Vrije Universiteit Brussel

Faculty of Science & Bio-Engineering Sciences
Department of Computer Science
Web and Information Systems Engineering Lab

A Usability Evaluation Framework for Web Mashup Makers for End-Users

Thesis submitted in fulfilment of the requirements for the award of the degree of Doctor in Computer Science

Wael Al Sarraj

Academic year 2011 - 2012

Promotor: Prof. Dr. Olga De Troyer

Abstract

Currently, more than two billions people access the Web for various purposes. The majority are people without programming or modelling background. Part of these people (called end-users) also likes to create their own Web applications to meet their daily needs. Mashup Makers are tools to create such end-user's Web applications. As such, Mashup Makers could become the dominant environment for end-user development of Web applications. Existing Mashup Makers promise that creating a Web Mashup is very easy and just a matter of a few mouse clicks. However, there is no evidence that this is indeed the case. On the contrary, research has already revealed usability problems with Mashup Makers.

Therefore, this thesis concentrates on the usability of Mashup Makers as development environments for Web applications for end-users. Usability is a key issue for the success of software artifacts, and especially if the artifacts are intended for non-technical users. Therefore, we target the achievement of a consolidated approach, model, and framework for the evaluation of the usability of Mashup Makers for end-users. Such a framework will not only allow evaluating the usability of existing Mashup Makers, but it will also provide key issues concerning usability (i.e. usability impact factors) that developers of Mashup Makers and of other future end-user development tools can take into consideration when developing new tools.

To come to such a framework, first two initial experimental studies, a pilot study and a user experiment, have been performed. These experiments revealed that existing usability problems could be the basis for deriving usability impact factors and afterwards deriving a conceptual evaluation model and evaluation framework.

Both the pilot study and the user experiment were designed to evaluate a variety of Mashup Makers from different usability perspectives. The literature investigation of the usability of Mashup Makers, as well as the results (findings) of both experiments suggested that the usability of Mashup Makers for end-users is affected by three main aspects: the user interface aspect, the functional aspect, and the user interaction aspect. This suggests that evaluating the usability of Mashup Makers should depend on the evaluation of those three main aspects, resulting into three types of impact factors (indicators). Those impact factors were refined using more detailed evaluation criteria and subsequently the criteria are refined using metrics that link to raw usability data.

A conceptual model of usability factors of Mashup Makers has been developed. This conceptual model reflects the conceptual approach taken and identifies the main aspects (indicators) of the usability evaluation of Mashup Makers for end-users. Furthermore, a usability evaluation framework has also been devised. As already indicated, this usability evaluation framework can be used to guide usability practitioners in the evaluation of Mashup Makers, as well as designers of new end-user tools. Experts in the domain have evaluated the proposed framework using an experimental study.

Dedication

This PhD thesis is dedicated to

The spirit of my father

My lovely mother

My beloved wife,

My beloved kids,

My lovely sisters and brothers,

And the rest of my family members and friends.

Acknowledgements

I owe gratitude to all professors, colleagues, and friends who helped in making this PhD research study possible. I would particularly like to thank the following individuals:

***Prof. Dr. Olga De Troyer**, my promoter and director of Laboratory for Web and Information Systems, for her great support, supervision and guidance during the whole research period; she was always there when I needed her. This PhD thesis would have been very difficult to produce without her help and expertise in the matter.*

I am indebted to the members of the Jury: Prof. Dr. Khaldoun Zriek (Université Paris 8, France), Prof. Dr. Philippe Thiran (Université de Namur, Belgium), Prof. Dr. Olga De Troyer (Vrije Universiteit Brussel, Belgium), Prof. Dr. Beat Signer (Vrije Universiteit Brussel, Belgium) Prof. Dr. Bernard Manderick (Vrije Universiteit Brussel, Belgium), and Prof. Dr. Eddy Vandijck (Vrije Universiteit Brussel, Belgium), for their acceptance to be members of the jury of my PhD thesis and for their critical review and helpful suggestions, which resulted in the accomplishment of this PhD thesis.

Special thanks go to all my colleagues and staff in the Laboratory for Web and Information Systems Engineering at Vrije Universiteit Brussel VUB for their valued input to this thesis.

I would like to express my heartfelt appreciation to Mr Steven Konkel (BTC administrative officer for Palestinian students) for his assistance and cooperation during the period of my research.

Great thanks go to my wife, my great parents (my mother and the spirit of my father), my sisters, my brothers and friends for their continuous support and encouragement during the research period; I am very grateful to all of you.

Finally, I highly appreciate the Belgian People and Government, Belgian Directorate General for Development Cooperation (DGCD) and Belgian Technical Corporation (BTC) for their help and financial support.

Thank You All Very Much

Wael

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

Glossary & Acronyms

API Application Programming Interface

Atom Atom Syndication Format, an XML format used for Web feeds and they are formats for publishing Web-based content in a manner consumable by special applications termed “feed readers.”

Browsing enrichment Improving browsing processes and environments with extra functionality.

Casual end-user A person who is a non-programmer and who has no background in the field of computer application development and/or modelling.

CD’s framework Cognitive Dimensions of Notation Framework for usability evaluation of visual programming languages.

Conceptual model A description of a portion of the ‘real word’ that is of interest in a particular application domain.

EAI Enterprise Application Integration is the use of software and computer systems architectural principles to integrate a set of enterprise computer applications.

Empirical study Experimental study performed to investigate the usability of Mashup makers.

End-user The person who uses a product; the consumer. An end user of a computer system is someone who operates the computer, as opposed to the developer of the system who creates new functions for end users.

EUD End User Development

Evaluator A person who is either a usability practitioner or a Mashup maker designer.

Faceted browsing Faceted browsing is also called “Faceted navigation' which gives the users the ability to find items based on more than one dimension, to see breakdowns and projections of the items along different axis, which helps users gather insights about the data they are exploring.

GUI Graphical User Interface

HCI Human Computer Interaction

IDE Integrated Development Environment

JSON JavaScript Object Notation is a lightweight data-interchange format

Mashup A web application that integrates, uses, and combines data, presentation or functionality from two or more sources to create new services.

Mashup makers Tools to create (end-user's) Web Mashup applications

Mashup maker approach The method or combination of methods used in a Mashup maker to create a Mashup application by casual the end-user.

MUEF Mashup Maker Usability Evaluation Framework for end-users.

Observer A person who manage and supervise the usability evaluation process.

Pilot study A preliminary study performed to determine the potential of a larger and more in-depth survey of the same subject matter.

QUIM Quality In use Model

REST Representational State Transfer defines a set of architectural principles by which Web developer can design Web services that focus on a system's resources.

RSS Rich Site Summary or Syndication is a format for delivering regularly changing Web content.

SPSS A computer program used for survey authoring and deployment (IBM SPSS Data Collection), data mining (IBM SPSS Modeler), text analytics, statistical analysis, and collaboration and deployment (batch and automated scoring services).

T-Test Assesses whether the means of two groups are *statistically* different from each other. This analysis is appropriate whenever you want to compare the means of two groups

Usability Indicator Abstract conceptual construct for indicating an aspect of the usability of a system that cannot directly be measured but aims to connect observable and measurable usability criteria.

UI User interface

Usability evaluation factor An entity, resource or a unit of information which

refer to or provide meaning of an evaluation of the usability of certain object

Usability criteria A usability evaluation factor that can be directly measured through at least on specific usability metric.

Usability metric A function (in MUEF a question or a statement) whose inputs are usability data and whose output is a single numerical value that can be interpreted as the degree to which the Mashup maker pocesses a given attribute that affects its usability.

Usability evaluation methods A set of methods used to evaluate the usability of the human computer interface provided by a product/system.

Usability quantification Presenting usability evaluation factors by calculated quantitative means

Web 2.0 Web applications that facilitate participatory information sharing, interoperability, and collaboration on the World Wide Web.

Web service A method of communication between two electronic devices over the Web (Internet).

Web skilled people A person who has learned to use the Web's capacities (browsing, searching, use of functionality commonly available in web applications) and can apply them often with a minimum use of time and energy, and can learn new Web capabilities with a minimal effort.

W3C World Wide Web Consortium

WIRE Mashup approach Mashup approach in which the user needs to wire components on the design area in order to create a Mashup.

WWW World Wide Web

XML eXtensible Markup Language

CHAPTER 1: INTRODUCTION

1.1 Preface

The evolution of the web over the past few years has fostered the growth of some new technologies, e.g., Blogs, Wiki's, Web Services, and Mashups. Web Mashups gained lots of momentum and attention from both academic and industry communities (Beemer and Gregg, 2009). A Web Mashup is a web application that integrates data from more than one source. A well-known example is the use of cartographic data from Google Maps to add location information to some customer's data, thereby creating a new service that was not originally provided by either source. According to Kulathuramaiyer (2007), a Mashup comprises an application that "combines multiple sets of data streams into a unified user experience".

Currently, more than two billion people access the web for various purposes (Internet world stats, [n.d.]). The majority are people without programming or modelling backgrounds (called end-users). Part of these people also likes to create their own web applications to meet their daily needs. Mashup Makers are tools to create such end-user's web applications. As such, Mashup Makers could become the dominant environment for end-user development of web applications (Yue, 2010). However, to achieve this, the usability of these Mashup Makers is essential. Usability is an essential factor affecting the quality of web applications development environments (Ham et al, 2007). There are many recent studies focusing on software usability impact factors and usability evaluation of software artefacts from various viewpoints (Ham et al, 2007; Seffah et al, 2006). However, little research is dedicated to the usability of Web Mashup Makers. Therefore, this dissertation is concerned with the usability evaluation of Web Mashup tools for end-users.

The aim of the thesis is to make a contribution to the investigation of usability evaluation of software artefacts by proposing and developing a usability evaluation framework for Mashup Makers for end-users.

This chapter introduces the research context and research problem. From this, the research objectives are formulated. An overview of the structure of the thesis is provided in the thesis outline.

1.2 Research Context

In the past five years, the web has experienced a surge in growth; a phenomenon described by O'Reilly (2009), as the emergence of Web 2.0, a new trend for web applications including Mashups that emphasizes services, participation, scalability, remixability, and collective intelligence. In general, the term Web 2.0 is commonly used to refer to the current generation of social web applications being developed today (Beemer and Gregg, 2009). However, in (Cappiello et al, 2011), it is stated that the development of modern Web 2.0 applications is increasingly characterized by the involvement of end-users with typically limited programming skills. According to these authors, an emerging practice is the development of Web Mashups.

The concept of Mashup is commonly known as follows: Mashups are Web 2.0 applications and services that allow the non-programmer web-user to mix applications from different sites that can be pulled together in order to experience the data in a novel and enhanced way (Ankolekar et al, 2007). Let's illustrate this with an example.

Suppose somebody wants to schedule a trip to Paris for a week. If the person wants to do this using the web, he needs to visit many websites to book his/her train ticket or/and air flights, hotel rooms, restaurants, schedule visits to museums and tourist places, look for local transportations, and find interesting shopping opportunities in Paris. In general, the person likes to compare different offers and prices.

