Lectures*". Proceedings of the IEEE International Professional Communication Conference (IPCC 2007), pages 1–9, Seattle, Washington, USA. IEEE. https://doi.org/10.1109/IPCC.2007.4464074.

[35] W. Blokzijl and R. Naeff, 2004. "*The Instructor as Stagehand: Dutch Student Responses to PowerPoint*". Business Communication Quarterly, 67(1):70–77, 2004. https://doi.org/10.1177/1080569903262046.

[36] A. Boltman, A. Druin, B. Bederson, J. Hourcade, C. Fast, M. Kjellin, D. Stanton, C. O'Malley, S. Cobb, Y. Sunblad, et al., 2002. "*The Nature of Children's Storytelling With and Without Technology*". Proceedings of the American Educational Research Association Conference (AERA 2002), New Orleans, Louisiana, USA.

[37] C. C. Bonwell and J. A. Eison, 1991. Active Learning: Creating Excitement in the Classroom. 1991 ASHE-ERIC Higher Education Reports. ERIC, 1991.

[38] A. Borning and R. Duisberg, 1986. "*Constraint-Based Tools for Building User Interfaces*". ACM Transactions on Graphics (TOG), 5(4):345–374, 1986. https://doi.org/10.1145/27623.29354.

[39] D. Bos, 2015. "*Multimodal Interaction for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2015.

[40] J. Brand, 2004. "*Presentation (Mis)Management: Content and Collaboration Strategies*". Delta, 3057, 2004.

[41] U. Brandes, M. Eiglsperger, J. Lerner, and C. Pich, 2013. Graph Markup Language (GraphML), 2013.

[42] J. Brooke, 1996. "*SUS - A Quick and Dirty Usability Scale*". Usability Evaluation in Industry, 189(194):4–7, 1996.

[43] M. D. Bush and J. D. Mott, 2009. "*The Transformation of Learning with Technology: Learner-Centricity, Content and Tool Malleability, and Network Effects*". Educational Technology, pages 3–20, 2009.

[44] V. Bush, 1945. "*As We May Think*". The Atlantic Monthly, 176(1):101–108, 1945. https://doi.org/10.1145/227181.227186.

[45] J. J. Cañas, M. T. Bajo, and P. Gonzalvo, 1994. "*Mental Models and Computer Programming*". International Journal of Human Computer Studies, 40(5):795–812, 1994. https://doi.org/10.1006/ijhc.1994.1038.

[46] J. H. Canós, M. I. Marante, and M. Llavador, 2010. "*SliDL: A Slide Digital Library Supporting Content Reuse in Presentations*". Proceedings of the 14th International Conference on Theory and Practice of Digital Libraries (ECDL 2010), pages 453–456, Glasgow, UK. Springer. https://doi.org/10.1007/978-3-642-15464-5_55.

[47] X. Cao, E. Ofek, and D. Vronay, 2005. "*Evaluation of Alternative Presentation Control Techniques*". Extended Abstracts Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2005), pages 1248–1251, Portland, Oregon, USA. ACM. https://doi.org/10.1145/1056808.1056888.

[48] D. Chattopadhyay, K. O'Hara, S. Rintel, and R. Rädle, 2016. "*Office Social: Presentation Interactivity for Nearby Devices*". Proceedings of the 34th ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2016), pages 2487–2491, San Jose, California, USA. ACM. https://doi.org/10.1145/2858036.2858337.

[49] D. Chattopadhyay, F. Salvadori, K. O'hara, and S. Rintel, 2018. "*Beyond Presentation: Shared Slideware Control as a Resource for Collocated Collaboration*". Human-Computer Interaction, 33(5-6):455–498, 2018. https://doi.org/10.1080/07370024.2017.1388170.

[50] J.-C. Chen, W.-T. Chu, J.-H. Kuo, C.-Y. Weng, and J.-L. Wu, 2006. "*Tiling Slideshow*". Proceedings of the 14th ACM International Conference on Multimedia (MM 2006), pages 25–34, Santa Barbara, California, USA. ACM. https://doi.org/10.1145/1180639.1180653.

[51] P.-N. Chou, C.-C. Chang, and P.-F. Lu, 2015. "*Prezi Versus PowerPoint: The Effects of Varied Digital Presentation Tools on Students' Learning Performance*". Computers & Education, 91:73–82, 2015. https://doi.org/10.1016/j.compedu.2015.10.020.

[52] M. Christianson and S. Payne, 2011. "*Helping Students Develop Skills for Better Presentations: Using the 20x20 Format for Presentation Training*". ICU Language Research Bulletin, 26:1–15, 2011.

[53] C. S. Claxton and P. H. Murrell, 1987. Learning Styles: Implications for Improving Educational Practices. ASHE-ERIC Higher Education Report No. 4, 1987. ERIC, 1987.

[54] W. S. Cleveland, 1993. Visualizing Data. Hobart Press, 1993. ISBN 978-0963488404.

[55] W. S. Cleveland, 1994. The Elements of Graphing Data. Hobart Press, 2nd Edition, 1994. ISBN 978-0963488411.

[56] R. J. Craig and J. H. Amernic, 2006. "*PowerPoint Presentation Technology and the Dynamics of Teaching*". Innovative Higher Education, 31(3):147–160, 2006. https://doi.org/10.1007/s10755-006-9017-5.

[57] A. Cross, E. Cutrell, and W. Thies, 2012. "*Low-cost Audience Polling Using Computer Vision*". Proceedings of the 25th annual ACM symposium on User Interface Software and Technology (UIST 2012), pages 45–54, Cambridge, Massachusetts, USA. ACM. https://doi.org/10.1145/2380116.2380124.

[58] J. H. Cross, D. Hendrix, and D. A. Umphress, 2004. "*jGRASP: An Integrated Development Environment with Visualizations for Teaching Java in CS1, CS2, and Beyond*". Proceedings of the 34th Annual Frontiers in Education Conference (FIE 2004), pages 1466–1467, Savannah, Georgia, USA. IEEE. https://doi.org/10.1109/FIE.2006.322748.

[59] J. C. Cuthell, 2005. "*The Impact of Interactive Whiteboards on Teaching, Learning and Attainment*". Proceedings of the Society for Information Technology & Teacher Education International Conference (SITE 2005), pages 1353–1355, AACE.

[60] M. A. da Gama Malcher and M. Endler, 2008. "*A Context-Aware Collaborative Presentation System for Handhelds*". Simpósio Brasileiro de Sistemas Colaborativos (2008), pages 1–11, Vila Velha, Brazil. IEEE. https://doi.org/10.1109/SBSC.2008.13.

[61] I. Damian, C. S. S. Tan, T. Baur, J. Schöning, K. Luyten, and E. André, 2015. "*Augmenting Social Interactions: Realtime Behavioural Feedback Using Social Signal Processing Techniques*". Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI 2015), pages 565–574, Seoul, Korea. ACM. https://doi.org/10.1145/2702123.2702314.

[62] L. Daniels, 1999. "*Introducing Technology in the Classroom: PowerPoint as a First Step*". Journal of Computing in Higher Education, 10(2):42–56, 1999. https://doi.org/10.1007/BF02948722.

[63] W. Dann, S. Cooper, and R. Pausch, 2001. "*Using Visualization To Teach Novices Recursion*". ACM SIGCSE Bulletin, 33(3):109–112, 2001. https://doi.org/10.1145/507758.377507.

[64] H. Dawkes, L. A. Tweedie, and B. Spence, 1996. "*VICKI: the VIsualisation Construction KIt*". Proceedings of the Workshop on Advanced Visual Interfaces (AVI 1996, pages 257–259, Gubbio, Italy. ACM. https://doi.org/10.1145/948449.948490.

[65] P. De Bra, G.-J. Houben, and H. Wu, 1999. "*AHAM: A Dexter-Based Reference Model for Adaptive Hypermedia*". Proceedings of the 10th ACM Conference on Hypertext and Hypermedia (HYPERTEXT 1999), pages 147–156, Darmstadt, Germany. ACM. https://doi.org/10.1145/294469.294508.

[66] J. Debie, 2014. "*Guided Data Visualisation for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2014.

[67] T. Devezas, S. Nunes, and M. T. Rodríguez, 2015. "*MediaViz: An Interactive Visualization Platform for Online Media Studies*". Proceedings of the International Workshop on Human-centric Independent Computing (HIC 2015), pages 7–11, Guzelyurt, Northern Cyprus. ACM. https://doi.org/10.1145/2808469.2808474.