For such a trip schedule, we investigated the number of websites this person should visit and time he would need to spend. We found that the minimum number of websites is around 47 websites (including 12 sites to check air flights, 2 sites for train, 15 sites for hotels, 8 sites for museums and 7 sites for shopping and 3 sites for local transportation) and the time needed would not be less than 7 hours. It is also worth mentioning that the user has to use many other resources such as pens, calculator, calendars, and papers or notes sheet to leave comments and to compare finding at different times.

In contrast to this situation, we found that one may need about only fifteen minutes to schedule this trip using a Mashup tool and he/she will not need to visit more than two

sites (being the Mashup Maker tool site to create the Mashup and the resulting site showing the findings (schedule and offers)). Such a mashup would also eliminate the need for other materials such as pens, calculators, calendars, and note sheets.

Yue (2010), states that the potential of Mashups Makers as end-user development tools for Web 2.0 applications is not only in its ubiquity; it is also a focal point of three interlinked major trends in information systems: Web 2.0, situational software applications, and end-user programming. Situational software application are software applications that can change how users access, perceive, and consume information for a specific purpose, letting them focus on what to do with information rather than where and how to acquire it (Balasubramaniam et al, 2008). Brancheau and Brown (1993) describe end-user development as "… the adoption and use of information technology by people outside the information system department, to develop software applications in support of organizational tasks". However, there is a great request to provide end-users with powerful and flexible environments, tailorable to the culture, skills and needs of a very diverse end-user population (Costabile et al, 2006).

If Web Mashup Makers are intended to become the end-user development tools for Web 2.0 applications, usability of these Mashup Makers is an essential factor affecting their quality (Ham et al, 2006) and acceptation. However, while usability cannot be accurately and fully evaluated in any way, it can be estimated or evaluated by some usability impact factors which provide a basis for decision making (Heo et al, 2009). A usability evaluation factor could be described as: “an entity, resource or a unit of information which refer to or provide meaning of an evaluation of the usability of certain object (Karwoweski et al, 2011). HCI (Human Computer Interaction) research, in particular research on development of usability evaluation frameworks can contribute to the improvement of systems used (Boott et al, 2001; Haklay and Harrison, 2002). This is due to at least two reasons. On the one hand, HCI techniques, including usability evaluation frameworks, are geared towards understanding how people interact with computer applications within an environment. On the other hand, they are built upon methods researched and validated in a number of scientific fields (Thomas and Macredie, 2002). From the definition of software usability framework in (Riehle, 2000), we can define a

usability evaluation framework as a framework that could provide structured approaches, models, guidelines and criteria's that help in evaluating the usability of a software artefact.

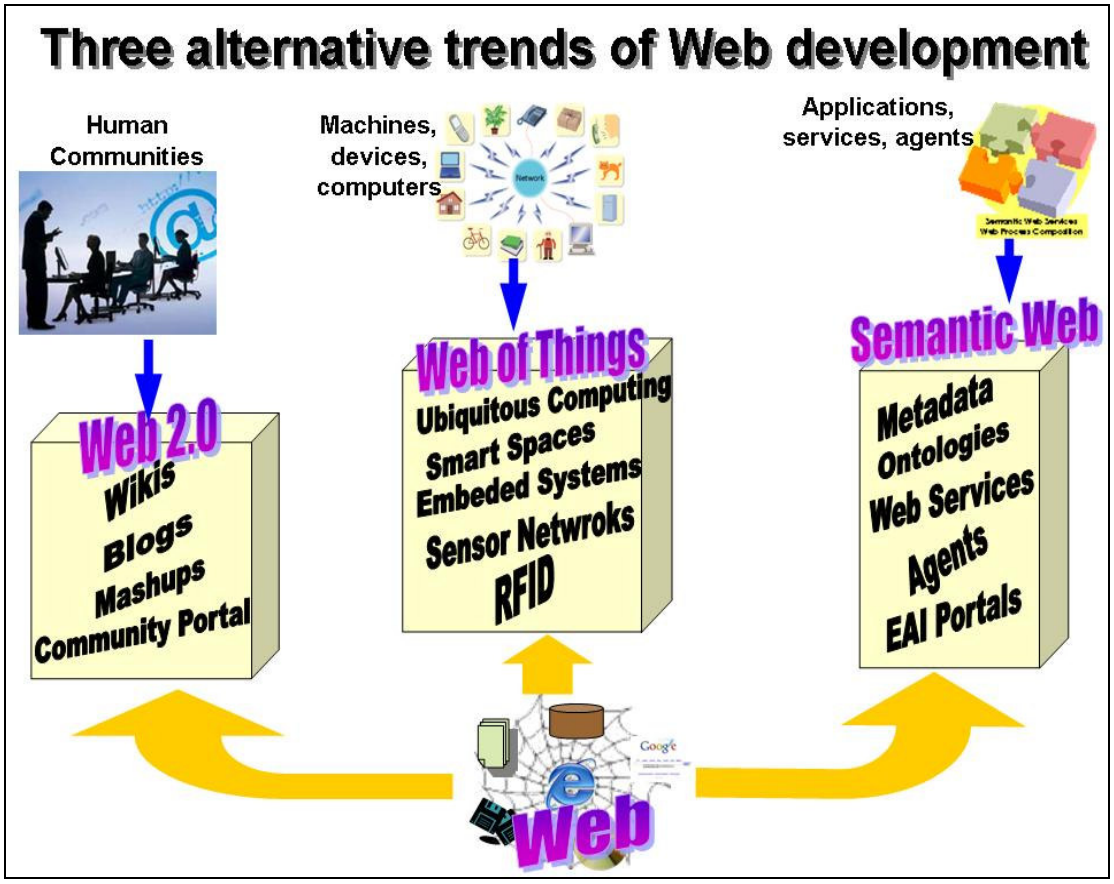

Figure 1.1: Web Development Trends (Terziyan, 2007)

Furthermore, Terziyan (2007) highlighted three main alternative trends of future Web development (see figure 1.1 taken from (Terziyan, 2007)). The first trend is the one related to new technologies dedicated to end-users and human communities; this trend includes technologies like Web 2.0, Wikis, Mashups, Social networks and community portals. The second trend is the one related to the concept of having an integrated relation between the triple representing the computing environments (machine, device and computer); this includes concepts as the Web of Things, ubiquitous computing, smart spaces, embedded systems, and sensor networks. The third trend is that related to innovations and improvements of web application, services and agents; this trend

includes concepts and aspects as Semantic Web Ontologies, Web services, Agents and EAI portals. In our opinion, the rational trend to web development could be the improvement of existing web technologies to satisfy end-users and to emphasis web technologies in every aspect of people life, not only to facilitate their life but also to help them integrate in the information era.

A lot of research and studies on web technologies and related concepts and aspects have been done the last two decades. Some research tracks deal with the technical aspects of web developments. Others handle the administrational aspects and concepts including research topics like management information systems and business intelligent systems. Some others tackle the human perspective and how one could achieve the best web environment for people, i.e. web usability. In general, websites lacking a systematic underlying design can suffer from enormous usability problems (Nielsen, 1992). One particular aspect of web usability, and which is the focus of our PhD work, is the usability evaluation of web applications development environments for non-technical users.

To accommodate the non-technical user in having better web artefacts that satisfy his/her needs, usability evaluation researches and studies have been introduced and done (Nielsen, 1993; 1999; 2003); (Heo et al, 2009); (Ham et al, 2006); (Donayee et al, 2006); (Seffah et al, 2002); (Hasan, 2009); (Blackwell et al, 1999); (Green et al, 2001). A lot of research has been performed to provide usability benchmarks and guidelines for such software systems and artefacts, e.g. (Donayee et al, 2006); (Seffah et al, 2002); (Hasan, 2009); (Blackwell et al, 1999); (Green et al, 2001). Those benchmarks and guidelines could help both designers and usability practitioners in providing better systems for end-users and providing complete and well defined frameworks of usability evaluation for web application development environment for end-users. To evaluate usability in a more systematic way, many studies examined factors or dimensions constituting usability (Bevan, 1999). For example, ISO/IEC 9241 (1998) defines three dimensions: effectiveness, efficiency, and satisfaction. Another example is the one described in (Nielsen, 1993): learnability, efficiency of use, memorability, errors, and satisfaction (Ham et al, 2006).

1.3 Research Relevance and Problem

Most of the research on web application development for end-users (Mashups in particular) is concerned with software engineering aspects, and not with the end-user perspective that is concerned with usability aspects. Studies and research on usability impact factors in this context barely exist or are not mature enough to identify and organize usability impact factors of Mashup Makers in a systematic way (Frøkjaeer et al, 2000).

Our research is based on the observation that Web Mashup Makers are often not easy to be used (correctly) by non-technical users, especially not the first time. This may result in high frustration and especially non-technical people may give up and not use the tools anymore. These are missed opportunities for the environments providers and developers and may be the cause for the environment (tool) to disappear (as we have seen a lot in the past few years). Therefore, the research is centred on the investigation of the usability evaluation of Mashup Makers for end-users. The aim is to provide a usability evaluation framework for Mashup Makers that can be used by usability practitioners and software environments developers.

Anticipation of user feedback and usability evaluation guidelines could be beneficial for Web Mashup tools developers for different reasons: to anticipate on usability problems during development process of Mashup Makers, to detect and correct development flaws, to select between development alternatives, and to realize both the functional and the business goals of Mashup Makers.

Usability guidelines, techniques and metrics have proven very helpful in evaluating traditional desktop computing applications, but they are not sufficient for ubiquitous applications that place more emphasis on intuitiveness, end-user daily needs, privacy, trust and other social aspects of computing (Theofanos and Scholtz, 2005). Furthermore, those usability guidelines can be provided in a more structured way as a usability evaluation framework that keeps development and implementation of web application

environments consistent. However, such a usability evaluation framework is currently not available.

1.4 Research Objectives

Our research objective is to investigate how one can **measure and improve** the usability of Mashup Makers for end-users. By end-users we mean casual web users, usually without programming background, who want to create their own (small) web applications. For this reason, we focus on general-purpose Mashups Makers (as opposed to specific-purpose Mashups Makers). In general, an end-user is seeking for a general-purpose Mashup maker for quickly creating small web applications for multiple purposes and with ease of use.

Our research is important for three reasons. First, it is important to check (or be able to check) if Mashup Makers indeed fulfil their promises and meet the needs of end-users. If they do not fulfil their promise, then it would be useful to give guidelines on how they can be improved. This brings us to the second reason, which concerns investigating the usability necessities for Mashup Makers for casual users in general. The third reason concerns the potential of Mashups Makers as end-user development tools for Web 2.0 applications. As already explained, Mashup Makers are considered as the dominant environments of end-user web applications development (Yue, 2010). It is also worthwhile to mention that usability of Web 2.0 applications composition for end-users is an emerging research field track of End User Development (EUD) (Lieberman et al, 2006).

In this PhD, we are concerned with a method pertained to usability inspection. We aim to develop a framework for supporting usability experts and Mashup Maker developers to evaluate the usability of Mashup Makers and predict likely usability problems in an analytical manner.

To achieve this goal, we have formulated the following research objectives:

(1) To discover the main issues related to Web Mashup Makers, Web Mashup usability evaluation approaches, and to have a concrete understanding of the usability of Mashup Makers for end-users.

(2) To deeply investigate usability issues of Mashup Makers for end-users by performing empirical studies (pilot studies and user experiments), and to draw on the findings of the empirical studies in establishing a consolidated usability evaluation model for Mashup Makers for end-users.

(3) To develop a usability evaluation framework for Mashup Makers for end-users which will support usability experts and Mashups Maker's designers evaluating the usability of their Mashup Makers and to validate the framework developed.

1.5 Thesis outline

This introductory chapter includes a preface of the research topic, research context, and the research objectives. The rest of the dissertation consists of three parts.

Part one: **The research method, background, and related work.**

This part consists of two chapters (chapter 2 and chapter 3). Chapter 2 presents the research method used in this thesis. This chapter presents an overview of the research philosophy, together with the research design and the methods employed. Justifications for selecting these methods are also given in this chapter. The second chapter also briefly previews the reliability and validity of the research method. The third chapter reviews the background for this thesis. This chapter reviews: Mashups and Mashup Makers, their types, their functionality and their composition approaches, as well as usability, usability evaluation methods, the usability of Mashup Makers, and the effectiveness of usability evaluation methods in identifying usability problems. This chapter also includes the related work.

Part two: **The empirical study and conceptual modelling of findings.**

This part consists of two chapters (chapter 4 and chapter 5). Chapter 4 presents the empirical studies performed in the research, both the qualitative and quantitative findings obtained from the pilot study and the user experiment/study are presented. The chapter also describes the user experiment's approach, goal, methodology, design, and results. It presents the results as a set of lists of common usability problems identified. The chapter also summarises the overall usability problems of the Mashup Makers. Chapter 5 defines the usability impact factors identified. The usability impact factors for Mashup Makers are presented as a Conceptual Evaluation Model. The role of this conceptual model is to structure the main usability indicators of Mashup Makers for end-users and to prepare for the next step of establishing the Usability Evaluation Framework of Mashup Makers for end-users. We identified three main aspects for usability impact factors and used these as the basis for the Conceptual Evaluation Model. The effectiveness of each aspect in identifying specific usability factors of Mashup Makers is explained.

The empirical studies were presented and published in the proceedings of two conferences. The pilot study is presented (and published in the proceedings) at the 9th International conference of Web engineering (ICWE2009) in June 2009 in Spain. The user experiment is presented (and published in the proceedings) at the 12th International Conference on Information Integration and Web-based Applications & Services (iiWAS2010) in November 2010 in France.

Part three: **The usability framework and its validation**.

This part consists of the last two chapters of the dissertation (chapter 6 and chapter 7). Chapter 6 presents our usability evaluation framework for Mashup Makers for end-users (called MUEF). The framework has a hierarchal multi-layered architecture. The chapter describes the framework, its components, as well as how to employ the framework. It also explains its usefulness. Chapter 7 presents the evaluation and validation process of the MUEF framework. This has been done using an empirical study with a number of experts in the domain. The chapter presents the experimental study, it approach,

objectives, design, performance and results. We also discuss the findings and its impact on future work.

Finally, chapter 8 presents the conclusions of this research. The chapter explains how the aims and objectives of this research have been accomplished. Then the chapter summarises the limitations of the research and gives recommendations for future work.

Part I: The research method and background

CHAPTER 2: RESEARCH METHOD

This chapter presents an overview of the research philosophy used, the objectives, the design of the research and the methods employed to achieve the aims and objectives of this research. This is followed by a discussion on the reliability of the research method for achieving the research objectives.

2.1 Research Philosophy

The aim of this section is to highlight the research philosophy related to this research work and to clarify our choices and the research philosophy adopted.

The design of any research starts with the selection of a topic and a paradigm or philosophy (Creswell, 1994). The research paradigm/philosophy offers a framework, consisting of theories, methods and ways of defining data, which explains the relationship between data and theory (Collis and Hussey, 2003), (Easterby-Smith et al, 1991). In (Easterby-Smith et al, 1991), it is stated that understanding the philosophical issues in a research study is very useful. Firstly, it can help to define the research design in terms of considering what type of evidence is required, how it will be gathered and interpreted, and how this will provide answers to the research questions. Secondly, it can help the researcher to identify which research design will work and which will not. Furthermore, it helps him/her to reveal the limitations of particular approaches. Also, it can help the researcher to determine, and even to develop, designs that may be not related to his/her experience; it may also suggest how to adjust research designs with regard to the limitations of different knowledge structures (Hasan, 2009).

There are two main research philosophies or paradigms that guide the design and methods of research. These are positivism and interpretivism. (Saunders et al, 2007a). These approaches have different propositions regarding common assumptions concerning obtaining knowledge and the process of research (Hasan, 2009). The most common assumptions are termed epistemology, ontology and the logic of the research. Epistemology concerns how a researcher will obtain knowledge during his/her inquiry/research; ontology concerns how each paradigm views reality (knowledge), or what is considered reality from the viewpoint of the researcher; and the logic of a

research describes the nature of the relationship between research and theory, which could be, according to Bryman (2008), either deductive or inductive.

We briefly outline the two approaches in terms of their assumptions. The positivism approach believes that: “the study of human behaviour should be conducted in the same way as studies conducted in natural sciences” (Collis and Hussey 2003). This implies using the scientific method approach of research, or the same methods, principles, procedures and ethos as the natural sciences (Creswell, 2003; Bryman, 2008).

On the other side, interpretivists believe that: “what is researched can’t be unaffected by the process of research” (Collis and Hussey, 2003). The researcher is not observing phenomena from outside the system, like the natural sciences, but he/she is involved with what is being researched (Nicholas, 2006; Collis and Hussey, 2003). Reality is subjective and socially constructed and can be understood by examining and investigating participants in the study (Collis and Hussey 2003).

¨In recent years, several academic institutions have attempted to integrate design, with technology and behavioural science in support of HCI (Human-Computer Interaction) education and research (Zimmerman et al, 2007) ¨. Zimmerman et al. (2007) also report that no agreed upon research model existed for interaction designers to make research contributions other than the development and evaluation of new design methods or mixed ones.

While our research topic is situated in the human-computer interaction science, we found it more practical to follow a mixed research philosophy/method somewhere on the border of the intersection between social science, business, computer science and engineering.

In (Saunders et al, 2007a), Saunders et al. describe scientific research as an onion with multi-layers as shown in figure 2.1. We found this figure and schema realistic and practical to highlight our adopted research philosophy.

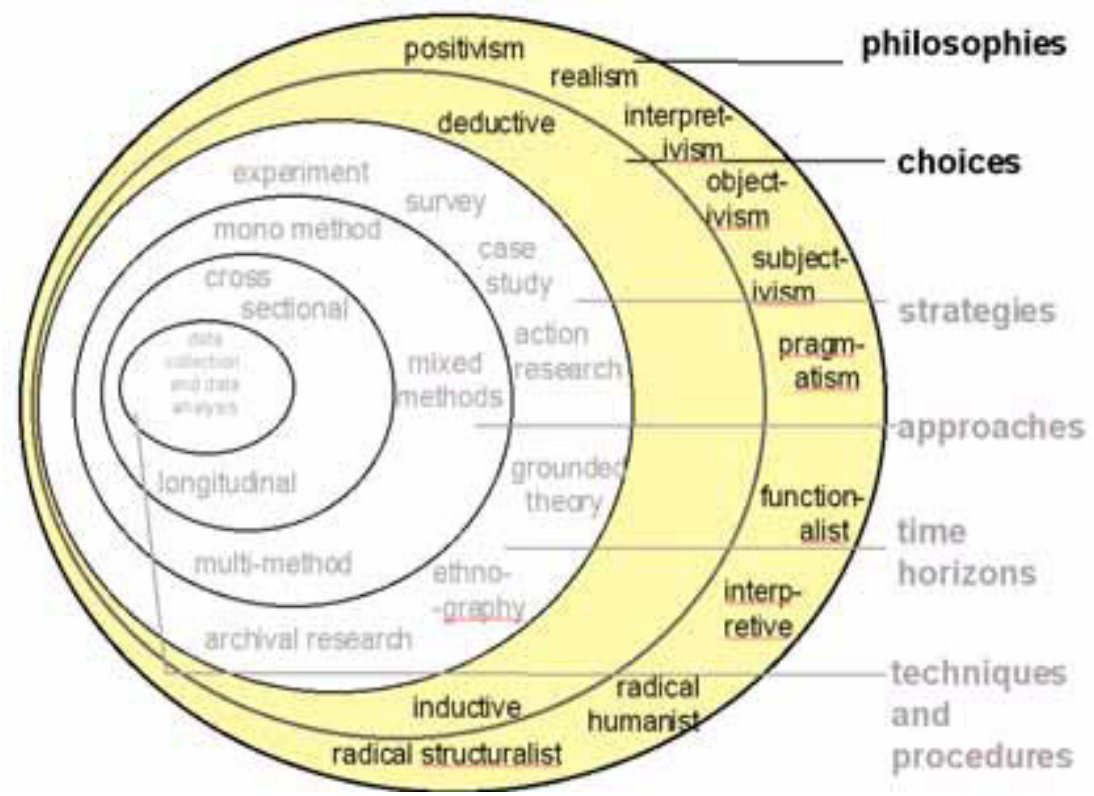

Figure 2.1: Research Onion (Saunders et al, 2007a)

In figure 2.1, the outer layer is the research philosophies layer. There, many concepts are mentioned such as positivism, interpretivitism, subjectivism, functionalism …etc. In our opinion, there could be an intersection or overlapping between those concepts and philosophies. The second layer of the research onion figure contains the research choices (quantitative and qualitative) quantitative research is confirmatory and deductive in nature and qualitative research is exploratory and inductive in nature (Saunders et al, 2007a). Further details about research choices are shown in figure 2.2 and explained in the text follows the figure. In figure 2.1, they are deductive (quantitative) or inductive (qualitative). Also here, we found our self following both research choice options and having in many cases a mixed research choice. The other layers of the research onion of figure 2.1 deal with the more detailed tasks undertaken during the research work, as well as with the detailed techniques (survey, grounded research, case study … etc.).