[68] R. B. Dewar and E. Schonberg, 2008. "*Computer Science Education: Where Are the Software Engineers of Tomorrow?*" Crosstalk: the Journal of Defense Software Engineering, 21(1):28–30, 2008.

[69] A. K. Dils, 2000. "*Using Technology in a Middle School Social Studies Classroom*". International Journal of Social Education, 15(1):102–12, 2000.

[70] A. Dix and G. Ellis, 1998. "*Starting Simple: Adding Value to Static Visualisation Through Simple Interaction*". Proceedings of the Working Conference on Advanced Visual Interfaces (AVI 1998), pages 124–134, L'Aquila, Italy. ACM. https://doi.org/10.1145/948496.948514.

[71] J.-L. Doumont, 2005. "*The Cognitive Style of PowerPoint: Slides Are Not All Evil*". Technical Communication, 52(1):64–71, 2005.

[72] J.-L. Doumont, 2009. Trees, Maps, and Theorems: Effective Communication for Rational Minds. Principiae, 2009.

[73] S. Downes, 2001. "*Learning Objects: Resources for Distance Education Worldwide*". The International Review of Research in Open and Distributed Learning, 2(1), 2001.

[74] S. M. Drucker, G. Petschnigg, and M. Agrawala, 2006. "*Comparing and Managing Multiple Versions of Slide Presentations*". Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology (UIST 2006), pages 47–56, Montreux, Switzerland. ACM. https://doi.org/10.1145/1166253.1166263.

[75] R. J. Dufresne, W. J. Gerace, W. J. Leonard, J. P. Mestre, and L. Wenk, 1996. "*Classtalk: A Classroom communication System for Active Learning*". Journal of Computing in Higher Education, 7(2):3–47, 1996. https://doi.org/10.1007/BF02948592.

[76] B. Dumas, B. Moerman, S. Trullemans, and B. Signer, 2014. "*ArtVis: Combining Advanced Visualisation and Tangible Interaction for the Exploration, Analysis and Browsing of Digital Artwork Collections*". Proceedings of the 12th International Working Conference on Advanced Visual Interfaces, AVI 2014, pages 65–72, Como, Italy. ACM. https://doi.org/10.1145/2598153.2598159.

[77] E. Duval and W. Hodgins, 2003. "*A LOM Research Agenda*". Proceedings of the Twelfth International World Wide Web Conference (WWW 2003), Budapest, Hungary.

[78] R. Eccles, T. Kapler, R. Harper, and W. Wright, 2008. "*Stories in GeoTime*". Information Visualization, 7(1):3–17, 2008. https://doi.org/10.1109/VAST.2007.4388992.

[79] D. Edge, S. Gulwani, N. Milic-Frayling, M. Raza, R. Adhitya Saputra, C. Wang, and K. Yatani, 2015. "*Mixed-Initiative Approaches to Global Editing in Slideware*". Proceedings of the 33rd ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2015), pages 3503–3512, Seoul, Korea. ACM. https://doi.org/10.1145/2702123.2702551.

[80] D. Edge, J. Savage, and K. Yatani, 2013. "*HyperSlides: Dynamic Presentation Prototyping*". Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2013), pages 671–680, Paris, France. ACM. https://doi.org/10.1145/2470654.2470749.

[81] D. Edge, X. Yang, Y. Kotturi, S. Wang, D. Feng, B. Lee, and S. Drucker, 2016. "*SlideSpace: Heuristic Design of a Hybrid Presentation Medium*". ACM Transactions on Computer-Human Interaction (TOCHI), 23(3):16, 2016. https://doi.org/10.1145/2898970.

[82] R. M. El Khoury and D. M. Mattar, 2012. "*PowerPoint in Accounting Classrooms: Constructive or Destructive?*" International Journal of Business and Social Science, 3(10):240–259, 2012.

[83] W. D. Ellis, 2013. A Source Book of Gestalt Psychology. Routledge, 2013. ISBN 978-0415864350.
[84] D. C. Engelbart, 1968. "*The Mother Of All Demos*", 1968.
[85] D. K. Farkas, 2006. "*Toward a Better Understanding of PowerPoint Deck Design*". Information Design Journal, 14(2):162–171, 2006. https://doi.org/10.1075/idj.14.2.08far.
[86] D. K. Farkas, 2008. "*A Heuristic for Reasoning About PowerPoint Deck Design*". Proceedings of the IEEE International Professional Communication Conference (ProComm 2008), Montreal, Canada. IEEE.
[87] E. Fernandes, H. Madhour, S. Miniaoui, and M. W. Forte, 2005. "*Phoenix Tool: A Support to Semantic Learning Model*". Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies (ICALT 2005), pages 948–949, Kaohsiung, Taiwan. IEEE. https://doi.org/10.1109/ICALT.2005.220.
[88] S. Few, 2009. Now You See It: Simple Visualization Techniques for Quantitative Analysis. Analytics Press, 2009. ISBN 978-0970601988.
[89] S. Few, 2009. "*Sometimes We Must Raise Our Voices*". Visual Business Intelligence Newsletter, 2009.
[90] S. Few, 2012. Show Me the Numbers: Designing Tables and Graphs to Enlighten. Analytics Press, 2012. ISBN 978-0970601995.
[91] M. H. Fischer, 2000. "*Do Irrelevant Depth Cues Affect the Comprehension of Bar Graphs?*" Applied Cognitive Psychology, 14(2):151–162, 2000. https://doi.org/10.1002/(SICI)1099-0720(200003/04)14:23.0.CO;2-Z.
[92] J. Fisler and S. Bleisch, 2006. "*eLML, the e-Lesson Markup Language: Developing Sustainable e-Learning Content Using an Open Source XML Framework*". Proceedings of the 2nd International Conference on Web Information Systems and Technologies (WEBIST 2006), pages 180–187, Setubal, Portugal. SciTePress. https://doi.org/10.5220/0001254701800187.
[93] M. Friendly and S. Institute, 2000. Visualizing Categorical Data. SAS Institute, 2000. ISBN 978-1580256605.
[94] D. Furcy, T. Naps, and J. Wentworth, 2008. "*Sorting Out Sorting: The Sequel*". ACM SIGCSE Bulletin, 40(3):174–178, 2008. https://doi.org/10.1145/1384271.1384319.
[95] Y. Gabriel, 2008. "*Against the Tyranny of PowerPoint: Technology-in-Use and Technology Abuse*". Organization Studies, 29(2):255–276, 2008. https://doi.org/10.1177/0170840607079536.
[96] T. Gan, Y. Wong, B. Mandal, V. Chandrasekhar, and M. S. Kankanhalli, 2015. "*Multi-sensor Self-Quantification of Presentations*". Proceedings of the 23rd ACM International Conference on Multimedia (MM 2015), pages 601–610, Brisbane, Australia. ACM. https://doi.org/10.1145/2733373.2806252.
[97] P. Garcia, 2004. "*Retooling PowerPoint for Hypermedia Authoring*". Proceedings of the Society for Information Technology & Teacher Education International Conference (SITE 2004), pages 4098–4099, AACE.
[98] J. Garner and M. Alley, 2013. "*How the Design of Presentation Slides Affects Audience Comprehension: A Case for the Assertion-Evidence Approach*". International Journal of Engineering Education, 29(6):1564–1579, 2013.
[99] R. Gaskins, 1984. "*Sample Product Proposal: Presentation Graphics for Overhead Projection*".
[100] R. Gaskins, 2007. "*PowerPoint at 20: Back to Basics*". Communications of the ACM, 50(12):15–17, 2007. https://doi.org/10.1145/1323688.1323710.
[101] C. E. George, 2000. "*Experiences with Novices: The Importance of Graphical Representations in Supporting Mental Models*". Proceedings of the 12th Annual Workshop of the Psychology of Programming Interest Group (PPIG 2000), pages 33–44, Cosenza, Italy.
[102] J. J. Gibson, 1962. "*Observations on Active Touch*". Psychological Review, 69(6):477–491, 1962. https://doi.org/10.1037/h0046962.
[103] S. Godin, 2004. "*Really Bad PowerPoint (and How to Avoid It). Do You Zoom, Inc*".

[104] E. Golub, 2004. "*Handwritten Slides on a TabletPC in a Discrete Mathematics Course*". Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2004), volume 36, pages 51–55, Norfolk, Virginia, USA. ACM. https://doi.org/10.1145/971300.971322.

[105] A. Gomes and A. J. Mendes, 2007. "*Learning to Program - Difficulties and Solutions*". Proceedings of the International Conference on Engineering Education (ICEE 2007), pages 283–287, Coimbra, Portugal. Taylor & Francis. https://doi.org/10.1080/03043790601135014.