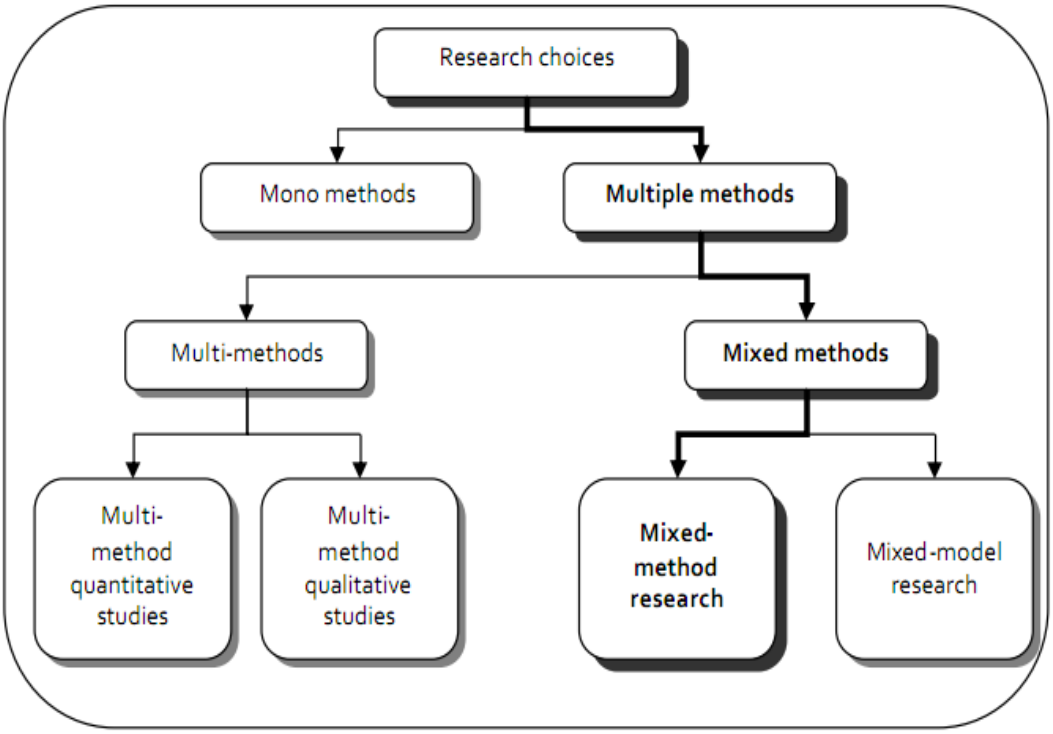

Figure 2.2: Choices of the research methods (Saunders et al, 2007b)

There are several choices for the research method (Saunders et al, 2007b). Researcher may employ only one single type of method or combine alternative methods. In order to apply multiple methods to study the same phenomenon, research may consider any of the two choices in the branch Multiple Methods (Figure 2.2). In our research, several research methods are combined for the purpose of facilitation and triangulation (Hammersley, 1996). Triangulation in the context of this research means the mixing of data or methods used so that diverse viewpoints or standpoints cast light up on a topic (Olsen, 2004).

At the beginning, our research was designed based on the positivist's paradigm. Positivists have ontology, ¨which is defined as the opinion of what is the truth, in which the reality is observable and the objective world exists¨ (Näslund, 2002). Moreover, epistemology is described as the interrelationship between researchers and what to be researched. In positivism, the researchers and what to be researched should be separated (Hussey and Hussey, 1997; Gummesson, 2000).

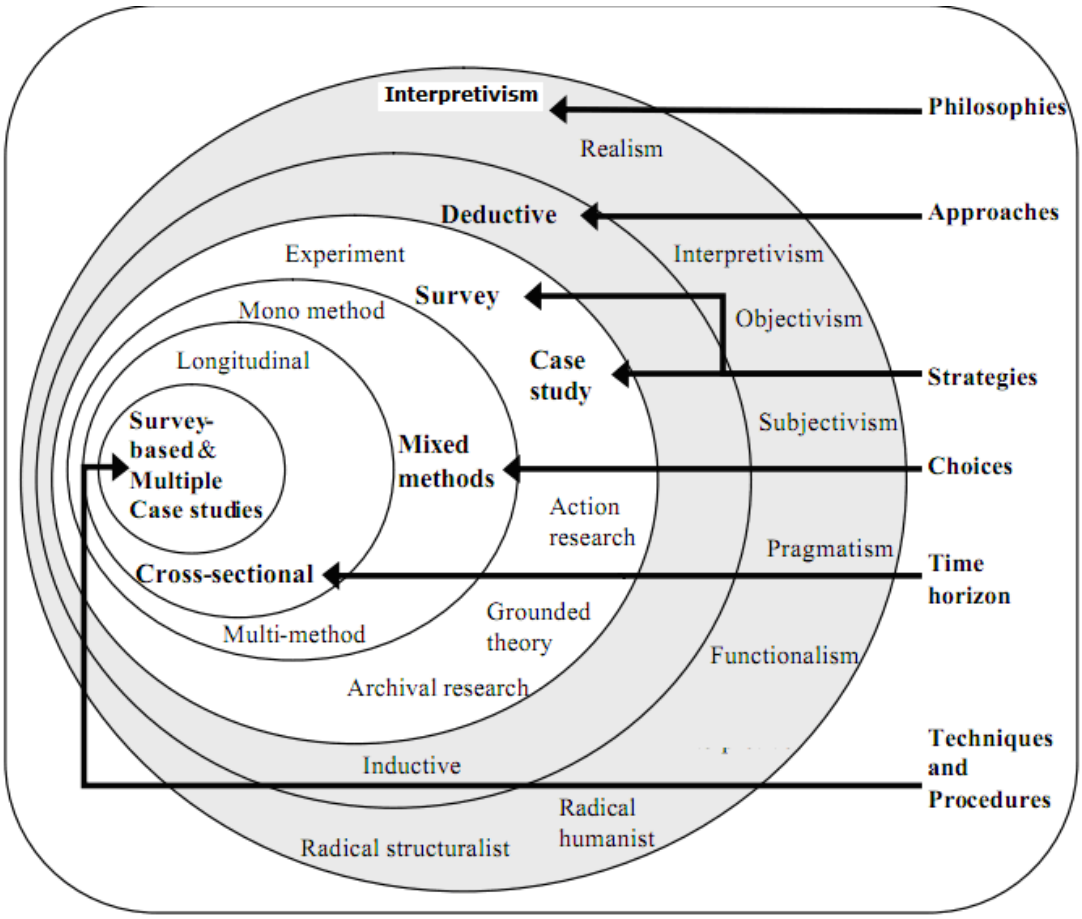

Figure 2.3: Adapted research onion (Saunders et al, 2007b)

Following the philosophical observations made above, and referring to the aims and objectives of this research (as mentioned in Chapter 1), this research has adopted an interpretivist approach. This selection can be justified as follows. Interpretivism is an appropriate approach with regard to our type of research problem. For example, Creswell (1994) showed that a research problem is related to a positivist approach if it evolves from the literature where variables and theories may exist that need to be tested and verified, while a research problem is related to an interpretivist approach when little information exists on the topic and more exploration is needed since the variables are largely unknown. Therefore, as we are in the second situation, it is clear that the interpretivism approach is an appropriate one to be adopted in this research, as it is not guided by theory that must be tested objectively. Instead, it is aimed at finding an

understanding regarding which usability methods are the best in evaluating usability issues for Mashup Makers.

Figure 2.3 represents our research approach in terms of the research onion of Saunder. The bold texts are those of the selected choices in the research. Accordingly, under interpretivism, the study employs mixed-methods analysis using cross sectional data.

A mixed procedure of qualitative and quantitative research methods is used. In figure 2.3 this procedure is represented in the layer of mixed-methods where the bold arrow Choices resides. The mixed procedure is the predominant method in our research. This is performed by the semi-structured surveys performed in the literature study and the empirical study/investigation of the usability of Mashup Makers. The pilot study presented in chapter 4, and done in the early stage of the research, is designed to facilitate the hypothesis validation and also to aid the measurement of the model (the research approach). Such a mixed procedure is the integration of difference research methods to study a single phenomenon in order to avoid sharing the same weakness (Voss et al, 2002). More details about the research design and method are provided in next sections 2.2 and 2.3.

2.2 Research Design

Herewith, we describe the milestones and major phases of our research work toward the achievement of the proposed usability evaluation framework.

There were three main research phases represented by the three parts mentioned in the thesis's outline in section 1.5.

2.2.1 Phase 1: Literature Study

In the first phase, a literature study on Mashup Makers, on usability issues related to, and on the criteria by which it is possible to evaluate the usability of Mashup Makers for end-user has been performed. In (Al Sarraj and De Troyer, 2009), we highlighted the main usability issues we have investigated during this phase. However, note that a literature

investigation is carried out during the whole research period in parallel with the other phases.

2.2.2 Phase 2: Empirical study and Conceptual Modelling of the Findings

In the second phase, we carried out an initial usability evaluation of 6 Mashup Makers, called the pilot study (Al Sarraj and De Troyer, 2009). From this pilot study, we obtained different points of refinement for the usability evaluation methodology used, as well as input for the preparation of the next step, the user experiment.

Next, we performed a user experiment/study (Al Sarraj and De Troyer, 2010). In this experiment, end-users were asked to evaluate Mashup Makers by performing some tasks. Our approach in evaluating the usability of Mashup Makers was based on the adoption of the four main dimensions of usability factors from the ISO9241 definition and ten usability evaluation criteria adopted from the Cognitive Dimensions Framework (Blackwell and Green, 2000).

Based on the results obtained in the empirical study, we identifying the main usability factors for Mashup Makers and developed a Conceptual Evaluation Model using these usability factors.

2.2.3 Phase 3: Usability Evaluation Framework and its Validation

In the final and accumulated phase and based on the Conceptual Evaluation Model developed in phase 2, we have defined an analytical multi-layered Usability Evaluation Framework for Mashup Makers for end-users. In this phase, we also performed an evaluation and validation process of the developed framework using an experimental study.

As explained above and in connection with the research philosophy we adopted and explained in section 2.1, the framework of usability evaluation of Mashup Makers for end-users was constructed by way of an interpretation of the different methods involved in this research. Specifically, in order to develop the framework and to identify usability

issues, we have interpreted users' actions while interacting with the Mashup Makers (tools), evaluator's comments, and the statistics obtained from both the pilot study and the user experiment, as well the feedbacks of the experts during and after the evaluation experiment performed in the process of the evaluation of the proposed framework (chapter 7).

2.3 Research Methods

In this section, we elaborate the three phases of the research design mentioned in the previous section. For each phase, the different steps followed, as well as their objectives and the research methods used, are given.

2.3.1 Phase 1: Literature Study

Step 1. Objective: To obtain an overview of the existing Web Mashup Makers, in order to discover the main issues related to Web Mashup technology and to have a concrete understanding of the possibilities and limitations of Web Mashup technology.

> **Method:** A literature study on Web Mashup technologies, and a study of tutorials of Web Mashup Makers.

Step 2. Objective: To obtain an overview of Mashup usability in order (1) to discover the main issues related to Mashup usability and to obtain a good understanding of Web Mashup usability; (2) to check related work in the context of measuring the usability of Web Mashup technology.

> **Method:** Literature study on usability, web usability and Web Mashups usability.

2.3.2 Phase 2: Empirical Study and Conceptual Modelling of the Findings

Step 3. Objective: To obtain a deeper understanding of the usability issues related to Mashup Makers for casual end-users.

Method: Pilot Study - Part A: Selection of some Web Mashup Markers for casual users and performing experiments with them in order to get practical knowledge and experiences on how these tools work. The experiments are performed by the author.

Step 4. Objective: To define a set of Mashup usability criteria, i.e. usability measurement factors that can be used to evaluate the usability of Web Mashup Makers for the target audience (casual end-users).

Method: Pilot study - Part B: Critical analysis of the results of Step 2 and Step 3, and the identification of missing and/or irrelevant usability issues. Further investigation of the relationship between the usability criteria identified and the target audience.

Step 5. Objective: Development of an experimental environment.

Method: Selection of a set of representative Web Mashup Makers to be used in the experiment; selection of a representative set of target users; preparation of the experiment that will be performed.

Step 6. Objective: To reach the second objective of the research: empirically investigating the usability of existing Mashup Makers.