[106] S. R. Gomez, R. Jianu, C. Ziemkiewicz, H. Guo, and D. Laidlaw, 2012. "*Different Strokes for Different Folks: Visual Presentation Design between Disciplines*". IEEE Transactions on Visualization and Computer Graphics, 18(12):2411–2420, 2012. https://doi.org/10.1109/TVCG.2012.214.

[107] L. Good and B. B. Bederson, 2002. "*Zoomable User Interfaces as a Medium for Slide Show Presentations*". Information Visualization, 1(1):35–49, 2002. https://doi.org/10.1057/palgrave.ivs.9500004.

[108] T. Goodison, 2002. "*ICT and Attainment at Primary Level*". British Journal of Educational Technology, 33(2):201–211, 2002. https://doi.org/10.1111/1467-8535.00253.

[109] A. Goodman, 2006. Why Bad Presentations Happen to Good Causes, and how to Ensure They Won't Happen to Yours. Cause Communications, 2006.

[110] W. H. Graf, 1998. "*Constraint-Based Graphical Layout of Multimodal Presentations*". Readings in Intelligent User Interfaces, pages 263–285, 1998.

[111] K. Grønbæk and R. H. Trigg, 1994. "*Design Issues for a Dexter-Based Hypermedia System*". Communications of the ACM, 37(2):40–50, 1994. https://doi.org/10.1145/175235.175238.

[112] A. G. Gross and J. E. Harmon, 2009. "*The Structure of PowerPoint Presentations: The Art of Grasping Things Whole*". IEEE Transactions on Professional Communication, 52(2):121–137, 2009. https://doi.org/10.1109/TPC.2009.2020889.

[113] P. Guo, 2014. "*Python Is Now the Most Popular Introductory Teaching Language at Top U.S. Universities*". BLOG CACM, https://cacm.acm.org/blogs/blog-cacm/176450, 2014.

[114] R. R. Hake, 1998. "*Interactive-Engagement Versus Traditional Methods: A Six-Thousand-Student Survey of Mechanics Test Data for Introductory Physics Courses*". American Journal of Physics, 66(1):64–74, 1998. https://doi.org/10.1119/1.18809.

[115] F. Halasz, M. Schwartz, K. Grønbæk, and R. H. Trigg, 1994. "*The Dexter Hypertext Reference Model*". Communications of the ACM, 37(2):30–39, 1994. https://doi.org/10.1145/175235.175237.

[116] H. Haller and A. Abecker, 2010. "*iMapping: A Zooming User Interface Approach for Personal and Semantic Knowledge Management*". Proceedings of the 21st ACM conference on Hypertext and Hypermedia (HT 2010), pages 119–128, Toronto, Canada. ACM. https://doi.org/10.1145/1810617.1810638.

[117] L. Hardman, D. C. Bulterman, and G. Van Rossum, 1994. "*The Amsterdam Hypermedia Model: Adding Time and Context to the Dexter Model*". Communications of the ACM, 37(2):50–62, 1994. https://doi.org/10.1145/175235.175239.

[118] J. Harris, 1990. "*Text Annotation and Underlining as Metacognitive Strategies to Improve Comprehension and Retention of Expository Text.*" page 15, 1990.

[119] S. Hasegawa, A. Tanida, and A. Kashihara, 2008. "*A Presentation Support Service Using Presentation Semantics*". Proceedings of the 8th IEEE International Conference on Advanced Learning Technologies (ICALT 2008), pages 104–105, Santander, Spain. IEEE. https://doi.org/10.1109/ICALT.2008.157.

[120] M. Hassib, S. Schneegass, P. Eiglsperger, N. Henze, A. Schmidt, and F. Alt, 2017. "*EngageMeter : A System for Implicit Audience Engagement Sensing Using Electroencephalography*". Proceedings of the 35th International Conference on Human Factors in Computing Systems (CHI 2017). https://doi.org/10.1145/3025453.3025669.

[121] J. Heymans, 2013. "*A WYSIWYG Template Authoring Solution for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2013.

[122] D. Hlynka and R. Mason, 1998. "*'PowerPoint' in the Classroom: What Is the Point?*" Educational Technology, 38(5):45–48, 1998.

[123] C. Hoellwarth and M. J. Moelter, 2011. "*The Implications of a Robust Curriculum in Introductory Mechanics*". American Journal of Physics, 79(5):540–545, 2011. https://doi.org/10.1119/1.3557069.

[124] M. Hofri, 1980. "*Two-dimensional Packing: Expected Performance of Simple Level Algorithms*". Information and Control, 45(1):1–17, 1980. https://doi.org/10.1016/S0019-9958(80)90817-7.

[125] A. Holzinger, M. D. Kickmeier-Rust, and D. Albert, 2008. "*Dynamic Media in Computer Science Education; Content Complexity and Learning Performance: Is Less More?*" Journal of Educational Technology & Society, 11(1):279–290, 2008.

[126] S. E. Hudson and S. P. Mohamed, 1990. "*Interactive Specification of Flexible User Interface Displays*". ACM Transactions on Information Systems (TOIS), 8(3):269–288, 1990. https://doi.org/10.1145/98188.98201.

[127] S. E. Hudson and I. Smith, 1996. "*Ultra-lightweight Constraints*". Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology (UIST 1996), pages 147–155, Seattle, Washington, USA. ACM. https://doi.org/10.1145/237091.237112.

[128] D. Huff, 1993. How to Lie with Statistics. WW Norton & Company, 1993. ISBN 978-0393310726.

[129] J. Hullman and N. Diakopoulos, 2011. "*Visualization Rhetoric: Framing Effects in Narrative Visualization*". IEEE Transactions on Visualization & Computer Graphics, 17(12):2231–2240, 2011. https://doi.org/10.1109/TVCG.2011.255.

[130] A. Hurlburt, 1982. The Grid: A Modular System for the Design and Production of Newpapers, Magazines, and Books. Wiley, 1982. ISBN 978-0471289234.

[131] N. Hurst, W. Li, and K. Marriott, 2009. "*Review of Automatic Document Formatting*". Proceedings of the 9th ACM Symposium on Document Engineering (DocEng 2009), pages 99–108, Munich, Germany. ACM. https://doi.org/10.1145/1600193.1600217.

[132] A. D. L. Initiative et al., 2004. "*Sharable Content Object Reference Model (SCORM)*". Advanced Distributed Learning, 2004.

[133] Y. Inoue-Smith, 2016. "*College-Based Case Studies in Using PowerPoint Effectively*". Cogent Education, 3(1):15, 2016. https://doi.org/10.1080/2331186X.2015.1127745.

[134] T. Jenkins, 2001. "*Teaching Programming – A Journey from Teacher to Motivator*". Proceedings of the 2nd Annual Conference of the LTSN Centre for Information and Computer Science, LTSN-ICS. ISBN 978-0954192709.

[135] T. Jenkins, 2001. "*The Motivation of Students of Programming*". Proceedings of the 6th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2001), pages 53–56, Canterbury, UK. ACM. https://doi.org/10.1145/377435.377472.

[136] T. Jenkins, 2002. "*On the Difficulty of Learning to Program*". Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences, pages 53–58, LTSN-ICS. ISBN 978-0954192716.

[137] E. Judson and D. Sawada, 2002. "*Learning from Past and Present: Electronic Response Systems in College Lecture Halls*". Journal of Computers in Mathematics and Science Teaching, 21(2):167–181, 2002.

[138] S. Kask, 2000. "*The Impact of Using Computer Presentations (CAP) on Student Learning in the Microeconomics Principles Course*". Meeting of the American Economic Association, Boston, Massachusetts, USA.

[139] H. Kassim and F. Ali, 2010. "*English Communicative Events and Skills Needed at the Workplace: Feedback from the Industry*". English for Specific Purposes, 29(3):168–182, 2010. https://doi.org/10.1016/j.esp.2009.10.002.

[140] Y. Kato, S. Shiramatsu, T. Ozono, and T. Shintani, 2014. "*Slide Repository: A Repository for Slide Reuse Using Research Episode Network*". Proceedings of the 3rd International Conference on Advanced Applied Informatics (IIAI-AAI 2014), pages 198–203, Kitakyushu, Japan. IEEE. https://doi.org/10.1109/IIAI-AAI.2014.50.

[141] R. H. Kay and A. LeSage, 2009. "*Examining the Benefits and Challenges of Using Audience Response Systems: A Review of the Literature*". Computers & Education, 53(3):819–827, 2009. https://doi.org/10.1016/j.compedu.2009.05.001.

[142] S. N. Keegan, 2007. "*Importance of Visual Images in Lectures: Case Study on Tourism Management Students*". Journal of Hospitality, Leisure, Sport & Tourism Education, 6(1):58–65, 2007. https://doi.org/10.3794/johlste.61.147.

[143] A. Kendon, 2004. Gesture: Visible Action as Utterance. Cambridge University Press, 2004. https://doi.org/10.1017/CBO9780511807572.

[144] S. Kennewell, H. Tanner, S. Jones, and G. Beauchamp, 2008. "*Analysing the Use of Interactive Technology to Implement Interactive Teaching*". Journal of Computer Assisted Learning, 24(1):61–73, 2008. https://doi.org/10.1111/j.1365-2729.2007.00244.x.

[145] A. Khalili, S. Auer, D. Tarasowa, and I. Ermilov, 2012. "*SlideWiki: Elicitation and Sharing of Corporate Knowledge Using Presentations*". Proceedings of the 18th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2012), pages 302–316, Galway, Ireland. Springer. https://doi.org/10.1007/978-3-642-33876-2_27.

[146] J. N. Khasnabish, M. Sodhi, J. Deshmukh, and G. Srinivasaraghavan, 2014. "*Detecting Programming Language from Source Code Using Bayesian Learning Techniques*". Workshop Proceedings of the 10th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM 2014), pages 513–522, St. Petersburg, Russia. Springer. https://doi.org/10.1007/978-3-319-08979-9_39.

[147] J. Kim, E. L. Glassman, A. Monroy-Hernández, and M. R. Morris, 2015. "*RIMES: Embedding Interactive Multimedia Exercises in Lecture Videos*". Proceedings of the 33rd ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2015), pages 1535–1544, Seoul, Korea. ACM. https://doi.org/10.1145/2702123.2702186.

[148] C. Kirstein and H. Muller, 1998. "*Interaction with a Projection Screen Using a Camera-Tracked Laser Pointer*". Proceedings of Multimedia Modeling (MMM 1998), pages 191–192, Lausanne, Switzerland. IEEE. https://doi.org/10.1109/MULMM.1998.723001.

[149] C. N. Klokmose, J. R. Eagan, S. Baader, W. Mackay, and M. Beaudouin-Lafon, 2015. "*Webstrates: Shareable Dynamic Media*". Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST 2015), pages 280–290, Charlotte, North Carolina, USA. ACM. https://doi.org/10.1145/2807442.2807446.

[150] H. Knoblauch, 2008. "*The Performance of Knowledge: Pointing and Knowledge in Powerpoint Presentations*". Cultural Sociology, 2(1):75–97, 2008. https://doi.org/10.1177/1749975507086275.

[151] S. Kochhar, 1990. "*Cooperative Computer-aided Design: A Paradigm of Automating the Design and Modeling of Graphical Objects*". Technical report, Harvard University, Cambridge, MA (United States).

[152] K. Koffka, 2014. Principles Of Gestalt Psychology. Mimesis International, 2014. ISBN 978-8857523934.

[153] W. Köhler, 1967. "*Gestalt Psychology*". Psychological Research, 31(1):18–30, 1967. https://doi.org/10.1007/BF00422382.

[154] A. Kohlhase, 2007. "*Semantic PowerPoint: Content and Semantic Technology for Educational Added-Value Services in MS PowerPoint*". Proceedings of the World Conference on Educational Media and Technology (EdMedia 2007), pages 3576–3583, Vancouver, Canada. AACE. ISBN 1880094622.

[155] J. Koomey and J. P. Holdren, 2008. Turning Numbers into Knowledge: Mastering the Art of Problem Solving. Analytics Press, 2nd Edition, 2008. ISBN 978-0970601926.

[156] S. Kopf, D. Schön, B. Guthier, R. Rietsche, and W. Effelsberg, 2015. "*A Real-time Feedback System for Presentation Skills*". Proceedings of the World Conference on Educational Media and Technology (EdMedia 2015), pages 1686–1693, Montréal, Canada. AACE.

[157] R. Kosara and J. Mackinlay, 2013. "*Storytelling: The Next Step for Visualization*". Computer, 46(5):44–50, 2013. https://doi.org/10.1109/MC.2013.36.

[158] S. M. Kosslyn, R. A. Kievit, A. G. Russell, and J. M. Shephard, 2012. "*PowerPoint® Presentation Flaws and Failures: A Psychological Analysis*". Frontiers in Psychology, 3:230, 2012. https://doi.org/10.3389/fpsyg.2012.00230.

[159] E. Kujansuu, 2005. "*Codewitz – Creating Learning Objects for Better Programming Skills*". Proceedings of E-Learn: World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education (E-Learn 2005), pages 93–97, Vancouver, Canada. AACE. ISBN 978-1-880094-57-0.

[160] A. Kumar and M. Sasikumar, 2007. "*PCML: A Pedagogy-Oriented Content Markup Language*". Innovations in E-Learning, Instruction Technology, Assessment, and Engineering Education, pages 13–17. Springer.

[161] K. Kurihara, M. Goto, J. Ogata, Y. Matsusaka, and T. Igarashi, 2007. "*Presentation Sensei: a Presentation Training System Using Speech and Image Processing*". Proceedings of the 9th International Conference on Multimodal Interfaces (ICMI 2007), pages 358–365, Nagoya, Japan. ACM. https://doi.org/10.1145/1322192.1322256.

[162] E. Lahtinen, K. Ala-Mutka, and H.-M. Järvinen, 2005. "*A Study of the Difficulties of Novice Programmers*". ACM SIGCSE Bulletin, 37(3):14–18, 2005. https://doi.org/10.1145/1067445.1067453.

[163] J. J. L'Allier, 1997. "*Frame of Reference: NETg's Map to the Products, their Structure and Core Beliefs*", 1997.

[164] J. Lanir, K. S. Booth, and A. Tang, 2008. "*MultiPresenter: A Presentation System for (Very) Large Display Spaces*". Proceedings of the 16th ACM international conference on Multimedia (MM 2008), pages 519–528, Vancouver, Canada. ACM. https://doi.org/10.1145/1459359.1459428.

[165] H. B. Lantz, 2004. Rubrics for Assessing Student Achievement in Science Grades K-12. Corwin, 2004. ISBN 978-0761931010.

[166] B. Lee, R. H. Kazi, and G. Smith, 2013. "*SketchStory: Telling More Engaging Stories with Data through Freeform Sketching*". IEEE Transactions on Visualization and Computer Graphics, 19(12):2416–2425, 2013. https://doi.org/10.1109/TVCG.2013.191.

[167] D. G. Levasseur and J. Kanan Sawyer, 2006. "*Pedagogy Meets PowerPoint: A Research Review of the Effects of Computer-Generated Slides in the Classroom*". Review of Communication, 6(1-2):101–123, 2006. https://doi.org/10.1080/15358590600763383.

[168] P. Levy, 2002. "*Interactive Whiteboards in Learning and Teaching in Two Sheffield Schools: A Developmental Study*", 2002.

[169] J. R. Lewis, 1995. "*IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use*". International Journal of Human-Computer Interaction, 7(1):57–78, 1995. https://doi.org/10.1080/10447319509526110.

[170] J. Li, Y. Wong, and M. S. Kankanhalli, 2016. "*Demo Paper: PreSense — An Assistive Presentation Self-Quantification System*". Proceedings of the IEEE International Symposium on Multimedia (ISM 2016), pages 401–402, San Jose, California, USA. IEEE. https://doi.org/10.1109/ISM.2016.0090.

[171] L. Lichtschlag, T. Karrer, and J. Borchers, 2009. "*Fly: A Tool to Author Planar Presentations*". Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2009), pages 547–556, Boston, Massachusetts, USA. ACM. https://doi.org/10.1145/1518701.1518786.

[172] L. Lichtschlag, P. Wacker, M. Ziefle, and J. Borchers, 2015. "*The Presenter Experience of Canvas Presentations*". Proceedings of the IFIP Conference on Human-Computer Interaction (INTERACT 2015), pages 289–297, Springer. https://doi.org/10.1007/978-3-319-22701-6.

[173] C. P. Lim, Y. Zhao, J. Tondeur, C. S. Chai, and C.-C. Tsai, 2013. "*Bridging the Gap: Technology Trends and Use of Technology in Schools*". Journal of Educational Technology & Society, 16(2):59–68, 2013.