Method: Performing the usability experiment prepared in Step 5; analysing the results by means of statistical methods, and summarizing the results.

Step 7. Objective: To draw on the findings of the empirical studies performed (pilot study and user experiment) and to develop a conceptual evaluation model of usability for Mashup makers for end-users.

Method: Constructing a well-defined conceptual evaluation model for the usability of Mashup Makers for end-users by identifying the components of such model and clarifying the main aspects related to the developed conceptual model.

2.3.3 Phase 3: Usability Evaluation Framework and its Validation

Step 8. Objective: To reach the third objective of the research: development of a usability evaluation framework for Mashup Makers for end-users.

> **Method:** Reinvestigation of recent research to keep track of new developments; evaluation of the approach used in Step 6; collecting and resuming guidelines, criteria's and benchmarks for Mashup usability into a coherent usability framework.

Step 9. Objective: To evaluate and validate the effectiveness and usefulness of the developed framework.

> **Method:** Performing an evaluation of the approach developed in Step 8; designing and performing a validation and evaluation process, discuss results, formulate conclusions and recommendations.

In this section, we have illustrated a step-by-step approach for conducting mixed-methods research in usability investigation of Mashup makers for end-users.

2.4 Conclusion

This chapter presented the research philosophy used and its justification, as well as the design of the research and methods used to achieve the aims and objectives of this research. Also, the chapter discussed the phases employed to achieve the objectives of the research: literature study, empirical study and modeling its findings, and the development of usability evaluation framework.

CHAPTER 3: BACKGROUND AND RELATED WORK

The evolution of the web over the past few years has fostered the growth of some new technologies, e.g., Blogs, Wiki's, Web Services, and Mashups. At a certain moment, Web Mashups gained lots of momentum and attention from both academic and industry communities (Beemer and Gregg, 2009).

Currently, more than two billions people access the web for various purposes (Internet world stats, [n.d.]) (see figure 3.1). Figure 3.1 shows the distribution of world's Internet usage and population statistics over the world regions. The majority of Internet users, the so-called end-users, are people without programming or modelling backgrounds. Part of these end-users also likes to create their own web applications to meet their daily needs. Mashup Makers are tools to create such end-user's web applications. As such, Mashup Makers could become the dominant environment for end-user development of web applications (Yue, 2010). However, to achieve this, the usability of these Mashup Makers is essential. Usability is an essential factor affecting the quality of any interactive application and in particular web application development environments (Ham et al, 2007). There are many recent studies focusing on software usability impact factors and usability evaluation of software artefacts from various viewpoints (Ham et al, 2007; Seffah et al, 2006). However, little research is dedicated to the usability of Web Mashup Makers. Therefore, this dissertation is concerned with the usability evaluation of Web Mashup tools for end-user.

WORLD INTERNET USAGE AND POPULATION STATISTICS
March 31, 2011

World Regions	Population (2011 Est.)	Internet Users Dec. 31, 2000	Internet Users Latest Data	Penetration (% Population)	Growth 2000-2011	Users % of Table
Africa	1,037,524,058	4,514,400	118,609,620	11.4 %	2,527.4 %	5.7 %
Asia	3,879,740,877	114,304,000	922,329,554	23.8 %	706.9 %	44.0 %
Europe	816,426,346	105,096,093	476,213,935	58.3 %	353.1 %	22.7 %
Middle East	216,258,843	3,284,800	68,553,666	31.7 %	1,987.0 %	3.3 %
North America	347,394,870	108,096,800	272,066,000	78.3 %	151.7 %	13.0 %
Latin America / Carib.	597,283,165	18,068,919	215,939,400	36.2 %	1,037.4 %	10.3 %
Oceania / Australia	35,426,995	7,620,480	21,293,830	60.1 %	179.4 %	1.0 %
WORLD TOTAL	6,930,055,154	360,985,492	2,095,006,005	30.2 %	480.4 %	100.0 %

Figure 3.1: Internet Users in the World – 2011 (Internet world stats, [n.d.]).

In this chapter, we first present an overview of Mashups, as well as an investigation of Mashup Makers as web application development environments for end-users. Next, we present different types of Mashup Makers, the most famous ones and their composition approaches. We also discuss and emphasize the importance of Mashup Makers for research on end-user development of web applications.

Then, we present an overview of the state of the art about usability, together with an investigation of common usability evaluation methods that could be used to evaluate Web Mashup Makers for end-users; the effectiveness of the various usability evaluation methods is also discussed. Finally a review of related work is presented.

This chapter is organized as follow: section 3.1 presents Mashups, Mashup types, Mashup Makers and composition approaches of Mashup Makers. Section 3.2 presents the state of the art of usability and usability evaluation methods and effectiveness of usability evaluation methods for Mashup Makers. Section 3.3 presents related work. And finally section 3.4 concludes the chapter.

3.1 Mashups

Mashup originally referred to the practice in pop music (Wikipedia [n.d.]) (notably hip-hop) of producing a new song by mixing two or more existing pieces. In computer technology, a Mashup is a web application that integrates uses, and combines data, presentation or functionality from two or more sources to create new services. A well-known example is the use of cartographic data from Google Maps to add location information to some customer's data, thereby creating a new service that was not originally provided by either source. According to (Kulathuramaiyer, 2007), a Mashup comprises an application that "combines multiple sets of data streams into a unified user experience". It refers to an ad hoc composition technology of web applications that allows users to draw upon content retrieved from external data sources to create entirely new services (Liu et al, 2007). Mashup approaches can be observed in many different fields, e.g., for enterprise information systems (Jhingran, 2006) or digital libraries (Kulathuramaiyer, 2007). The most common way to develop a Mashup web application is

by accessing content for the Mashup via a public interface or API. This allows developers to directly feed data from databases and other sources and enable developers to build rich content applications that make information more useful to users. Mashup is a hallmark of Web 2.0 and attracts both industry and academia recently.

3.1.1 How Mashups work? "The common scenario"

As shown in Figure 3.2 (Zillner, 2007), the user requests to combine available data from two or more sources (from two or more API content providers). That data is made available by relevant web protocols such as REST, RSS and Web services (W3C, [n.d.]). The data is scraped from the output of these APIs, and then the scraped data is passed to the Mashup site where the logic resides, it could be server-side (dynamic content aggregation) and/or client-side scripting or both of them (Ort et al, 2007). The application then is rendered graphically and transferred to the client's web browser where user interaction takes place. The mashing is usually done by a client side web language, e.g., JavaScript, Ajax (Ort et al, 2007).

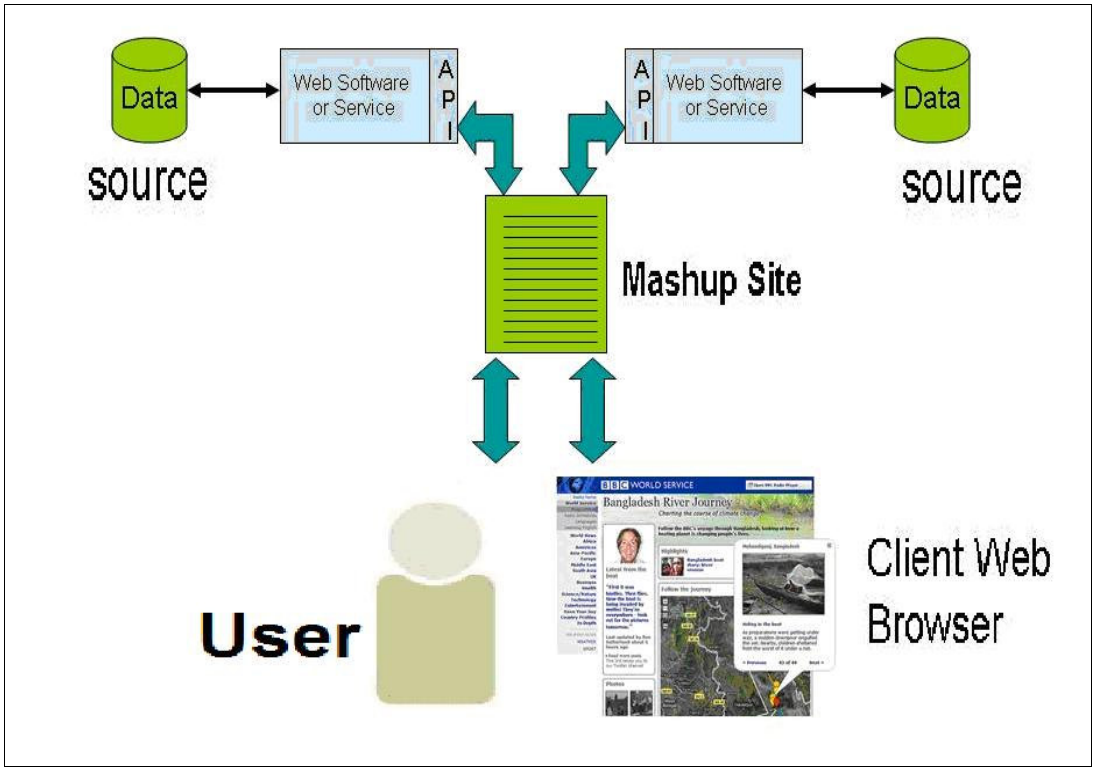

Figure 3.2: How Mashups work (Zillner, 2007)

3.1.2 Mashup Architecture

The actual Mashup is usually created in a Web browser, by "drag and drop" applications from different sources together. However, there must be some backend infrastructure to support the Mashup (Liu et al, 2007). In (Merrill, 2006), Merrill identifies Mashup as an application which architecturally is comprised of three different participants: API/content providers, the mashup hosting site, and the consumer's web browser, which is very similar to the popular three-tier architecture (Merrill, 2006). The architecture is shown in Figure 3.3.