[174] R. Linder, N. Lupfer, A. Kerne, A. M. Webb, C. Hill, Y. Qu, K. Keith, M. Carrasco, and E. Kellogg, 2015. "*Beyond Slideware: How a Free-form Presentation Medium Stimulates Free-form Thinking in the Classroom*". Proceedings of the ACM SIGCHI Conference on Creativity and Cognition Pages (C&C 2015), pages 285–294, Glasgow, UK. ACM. https://doi.org/10.1145/2757226.2757251.

[175] S. Little, J. Geurts, and J. Hunter, 2002. "*Dynamic Generation of Intelligent Multimedia Presentations through Semantic Inferencing*". Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2002), pages 158–175, Rome, Italy. Springer. https://doi.org/10.1007/3-540-45747-X_12.

[176] Y. Liu, D. Edge, and K. Yatani, 2013. "*SidePoint: A Peripheral Knowledge Panel for Presentation Slide Authoring*". Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2013), pages 681–684, Paris, France. ACM. https://doi.org/10.1145/2470654.2470750.

[177] Z. Liu and J. Stasko, 2010. "*Mental Models, Visual Reasoning and Interaction in Information Visualization: A Top-down Perspective*". IEEE Transactions on Visualization & Computer Graphics, 16(6):999–1008, 2010. https://doi.org/10.1109/TVCG.2010.177.

[178] A. Löser, C. Grune, and M. Hoffmann, 2002. "*A Didactic Model, Definition of Learning Objects and Selection of Metadata for an Online Curriculum*". Proceedings of the International Workshop of Interactive Computer Aided Learning (ICL 2002), Villach, Austria. Kassel University Press. ISBN 3-933146-83-6.

[179] R. B. Lowry, 1999. "*Electronic Presentation of Lectures–Effect upon Student Performance*". Journal of the Tertiary Education Group of the Royal Society of Chemistry, 1999.

[180] L. Ma, J. Ferguson, M. Roper, and M. Wood, 2007. "*Improving the Viability of Mental Models Held by Novice Programmers*". Proceedings of the 11th Workshop on Pedagogies and Tools for the Teaching and Learning of Object Oriented Concepts (ECOOP, TLOOC 2007), Springer-Verlag. https://doi.org/10.1145/1562877.1562931.

[181] J. Mackiewicz, 2007. "*Audience Perceptions of Fonts in Projected PowerPoint Text Slides*". Technical Communication, 54(3):295–307, 2007. https://doi.org/10.1109/IPCC.2006.320391.

[182] C. MacPherson, 2005. "*PowerPoint Slammed as Poor Teaching Tool*". On Campus News, 12(9), 2005.

[183] E. Mantei, 2000. "*Using Internet Class Notes and PowerPoint in the Physical Geology Lecture*". Journal of College Science Teaching, 29(5):301–305, 2000.

[184] U. Masao and H. Kôiti, 1999. "*Automatic Slide Presentation from Semantically Annotated Documents*". Proceedings of the Workshop on Coreference and its Applications (CorefApp 1999), pages 25–30, College Park, Maryland, USA. Association for Computational Linguistics. https://doi.org/10.3115/1608810.1608816.

[185] R. E. Mayer, 1981. "*The Psychology of How Novices Learn Computer Programming*". ACM Computing Surveys (CSUR), 13(1):121–141, 1981. https://doi.org/10.1145/356835.356841.

[186] R. E. Mayer and R. Moreno, 2002. "*Verbal Redundancy in Multimedia Learning: When Reading Helps Listening*". Journal of Educational Psychology, 94:156–163, 2002. https://doi.org/10.1037/0022-0663.94.1.156.

[187] M. McCracken, V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y. B.-D. Kolikant, C. Laxer, L. Thomas, I. Utting, and T. Wilusz, 2001. "*A Multi-National, Multi-Institutional Study of Assessment of Programming Skills of First-Year CS students*". Proceedings of the 6th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2001), pages 125–180, Canterbury, UK. ACM. https://doi.org/10.1145/572133.572137.

[188] Y. Mejova, K. De Schepper, L. Bergman, and J. Lu, 2011. "*Reuse in the Wild: an Empirical and Ethnographic Study of Organizational Content Reuse*". Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2011), pages 2877–2886, Vancouver, Canada. ACM. https://doi.org/10.1145/1978942.1979370.

[189] N. Mercer, S. Hennessy, and P. Warwick, 2010. "*Using Interactive Whiteboards to Orchestrate Classroom Dialogue*". Technology, Pedagogy and Education, 19(2):195–209, 2010. https://doi.org/10.1080/1475939X.2010.491230.

[190] P. Meștereagă, 2014. "*Intelligent Source Code Visualisation for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2014.

[191] G. A. Miller, 1956. "*The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information*". Psychological Review, 63(2):81, 1956. https://doi.org/10.1037/h0043158.

[192] I. Milne and G. Rowe, 2002. "*Difficulties in Learning and Teaching Programming — Views of Students and Tutors*". Education and Information technologies, 7(1):55–66, 2002. https://doi.org/10.1023/A:1015362608943.

[193] R. Miner, 2005. "*The Importance of MathML to Mathematics Communication*". Notices of the AMS, 52(5):532–538, 2005.

[194] A. Moreno, N. Myller, E. Sutinen, and M. Ben-Ari, 2004. "*Visualizing Programs with Jeliot 3*". Proceedings of the Working Conference on Advanced Visual Interfaces (AVI 2004), pages 373–376, Gallipoli, Italy. ACM. https://doi.org/10.1145/989863.989928.

[195] R. Moreno and R. E. Mayer, 2000. "*A Coherence Effect in Multimedia Learning: The Case for Minimizing Irrelevant Sounds in the Design of Multimedia Instructional Messages*". Journal of Educational psychology, 92(1):117–125, 2000. https://doi.org/10.1037/0022-0663.92.1.117.

[196] T. Moscovich, K. Scholz, J. F. Hughes, and D. Salesin, 2004. "*Customizable Presentations*". Technical Report CS-04-16, 2004.

[197] S. T. Moulton, S. Türkay, and S. M. Kosslyn, 2017. "*Does a Presentation's Medium Affect Its Message? PowerPoint, Prezi, and Oral Presentations*". PLOS ONE, 12(7), 2017. https://doi.org/10.1371/journal.pone.0178774.

[198] J. Muller-Brockmann, 1996. Grid Systems in Graphic Design : A Visual Communication Manual for Graphic Designers, Typographers and Three Dimensional Designers. Niggli Verlag, 1996. ISBN 978-3721201451.

[199] B. Myers, S. E. Hudson, and R. Pausch, 2000. "*Past, present, and future of user interface software tools*". ACM Transactions on Computer-Human Interaction (TOCHI), 7(1):3–28, 2000.

[200] B. A. Myers, D. Giuse, A. Mickish, B. Vander Zanden, D. Kosbie, R. McDaniel, J. Landay, M. Goldberg, and R. Pathasarathy, 1994. "*The Garnet User Interface Development Environment*". Proceedings of the Conference Companion on Human Factors in Computing Systems (CHI 1994), pages 457–458, Boston, Massachusetts, USA. ACM. https://doi.org/10.1145/259963.259990.

[201] L. Nelson, S. Ichimura, E. R. Pedersen, and L. Adams, 1999. "*Palette: A Paper Interface for Giving Presentations*". Proceedings of the ACM SIGCHI conference on Human Factors in Computing Systems (CHI 1999), pages 354–361, Pittsburgh, Pennsylvania, USA. ACM. https://doi.org/10.1145/302979.303109.

[202] T. H. Nelson, 1995. "*The Heart of Connection: Hypermedia Unifiedby Transclusion*". Communications of the ACM, 38(8):31–34, 1995. https://doi.org/10.1145/208344.208353.

[203] T. H. Nelson, 2008. Geeks Bearing Gifts: How the Computer World Got this Way. Mindful Press, 2008. ISBN 978-0578004389.

[204] A.-T. Nguyen, W. Chen, and M. Rauterberg, 2012. "*Feedback System for Presenters Detects Nonverbal Expressions*". SPIE Newsroom, 2012. https://doi.org/10.1117/2.1201212.004588.

[205] A.-T. Nguyen, W. Chen, and M. Rauterberg, 2012. "*Online Feedback System for Public Speakers*". Proceedings of the IEEE Symposium on E-Learning, E-Management and E-Services (IS3e 2012), pages 1–5, Kuala Lumpur, Malaysia. IEEE. https://doi.org/10.1109/IS3e.2012.6414963.

[206] A. Nikitina, 2011. Successful Public Speaking. Bookboon, 2011. ISBN 978-8776819477.