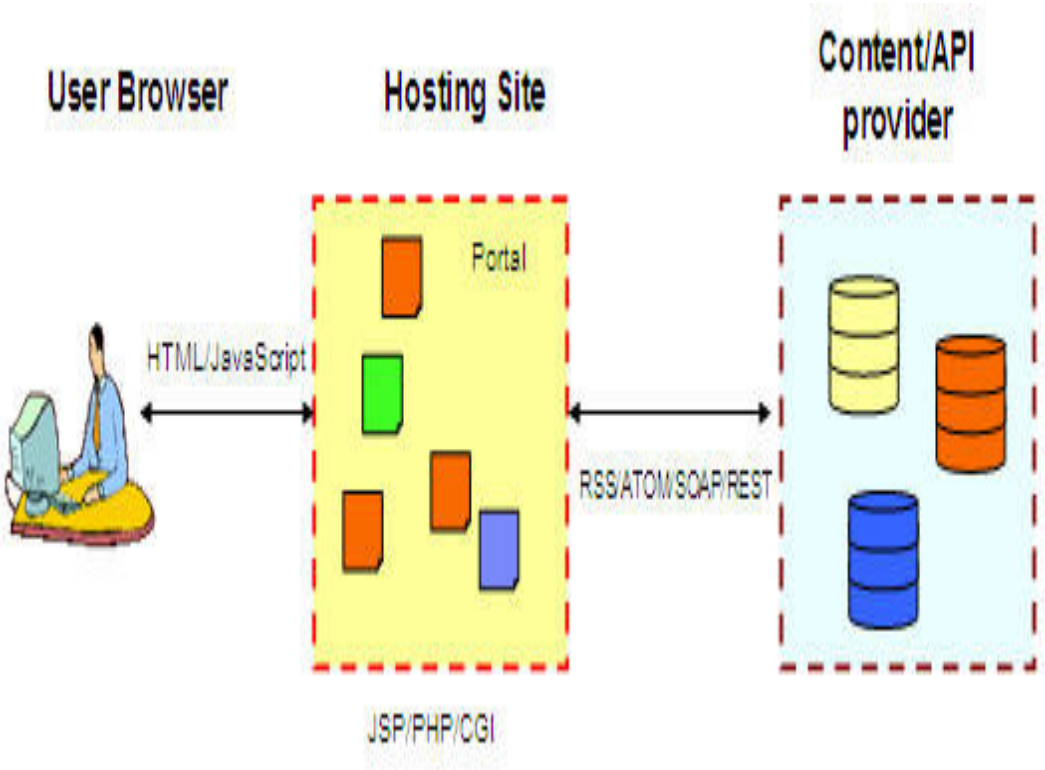

Figure 3.3: Mashup Architecture (Liu et al, 2007)

**** The API/content providers***

The API/content providers are the (sometimes unwitting) providers of the content being mashed-up. To facilitate data retrieval, providers often expose their content through web protocols such as REST, Web Services, and RSS/Atom. However, many interesting

potential data-sources do not (yet) conveniently expose APIs. Mashups that extract content from sites like Wikipedia, TV Guides, and virtually all government and public domain websites do so by a technique known as screen scraping (Merrill, 2006). In this context, screen scraping connotes the process by which a tool attempts to extract information from the content provider by attempting to parse the provider's web pages, which were originally intended for human consumption.

The Mashup hosting site.

The Mashup hosting site is where the Mashup is hosted. Just because this is where the mashup logic resides, it is not necessarily where it is executed. On one hand, Mashups can be implemented similarly to traditional web applications using server side dynamic content generation technologies like Java servlets, CGI, PHP or ASP. Alternatively, mashed content can be generated directly within the client's browser through client-side scripting (e.g., JavaScript) or applets. This client-side logic is often the combination of code directly embedded in the Mashup's web pages as well as scripting API libraries or applets (furnished by the content providers) referenced by these web pages. Mashups using this approach can be termed Rich Internet Applications (RIAs), meaning that they are very oriented towards an interactive user-experience. The benefits of client-side mashing up include fewer overheads on behalf of the Mashup server (data can be retrieved directly from the content provider) and a more seamless user-experience (pages can request updates for portions of their content without having to refresh the entire page). The Google Maps API is intended for access through browser-side JavaScript, and is an example of client-side technology. Often Mashups use a combination of both server and client-side logic to achieve their data aggregation.

**The consumer's Web browser*.

The consumer's web browser is where the application is rendered graphically and where user interaction takes place. As described above, Mashups often use client-side logic to assemble and compose the mashed content.

Mashups can be divided into many categories depending on their usage, such as: mapping, video, photo, search, shopping and news. According to programmableweb.com (Programmableweb, [n.d.]) which is the most well-known website dealing with APIs and Mashup tracking on the web, at the time of writing this thesis the top Mashup tags on the Internet are ‘mapping’, ‘video’ and ‘social’ (see figure 3.4). In other research work, Mashups are classified as patterns like those highlighted by (Wong and Hong, 2008). Those patterns are as follows: (1) Aggregation: A common function of Mashups is to aggregate multiple websites together or summarize sets of data. But this takes on multiple forms. (2) Alternate User Interface & In-situ Use. These Mashups don’t combine multiple websites at all but rather aim to support new methods of interacting with data from the website or support specific use cases. (3) Personalization. A number of Mashups personalize based on either personal information from the websites they are based on or new personal information from users. (4) Focused View of Data. This pattern is where a Mashup exists to index or categorize a subset of another website’s entire contents. (5) Real-time Monitoring. A number of Mashups support real-time monitoring (Wong and Hong, 2008).

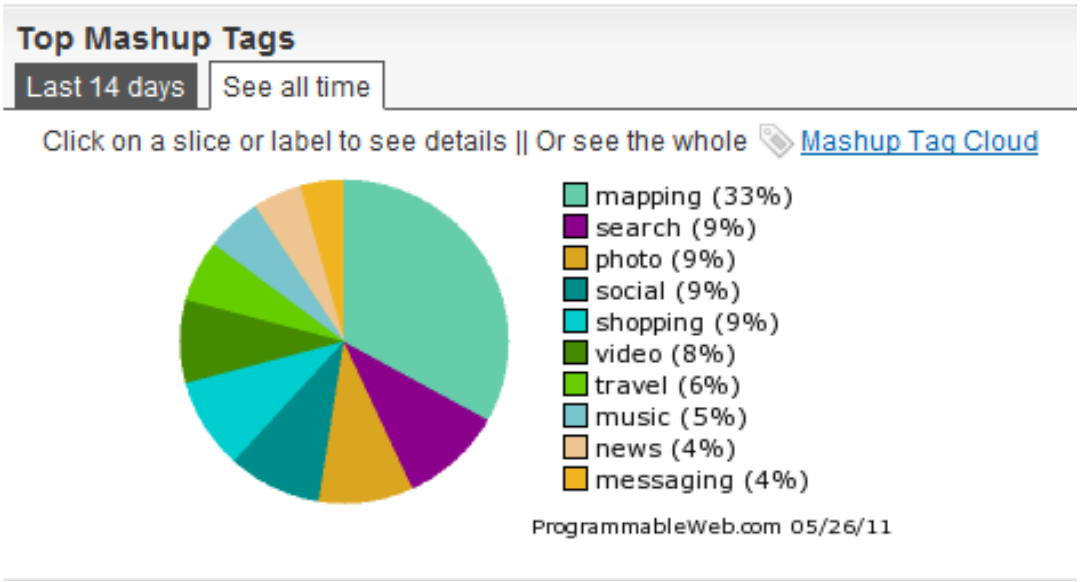

Figure 3.4: Top Mash-ups tags taken from programmableweb.com at May 26, 2011

3.1.3 Mashup Makers

There are several Mashup creation tools, we prefer to call them Mashup Makers, e.g., Yahoo Pipes (Yahoo! pipes [n.d.]), Microsoft Popfly (Microsoft Popfly [n.d.]), Intel MashMaker (Intel MashupMaker, [n.d.]), IBM Mashup Center (IBM [n.d.]), OpenKapow (OpenKapow [n.d.]), Open-Mashups Studio (OpenMashups studio, [n.d.]), Dapper (Dapper [n.d.]), Apatar (Apatar [n.d.]), Serena (Serena [n.d.]), and Jackbe (JackBe [n.d.]). From an end-user perspective, designing a Mashup by a Mashup Maker is a matter of either, using a cascading number of steps or by drop, define and link components and run applications (Al Sarraj and De Troyer, 2010). A Mashup Maker is a web tool with a user interface which provides the user the capability to combine web data, applications, and feeds to produce a Mashup that is useful for the user and does not exist in a single website.

In the following subsections we review some of the most well known Mashup Makers for end-users.

3.1.3.1 Yahoo! Pipes

Yahoo! Pipes (Yahoo! pipes [n.d.]) is a visual drag and drop Mashup creation tool for fetching and merging data from different sources. It does not require knowledge of programming languages, but still requires good understanding of data formats. The composition tool runs in a browser and is based on standard web technologies. The mash-up creation area is visually divided into 3 panes – on the left there is a library that lists all functional modules that can be pulled onto the canvas. In the bottom there is a debugger area that allows checking intermediate outputs. Modules are linked with connectors or "pipes" which define the data flow (see Figure 3.5). Many different things are possible with Yahoo! Pipes: one can combine many feeds into one, then sort, filter and translate it; geo code favourite feeds and browse items on an interactive map; and you can create power widgets/badges and place them on a personal website. Pipes support variety of output formats such as RSS, JSON, KML (W3C [n.d.]) as well as some others.

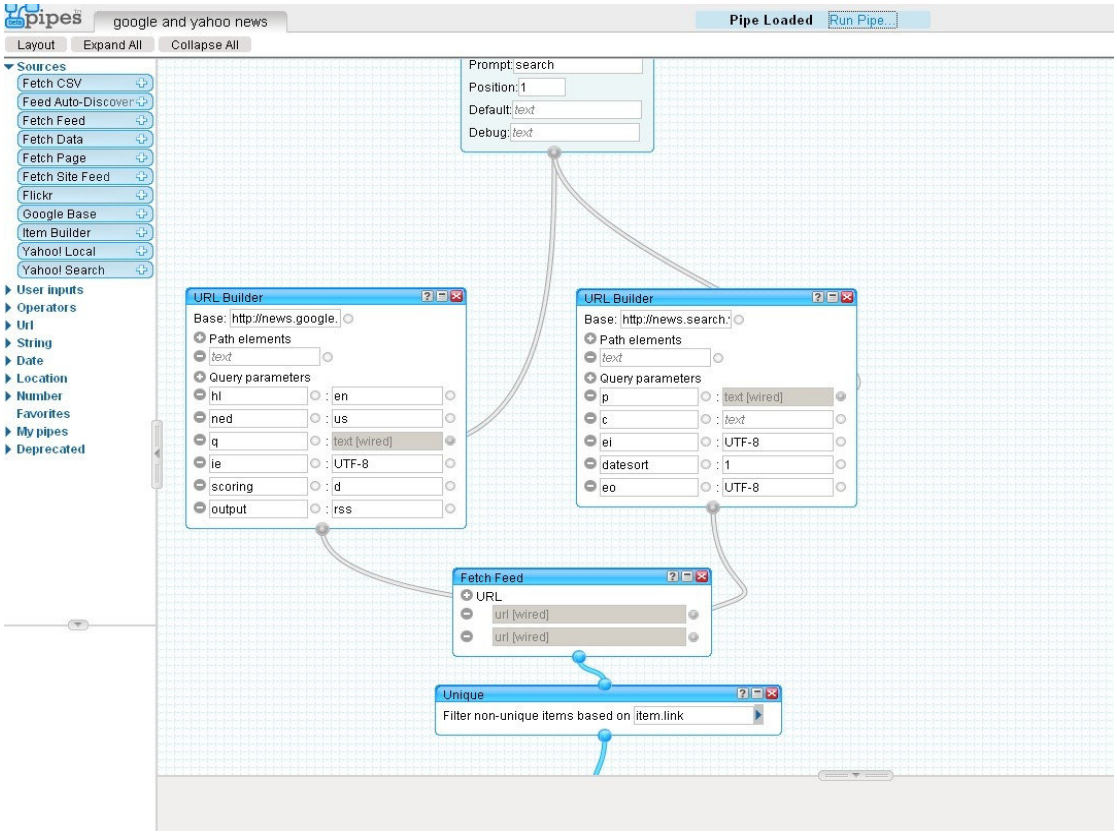

Figure 3.5: Yahoo! pipes data Mashup tool

3.1.3.2 Microsoft Popfly

Microsoft® Popfly™ (Microsoft Popfly [n.d.]) was a website and tool to help people create and share websites, Mashups, and other kinds of experiences. It had two parts: the social network, which is called "Popfly Space" and the online tool for creating different kinds of experiences, which is called "Popfly Creator." We are interested in the latter part of the service.