[207] M. Norrie and A. Wsurgler, 1997. "*OM Framework for Object-Oriented Data Management*". Object Data Models, page 24, 1997.

[208] P. Norvig, 2003. "*PowerPoint: Shot with its Own Bullets*". The Lancet, 362(9381):343–344, 2003. https://doi.org/10.1016/S0140-6736(03)14056-1.

[209] S. Olson and D. G. Riordan, 2012. "*Engage to Excel: Producing One Million Additional College Graduates with Degrees in Science, Technology, Engineering, and Mathematics*". Report to the President, 2012.

[210] K. Pardon, 2018. "*An Extensible Presentation Framework for Real-time Data Acquisition, Interaction and Augmented Reality Visualisations*".

[211] I. Parker, 2001. "*Absolute PowerPoint: Can a Software Package Edit Our Thoughts?*" The New Yorker, Annals of Business Section, pages 76–87, 2001.

[212] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, 2007. "*A Design Science Research Methodology for Information Systems Research*". Journal of Management Information Systems, 24(3):45–77, 2007. https://doi.org/10.2753/MIS0742-1222240302.

[213] K. Perlin and D. Fox, 1993. "*Pad: An Alternative Approach to the Computer Interface*". Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 1993), pages 57–64, Anaheim, California, USA. ACM. https://doi.org/10.1145/166117.166125.

[214] E. Popovich, 2006. "*Presentermouse Laser-pointer Tracking System*", 2006.

[215] M. Prince, 2004. "*Does Active Learning Work? A Review of the Research*". Journal of Engineering Education, 93(3):223–231, 2004. https://doi.org/10.1002/j.2168-9830.2004.tb00809.x.

[216] L. Pschetz, K. Yatani, and D. Edge, 2014. "*TurningPoint: Narrative-driven Presentation Planning*". Proceedings of the 32nd ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2014), pages 1591–1594, Toronto, Canada. ACM. https://doi.org/10.1145/2556288.2557389.

[217] D. Raggett, 2006. "*Slidy - A Web Based Alternative to Microsoft PowerPoint*". XTech, 2006.

[218] E. L. Rankin and D. J. Hoaas, 2001. "*The Use of PowerPoint and Student Performance*". Atlantic Economic Journal, 29(1):113–113, 2001. https://doi.org/10.1007/BF02299936.

[219] E. I. Reuss, B. Signer, and M. C. Norrie, 2008. "*PowerPoint Multimedia Presentations in Computer Science Education: What Do Users Need?*" Proceedings of the 4th Symposium of the Austrian HCI and Usability Engineering Group (USAB 2008), pages 281–298, Graz, Austria. Springer. https://doi.org/10.1007/978-3-540-89350-9_20.

[220] G. Robertson, R. Fernandez, D. Fisher, B. Lee, and J. Stasko, 2008. "*Effectiveness of Animation in Trend Visualization*". IEEE Transactions on Visualization and Computer Graphics, 14(6):1325–1332, 2008. https://doi.org/10.1109/TVCG.2008.125.

[221] R. Robson, 2004. "*Context and the Role of Standards in Increasing the Value of Learning Objects*". Online Education Using Learning Objects, pages 159–167, 2004.

[222] R. Roels, Y. Baeten, and B. Signer, 2016. "*An Interactive Data Visualisation Approach for Next Generation Presentation Tools*". Proceedings of the 8th International Conference on Computer Supported Education (CSEDU 2016), pages 123–133, Rome, Italy. SciTePress. https://doi.org/10.5220/0005806401230133.

[223] R. Roels, Y. Baeten, and B. Signer, 2016. "*Interactive and Narrative Data Visualisation for Presentation-based Knowledge Transfer*". Communications in Computer and Information Science (CCIS), 739, pages 237–258, Springer. https://doi.org/10.1007/978-3-319-63184-4_13.

[224] R. Roels, A. De Witte, and B. Signer, 2018. "*INFEX: A Unifying Framework for Cross-Device Information Exploration and Exchange*". Proceedings of the ACM on Human-Computer Interaction, 2(EICS):2, 2018. https://doi.org/10.1145/3179427.

[225] R. Roels, P. Meștereagă, and B. Signer, 2015. "*An Interactive Source Code Visualisation Plug-in for the MindXpres Presentation Platform*". Communications in Computer and Information Science (CCIS), 583, pages 169–188, Springer. https://doi.org/10.1007/978-3-319-29585-5_10.

[226] R. Roels, P. Meștereagă, and B. Signer, 2015. "*Towards Enhanced Presentation-based Teaching of Programming: An Interactive Source Code Visualisation Approach*". Proceedings of the 7th International Conference on Computer Supported Education (CSEDU 2015), pages 98–107, Lisbon, Portugal. SciTePress. https://doi.org/10.5220/0005445300980107.

[227] R. Roels and B. Signer, 2013. "*An Extensible Presentation Tool for Flexible Human-Information Interaction*". Demo Proceedings of the 27th International BCS Human Computer Interaction Conference (BCS-HCI 2013), pages 59–60, London, UK. British Computer Society.

[228] R. Roels and B. Signer, 2014. "*MindXpres: An Extensible Content-Driven Cross-Media Presentation Platform*". Proceedings of the 15th International Conference on Web Information Systems Engineering (WISE 2014), pages 215–230, Thessaloniki, Greece. Springer. https://doi.org/10.1007/978-3-319-11746-1_16.

[229] R. Roels and B. Signer, 2019. "*A Conceptual Framework and Content Model for Next Generation Presentation Solutions*". Proceedings of the ACM on Human-Computer Interaction, 3(EICS), 2019. https://doi.org/10.1145/3331149.

[230] R. Roels, C. Vermeylen, and B. Signer, 2014. "*A Unified Communication Platform for Enriching and Enhancing Presentations with Active Learning Components*". Proceedings of the 14th IEEE International Conference on Advanced Learning Technologies (ICALT 2014), pages 131–135, Athens, Greece. IEEE. https://doi.org/10.1109/ICALT.2014.46.

[231] J. Rogers and C. Philippe, 2014. "*The Tax Burden of Typical Workers in the EU 28*". New Direction, 17, 2014.

[232] A. Rosenberg and J. Hirschberg, 2005. "*Acoustic/Prosodic and Lexical Correlates of Charismatic Speech*". Proceedings of the 9th European Conference on Speech Communication and Technology (Eurospeech 2005), Lisbon, Portugal. ISCA.

[233] H. Rosling, 2006. "*Ted talk: Hans rosling shows the best stats you've ever seen*".

[234] S. Roth and C. Schulz, 2013. "*A Pattern-based Approach to Presentations using Slide Facilities*". Proceedings of the 18th European Conference on Pattern Languages of Program (EuroPLoP 2013), pages 15:1–15:12, Irsee, Germany. ACM. https://doi.org/10.1145/2739011.2739026.

[235] L. Sa and W.-J. Hsin, 2010. "*Traceable Recursion with Graphical Illustration for Novice Programmers*". InSight: A Journal of Scholarly Teaching, 5:54–62, 2010.

[236] B. Saket, S. Yang, H. Tan, K. Yatani, and D. Edge, 2014. "*TalkZones: Section-based Time Support for Presentations*". Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services (MobileHCI 2014), pages 263–272, Toronto, Canada. ACM. https://doi.org/10.1145/2628363.2628399.

[237] T. Samara, 2014. Design Elements: Understanding the Rules and Knowing When to Break Them. Rockport publishers, 2nd Edition, 2014. ISBN 978-1592539277.

[238] A. Savoy, R. W. Proctor, and G. Salvendy, 2009. "*Information Retention from PowerPoint™ and Traditional Lectures*". Computers & Education, 52(4):858–867, 2009. https://doi.org/10.1016/j.compedu.2008.12.005.

[239] G. Scagnetti and S. Kim, 2012. "*The Diagram of Information Visualization*". Parsons Journal for Information Mapping, 4(4):1–8, 2012.

[240] S. Schluep, M. Bettoni, and S. G. Schär, 2006. "*Modularization and Structured Markup for Learning Content in an Academic Environment*". International Journal on E-Learning, 5(1):35–44, 2006.

[241] A. Schmidt, 2005. "*Bridging the Gap Between Knowledge Management and E-Learning with Context-Aware Corporate Learning*". Proceedings of the Biennial Conference on Professional Knowledge Management/Wissensmanagement (WM 2005), pages 203–213, Kaiserslautern, Germany. Springer. https://doi.org/10.1007/11590019_23.