Similar to Yahoo! Pipes, Popfly had a pane with a functional block on the left and a canvas for assembling applications on the right. Modules are linked with connectors on the canvas. Popfly relied on the Silverlight technology from Microsoft to deliver visually appealing blocks and the composition environment. This was a drawback since it required an extra step – installing of Silverlight plug-in before user can use it. It looked very simple and it was appealing to use the service (see Figure 3.6).

Figure 3.6: Popfly in the mashup creation mode

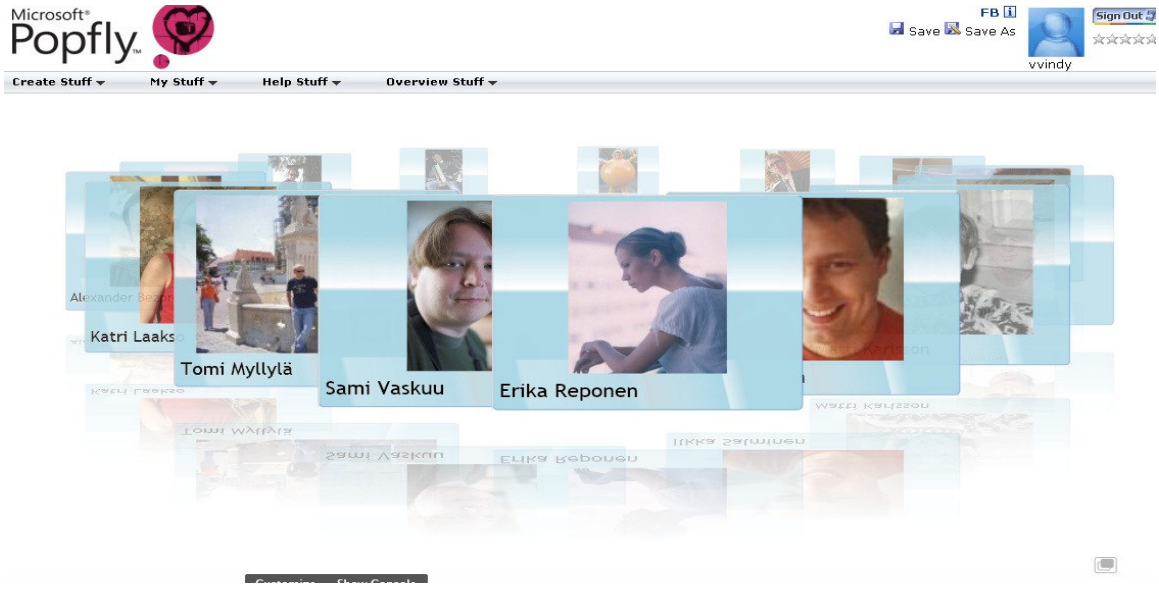

Figure 3.7: Popfly Mash-up output – image carousel with pictures of friends from Facebook.

The 3D graphics used gives the feeling that you are playing a computer game and not doing some time consuming programming task. Modules provide a recommendation about possible links with the other blocks thus directing users in the right direction. Users can choose from multiple options to visualize results. There are impressive visualization options for image sets – albums, carousels, and books (see Figure 3.7).

3.1.3.3 Google Mashup tools

Google (Google [n.d.]) decided on a different approach. Instead of launching an all-encompassing Mashup making application, they offer a multitude of tools - some, like Google Mashup Maker, aimed at developers, while others, e.g., MyMaps, can be used by anyone. Their impressive collection of APIs can be found easily using the Google search engine, and while they don't give many options of mash-up-creation to the layman, their openness has resulted in a huge number of great Mash-ups based on their services. Google Mash-up Editor (Google Mashup Editor [n.d.]) was "simple" if one is a developer and familiar with technologies like XML tags, JavaScript, CSS (W3C [n.d.]), and HTML (W3C [n.d.]). A Mashup application was described in a form of high level XML (W3C [n.d.]) based language that was interpreted by Google Mash-up engine. That service was in beta phase and was accessible by invited set of people only. The Google Mashup Editor documentation allowed classifying it as a software developer tool and not as an end-user Mashup environment. Therefore, we will not consider Google Mashup editor further on. Google Mashup Editor is not available any more.

3.1.3.4 Marmite

The Marmite (Wong and Hong, 2007) idea is that a mashup structure comprises sources, processors and sinks. Sources enable adding data into Marmite by querying databases, extracting information from web pages, and so on. Processors allow modifying, combining or deleting existing rows/columns (geocoding, filtering). Sinks then redirect the flow of data out of Marmite (showing data on a map, saving it to a file/web page). Since Marmite runs as a part of the web browser it provides an easy and visual way to specify a part of any web page as a "source" that can be processed by the tool. However, it is a research project meaning that functionality and implementation quality is much

lower compared to professional tools. Marmite is implemented as a plug-in for FireFox. It implements the data flow model, but papers on Marmite mention about the plans to support a spreadsheet model or mixed spreadsheet and dataflow model (Figure 3.8).

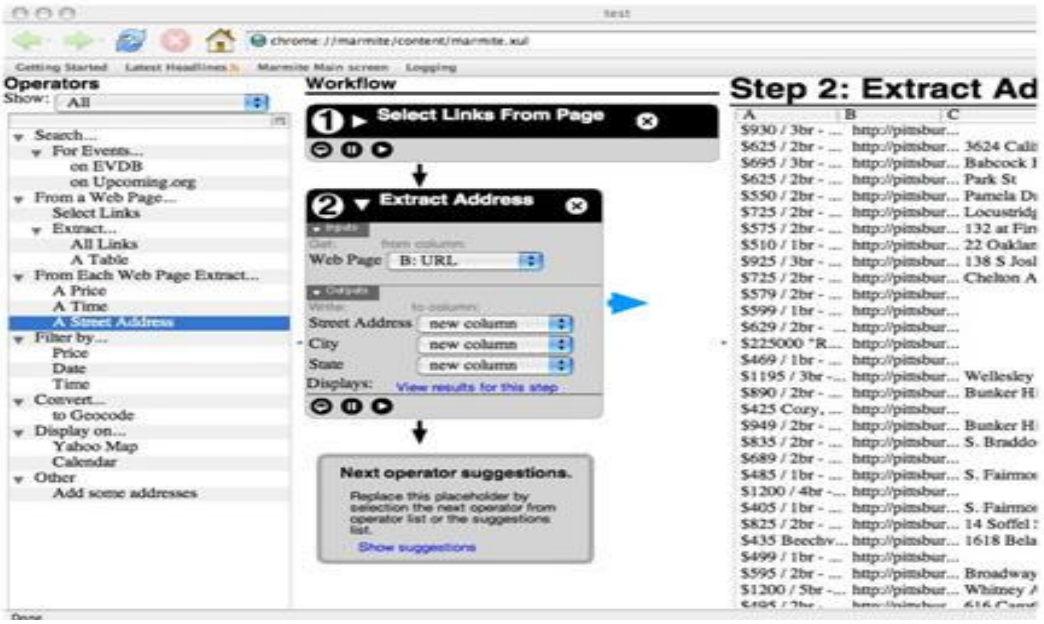

Figure 3.8: Marmite Mashup tool

2.1.3.5 Intel® Mash Maker

Intel® Mash Maker (Intel MashupMaker, [n.d.]) is provided as an extension to the web browser that allows a user to easily augment the page that he/she is browsing with information from other websites. As the user browses the web, the Mash Maker toolbar suggests mashups that it can apply to the current page in order to make it more useful for you. For example: plot all items on a map, or display the leg space for all flights. Intel® Mash Maker learns from the wisdom of the community. Any user can teach Mash Maker new mashups, using a simple copy and paste interface, and once one user has taught Mash Maker a mashup, this mashup will be automatically suggested to other users (Ennals and Gay, 2007). Intel® Mash Maker also relies on the community to teach it about the structure and semantics of web pages, using a built-in structure editor. There is no dedicated page on the web where you have to go and construct the mashup application. The user just has to install the toolbar in the browser and start browsing the

web. That plug-in supports multiple modes that give the opportunity either to use existing mashups or to define page structure if needed by turning your browser into a DOM explorer tool by opening new panes together with the main page (see Figure 3.9).

Figure 3.9: Intel Mash maker is integrated in the browser and can open extra panes around the main page.

The programming paradigm could be called “annotate and mix while browsing”. The service was in beta and not available to wide public at the moment of writing the thesis.

3.1.3.6 IBM QEDWiki

IBM QEDWiki (IBM [n.d.]) is a browser-based assembly canvas that can be used to create simple mashups. It utilizes the collaboration idea of a Wiki so that every change in the Wiki page is versioned. Mashup creation involves 3 steps: assemble, wire, and share. QEDWiki uses software components (or services) made available by content providers, e.g., QEDWiki easily integrates with widgets like EditGrid (IBM Mashup center [n.d.]).

It provides both web end-users and developers with a single web application framework for hosting and developing a broad range of Web 2.0 applications.

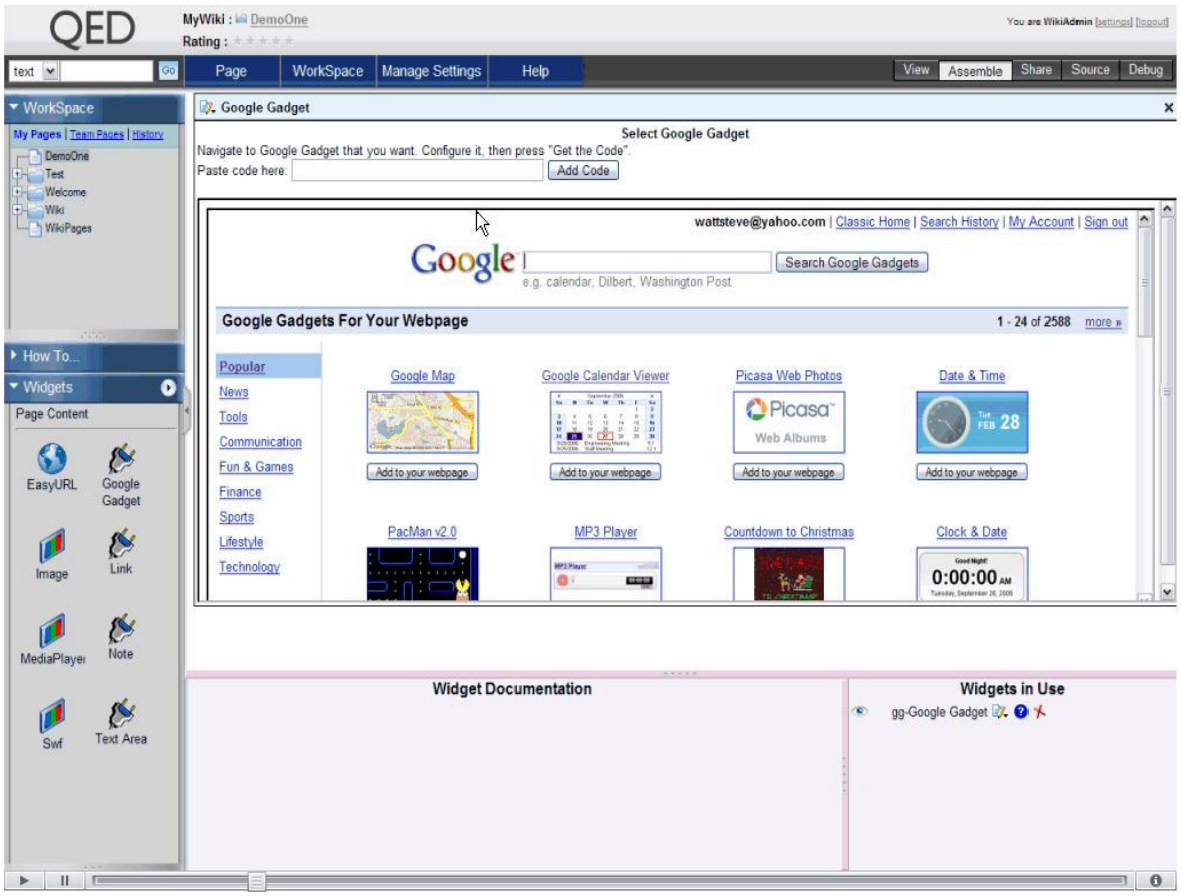

Figure 3.10: Composition screen of QEDWiki with list of tasks and Edit Grid placed on the canvas.