[242] J. Schneider, D. Börner, P. Van Rosmalen, and M. Specht, 2015. "*Presentation Trainer, Your Public Speaking Multimodal Coach*". Proceedings of the 17th ACM International Conference on Multimodal Interaction (ICMI 2015), pages 539–546, Seattle, Washington, USA. ACM. https://doi.org/10.1145/2818346.2830603.

[243] G. Schraw, 1998. "*Processing and Recall Differences among Seductive Details*". Journal of Educational Psychology, 90(1):3–12, 1998. https://doi.org/10.1037/0022-0663.90.1.3.

[244] E. Segel and J. Heer, 2010. "*Narrative Visualization: Telling Stories with Data*". IEEE Transactions on Visualization & Computer Graphics, 16(6):1139–1148, 2010. https://doi.org/10.1109/TVCG.2010.179.

[245] D. E. Shallcross and T. G. Harrison, 2007. "*Lectures: Electronic Presentations versus Chalk and Talk - A Chemist's View*". Chemistry Education Research and Practice, 8(1):73–79, 2007. https://doi.org/10.1039/B6RP90021F.

[246] M. Sharmin, L. Bergman, J. Lu, and R. Konuru, 2012. "*On Slide-based Contextual Cues for Presentation Reuse*". Proceedings of the ACM International Conference on Intelligent User Interfaces (IUI 2012), pages 129–138, Lisbon, Portugal. ACM. https://doi.org/10.1145/2166966.2166992.

[247] A. W. Siegman and S. Feldstein, 2014. Nonverbal Behavior and Communication. Psychology Press, 2nd Edition, 2014. ISBN 978-0805800180.

[248] M. Siegrist, 1996. "*The Use or Misuse of Three-dimensional Graphs to Represent Lower-dimensional Data*". Behaviour & Information Technology, 15(2):96–100, 1996. https://doi.org/10.1080/014492996120300.

[249] B. Signer, 2017. Fundamental Concepts for Interactive Paper and Cross-Media Information Spaces. Books on Demand, 2017. ISBN 978-3837027136.

[250] B. Signer and M. C. Norrie, 2007. "*As We May Link: A General Metamodel for Hypermedia Systems*". Proceedings of the 26th International Conference on Conceptual Modeling (ER 2007), pages 359–374, Auckland, New Zealand. Springer. https://doi.org/10.1007/978-3-540-75563-0_25.

[251] B. Signer and M. C. Norrie, 2007. "*PaperPoint: A Paper-Based Presentation and Interactive Paper Prototyping Tool*". Proceedings of the 1st international conference on Tangible and Embedded Interaction (TEI 2007), pages 57–64, Baton Rouge, Louisiana, USA. ACM. https://doi.org/10.1145/1226969.1226981.

[252] H. Siirtola, 2007. Interactive Visualization of Multidimensional Data. Tampere University Press, 2007. ISBN 978-9514469398.

[253] T. Simons, 2005. "*Does PowerPoint Make You Stupid?*" Presentations Magazine, 1(8):3, 2005.

[254] H. J. Smith, S. Higgins, K. Wall, and J. Miller, 2005. "*Interactive Whiteboards: Boon or Bandwagon? A Critical Review of the Literature*". Journal of Computer Assisted Learning, 21(2):91–101, 2005. https://doi.org/10.1111/j.1365-2729.2005.00117.x.

[255] P. A. Smith and G. I. Webb, 1995. "*Reinforcing a Generic Computer Model for Novice Programmers*". ASCILITE'95, 1995.

[256] K. Soni, 2017. "*An Extensible Virtual Assistant for the PowerPoint Presentation Tool*".

[257] J. Sorva and T. Sirkiä, 2010. "*UUhistle: A Software Tool for Visual Program Simulation*". Proceedings of the 10th Koli Calling International Conference on Computing Education Research (Koli Calling 2010), pages 49–54, Koli, Finland. ACM. https://doi.org/10.1145/1930464.1930471.

[258] R. Spence, 2007. Information Visualization: Design for Interaction. Pearson, 2nd Edition, 2007. ISBN 978-0132065504.

[259] R. Spicer, Y.-R. Lin, A. Kelliher, and H. Sundaram, 2012. "*NextSlidePlease: Authoring and Delivering Agile Multimedia Presentations*". ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 8(4):53, 2012. https://doi.org/10.1145/2379790.2379795.

[260] B. Stevens, 2005. "*What Communication Skills do Employers Want? Silicon Valley Recruiters Respond*". Journal of Employment Counseling, 42(1):2–9, 2005. https://doi.org/10.1002/j.2161-1920.2005.tb00893.x.

[261] E. Strangert and J. Gustafson, 2008. "*What Makes a Good Speaker? Subject Ratings, Acoustic Measurements and Perceptual Evaluations*". Proceedings of the 9th Annual Conference of the International Speech Communication Association (INTERSPEECH 2008), pages 1688–1691, Brisbane, Australia. ISCA.

[262] C. Süßand B. Freitag, 2002. "*LMML - The Learning Material Markup Language Framework*". Proceedings of the International Workshop of Interactive Computer Aided Learning (ICL 2002), Villach, Austria. Kassel University Press. ISBN 3-933146-83-6.

[263] C. Süß, R. Kammerl, and B. Freitag. "*A Teachware Management Framework for Multiple Teaching Strategies*". Proceedings of the 12th World Conference on Educational Multimedia, Hypermedia &, year = 2000, pages = 1101–1106, doi = , address = Montréal, Canada, publisher = AACE, Telecommunications (ED-MEDIA 2000). ISBN 1-880094-40-7.

[264] J. E. Susskind, 2005. "*Powerpoint's Power in the Classroom: Enhancing Students' Self-Efficacy and Attitudes*". Computers & Education, 45(2):203–215, 2005. https://doi.org/10.1016/j.compedu.2004.07.005.

[265] J. E. Susskind, 2008. "*Limits of PowerPoint's Power: Enhancing Students' Self-Efficacy and Attitudes but Not Their Behavior*". Computers & Education, 50(4):1228–1239, 2008. https://doi.org/10.1016/j.compedu.2006.12.001.

[266] D. F. Swayne, D. T. Lang, A. Buja, and D. Cook, 2003. "*GGobi: Evolving from XGobi into an Extensible Framework for Interactive Data Visualization*". Computational Statistics & Data Analysis, 43(4):423–444, 2003. https://doi.org/10.1016/S0167-9473(02)00286-4.

[267] A. Szabo and N. Hastings, 2000. "*Using IT in the Undergraduate Classroom: Should We Replace the Blackboard with PowerPoint?*" Computers & Education, 35(3):175–187, 2000. https://doi.org/10.1016/S0360-1315(00)00030-0.

[268] M. I. Tanveer, E. Lin, and M. E. Hoque, 2015. "*Rhema: A Real-Time In-Situ Intelligent Interface to Help People with Public Speaking*". Proceedings of the 20th International Conference on Intelligent User Interfaces (IUI 2015), pages 286–295, Atlanta, Georgia, USA. ACM. https://doi.org/10.1145/2678025.2701386.

[269] J. Teevan, D. Liebling, A. Paradiso, C. Garcia Jurado Suarez, C. von Veh, and D. Gehring, 2012. "*Displaying Mobile Feedback During a Presentation*". Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services, pages 379–382, San Francisco, California, USA. ACM. https://doi.org/10.1145/2371574.2371633.

[270] M. T. Thielsch and I. Perabo, 2012. "*Use and Evaluation of Presentation Software*". Technical Communication, 59(2):112–123, 2012.

[271] A. Thomas, 2003. "*Little Touches That Spell Success*". Times Educational Supplement, 23:56–67, 2003.

[272] M. Thorogood, 2016. "*slideDeck.js: A Platform for Generating Accessible and Interactive Web-Based Course Content*". Proceedings of the 21st Western Canadian Conference on Computing Education (WCCCE 2016), pages 13:1–13:5, Kamloops, Canada. ACM. https://doi.org/10.1145/2910925.2910941.

[273] J. Tidwell, 2010. Designing Interfaces: Patterns for Effective Interaction Design. O'Reilly Media, 2nd Edition, 2010. ISBN 978-1449379704.

[274] J. Tondeur, E. De Bruyne, M. Van den Driessche, S. McKenney, and D. Zandvliet, 2015. "*The Physical Placement of Classroom Technology and its Influences on Educational Practices*". Cambridge Journal of Education, 45(4):537–556, 2015. https://doi.org/10.1080/0305764X.2014.998624.

[275] H. Trinh, L. Ring, and T. Bickmore, 2015. "*DynamicDuo: Co-presenting with Virtual Agents*". Proceedings of the 33rd ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2015), pages 1739–1748, Seoul,Korea. ACM. https://doi.org/10.1145/2702123.2702584.

[276] H. Trinh, K. Yatani, and D. Edge, 2014. "*PitchPerfect: Integrated Rehearsal Environment for Structured Presentation Preparation*". Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2014), pages 1571–1580, Toronto, Canada. ACM. https://doi.org/10.1145/2556288.2557286.

[277] E. Tufte and P. Graves-Morris, 2014. "*The visual display of quantitative information.; 1983*".

[278] E. R. Tufte, 2006. The Cognitive Style of PowerPoint: Pitching Out Corrupts Within. Graphics Press, 2nd Edition, 2006. ISBN 978-0961392161.

[279] S. Ullman, 2000. High-Level Vision: Object Recognition and Visual Cognition. A Bradford Book, 2000. ISBN 978-0262710077.

[280] G. Undrill and F. McMaster, 2013. "*PowerPoint: Avoiding the Slide to Damnation*". Advances in psychiatric treatment, 19(1):14–22, 2013. https://doi.org/10.1192/apt.bp.110.008805.

[281] K. van Gyseghem, 2013. "*Dynamic Graph Visualisations for the MindXpres Presentation Tool*". Bachelor's Thesis. Vrije Universiteit Brussel, 2013.

[282] D. Vanacken, K. Luyten, K. Coninx, and M. Cardinaels, 2011. "*The Organic Presentation Tool*". Proceedings of Create 11: The Interaction Design Symposium (CREATE), London, UK.

[283] J. Vandermeersch, 2015. "*Content Migration and Layout for the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2015.

[284] J. Á. Velázquez-Iturbide and A. Pérez-Carrasco, 2010. "*InfoVis Interaction Techniques in Animation of Recursive Programs*". Algorithms, 3(1):76–91, 2010. https://doi.org/10.3390/a3010076.

[285] K. Verbert and E. Duval, 2008. "*ALOCOM: A Generic Content Model for Learning Objects*". International Journal on Digital Libraries, 9(1):41–63, 2008. https://doi.org/10.1007/s00799-008-0039-8.

[286] K. Verbert, X. Ochoa, and E. Duval, 2008. "*The ALOCOM Framework: Towards Scalable Content Reuse*". Journal of Digital Information, 9(1):1–24, 2008.

[287] J. Vereyken, 2016. "*Investigating Graph-based Storage Backends for Link Services*". Master's Thesis. Vrije Universiteit Brussel, 2016.

[288] C. Vermeylen, 2013. "*Audience-driven Presentations based on the MindXpres Presentation Tool*". Master's Thesis. Vrije Universiteit Brussel, 2013.

[289] F. B. Viegas, M. Wattenberg, F. Van Ham, J. Kriss, and M. McKeon, 2007. "*ManyEyes: A Site for Visualization at Internet Scale*". IEEE Transactions on Visualization & Computer Graphics, 13(6):1121–1128, 2007. https://doi.org/10.1109/TVCG.2007.70577.

[290] A. T. Virtanen, E. Lahtinen, and H.-M. Järvinen, 2005. "*VIP, A Visual Interpreter for Learning Introductory Programming with C++*". Proceedings of the 5th Koli Calling Conference on Computer Science Education (Koli Calling 2005), pages 125–130, Koli, Finland.

[291] F. Vogt, J. Wong, S. Fels, and D. Cavens, 2003. "*Tracking Multiple Laser Pointers for Large Screen Interaction*". Adjunct Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology (UIST 2003), pages 95–96, Vancouver, Canada. ACM. ISBN 1-58113-636-6.

[292] E. Wagner, 2002. "*Steps to Creating a Content Strategy for Your Organization*". Best of the, page 103, 2002.

[293] C. Ware, 2012. Information Visualization: Perception for Design. Morgan Kaufmann, 2012. ISBN 978-0123814647.

[294] J. Webster and H. Ho, 1997. "*Audience Engagement in Multimedia Presentations*". ACM SIGMIS Database, 28(2):63–77, 1997. https://doi.org/10.1145/264701.264706.

[295] L. Weitzman and K. Wittenburg, 1994. "*Automatic Presentation of Multimedia Documents Using Relational Grammars*". Proceedings of the 2nd ACM international Conference on Multimedia (MULTIMEDIA 1994), pages 443–451, San Francisco, California, USA. ACM. https://doi.org/10.1145/192593.192718.

[296] M. Wertheimer, 1923. "*Untersuchungen zur Lehre von der Gestalt. II*". Psychologische Forschung, 4(1):301–350, 1923. https://doi.org/10.1007/BF00410640.

[297] S. Williams and S. Steinweg, 2009. "*Use of Tablet PCs in Teacher Education and K-12 Settings*". Proceedings of the Society for Information Technology & Teacher Education International Conference (SITE 2009), pages 2189–2192, AACE.

[298] T. Wörtwein, M. Chollet, B. Schauerte, L.-P. Morency, R. Stiefelhagen, and S. Scherer, 2015. "*Multimodal Public Speaking Performance Assessment*". Proceedings of the 17th ACM International Conference on Multimodal Interaction (ICMI 2015), pages 43–50, Seattle, Washington, USA. ACM. https://doi.org/10.1145/2818346.2820762.

[299] C. Yan, 2015. "*TabletPoint: A Pen and Tablet-based Presentation Solution*". Master's Thesis. Vrije Universiteit Brussel, 2015.

[300] X. Yang, T. Mei, Y.-Q. Xu, Y. Rui, and S. Li, 2016. "*Automatic Generation of Visual-Textual Presentation Layout*". ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 12(2):33, 2016. https://doi.org/10.1145/2818709.

[301] N. M. Yassin, 2005. "*Learning Object Content Models: A Comparative Analysis*". Proceedings of the 1st Postgraduate Annual Research Seminar (PARS 2005), pages 87–91, Johor, Malaysia. FSKSM, UTM.

[302] J. S. Yi, Y. ah Kang, J. T. Stasko, J. A. Jacko, et al., 2007. "*Toward a Deeper Understanding of the Role of Interaction in Information Visualization*". IEEE Transactions on Visualization & Computer Graphics, 13(6):1224–1231, 2007. https://doi.org/10.1109/TVCG.2007.70515.

[303] J. R. Young, 2004. "*When Good Technology Means Bad Teaching*". The Chronicle of Higher Education, 51(12):A31–A37, 2004.

[304] M. Zahir, 2016. "*An Extensible Platform for Real-Time Feedback in Presentation Training*". Master's Thesis. Vrije Universiteit Brussel, 2016.

[305] B. V. Zanden and B. A. Myers, 1990. "*Automatic, Look-and-feel Independent Dialog Creation for Graphical User Interfaces*". Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 1990, pages 27–34, Seattle, Washington, USA. ACM. https://doi.org/10.1145/97243.97248.

[306] T. Zarraonandia, I. Aedo, P. Díaz, and A. Montero, 2013. "*An Augmented Lecture Feedback System for Supporting Learner and Teacher Communication*". British Journal of Educational Technology, 44(4):616–628, 2013. https://doi.org/10.1111/bjet.12047.

[307] S. Zevin and C. Holzem, 2017. "*Machine Learning Based Source Code Classification Using Syntax Oriented Features*". arXiv:1703.07638, https://arxiv.org/abs/1703.07638, 2017.

[308] J. Zhang, a. Xiao, T. Watanabe, Y. Ishikawa, and a. and, 2015. "*Detecting Reused Elements in Presentation Slides*". International Journal of Knowledge Engineering, 1(2):154–159, 2015. https://doi.org/10.7763/IJKE.2015.V1.27.

[309] J. Zhang, C. Xiao, T. Watanabe, and Y. Ishikawa, 2014. "*Content-Based Element Search for Presentation Slide Reuse*". IEICE TRANSACTIONS on Information and Systems, 97(10):2685–2696, 2014. https://doi.org/10.1587/transinf.2014EDP7023.

[310] M. X. Zhou and S. Ma, 2000. "*Toward Applying Machine Learning to Design Rule Acquisition for Automated Graphics Generation*". Proceedings of the AAAI Spring Symposium on Smart Graphics (2000), pages 16–23, Palo Alto, California, USA. AAAI.

[311] D. E. Zongker and D. H. Salesin, 2003. "*On Creating Animated Presentations*". Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation (SCA 2003), pages 298–308, San Diego, California, USA. Eurographics Association. ISBN 1-58113-659-5.