Here is the list of sample applications - Web content management for a typical collection of Wiki pages, traditional form processing for database-oriented CRUD (Create/Read/Update/Delete) applications, document-based collaboration, rich interactive applications that bind together disparate services, situational applications (or mashups). Mashup Creator places different components on the canvas and defines relations between them (see Figure 3.10). It is very close to the visual programming model that was supported by Visual Basic. Users define the layout of the application screen by dragging and dropping data widgets and extra services onto the canvas. There are functional widgets with no visual appearance that hook to widgets that are placed already on the canvas (e.g., send SMS module attaches to the address list).

3.1.3.7 Dapper –Yahoo!

Dapper (Dapper - Yahoo! [n.d.]) stands for Data mapper. The main purpose of the service is to convert any type of content into a standard form that can be reused (RSS, XML). It also has a set of publishing features that turn that content into Google Gadget (Google Gadget [n.d.]), Netvibes Module (Netvibes [n.d.]), iCalendar (iCalenar [n.d.]), Flash widgets (Flashwidgetz [n.d.]) and so on. It is a web application that visually runs is a wizard mode asking the user to fill-in some field at each step in order to create a "dapp" (data imported). The user interface is very minimalist, but it gets the things done. Dapps can be made public and indeed for popular services like YouTube and Flickr there is a huge collection of dapps available. Typically there is no need to create a separate dapp. In many cases dapp is a good candidate to be tuned into a map mashup or image loop. The user defines the output format or visualization type to use. The next level of development is to combine those dapps into an aggregator service. The typical example is to combine search result from several search engines or video clips from alternative video services similar to a movie aggregator (http://www.dapper.net/dapplications/Magg/) (see figure 3.11).

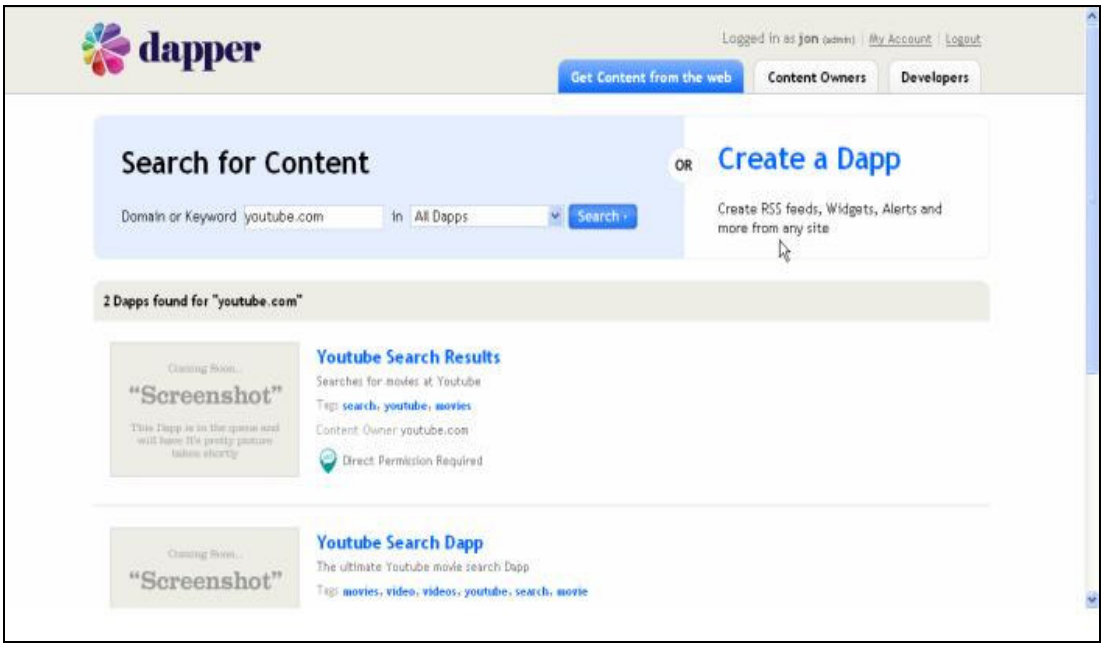

Figure 3.11: Dapper Data Mashup Maker

3.1.3.8 OpenKapow

OpenKapow (OpenKapow [n.d.]) is the perfect application for those who think Dapper is too simple. It works with the concept of "robots" - you download a desktop application called Robomaker that is used to gather data from websites. While Dapper is good at recognizing important data chunks on sites, Robomaker takes this to the next level, allowing you to automate complex processes and simulate a real person's behaviour in a web browser to retrieve the data you need. You can then create three different types of robots - RSS, REST or Web Clip robots, which enable you to either create RSS feeds, create an API out of a website or simply collect one piece of functionality from a site and use it somewhere else. All this makes OpenKapow a good tool for serious mashup-making, which will mostly be used by developers to aid them in their work (figure 3.12).

Figure 3.12: OpenKapow

3.1.3.9 Potluck

Potluck (Huynh et al, 2007) is a tool that lets casual users—non-programmers—make mashups.

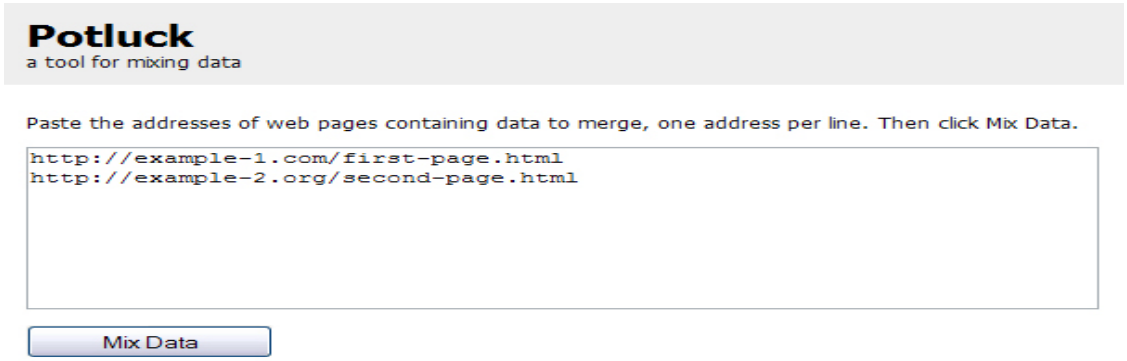

Figure 3.13: The starting screen of Potluck takes URLs to Exhibit-powered web pages. Clicking Mix Data yields the mixed data in a screen like in figure 3.13.

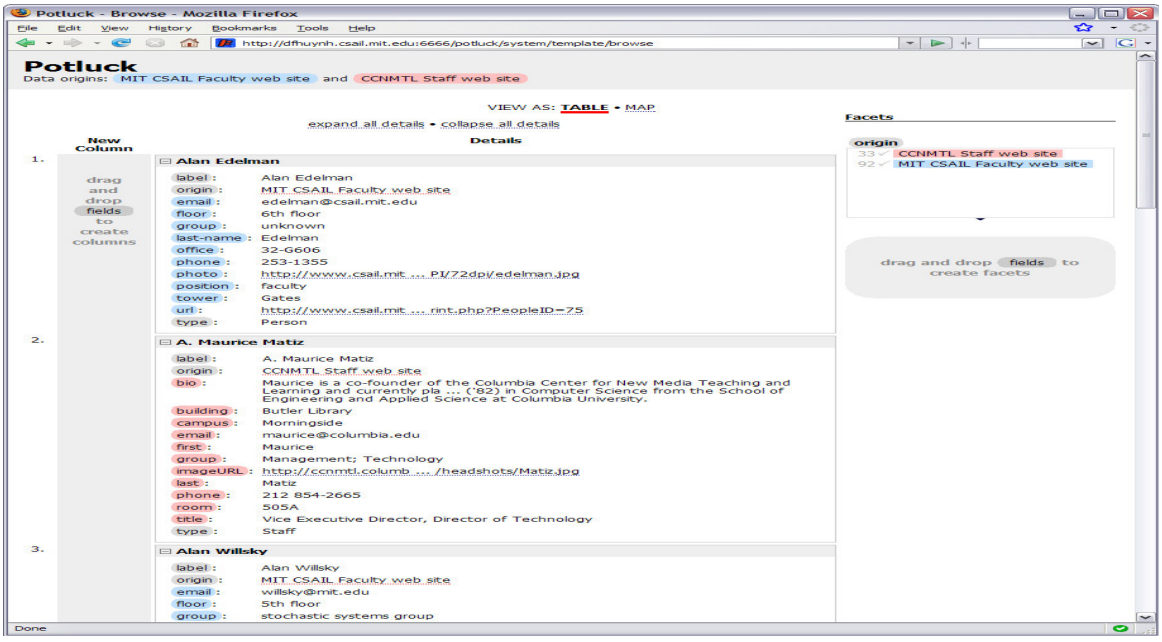

Figure 3.14: Potluck's user interface shows data that has just been mixed together but not yet processed by the user. Fields are rendered as draggable "field tags," color-coded to indicate their origins. There are two drop target areas for creating columns and facets.

Potluck allows the user to merge fields from different data sources, so that they are treated identically for sorting, filtering, and visualization. Fields are merged using simple drag and drop of field names. Potluck provides an efficient means for the user to clean up data syntactically, homogenize data formats, and extract fields syntactically embedded within existing fields, all through the application of simultaneous editing (Huynh et al, 2007). Potluck supports faceted browsing (Huynh et al, 2007) to let users explore and identify subsets of data of interest or subsets of data that need alignment and clean up. Figures 3.13 and 3.14 show the interface of the Potluck.

3.1.3.10 Open Mashup studio

Open Mashup studio (OpenMashups studio, [n.d.]) was a Web Mashup design tool that could be downloaded from the site and running as online desktop software (see figure 3.15).

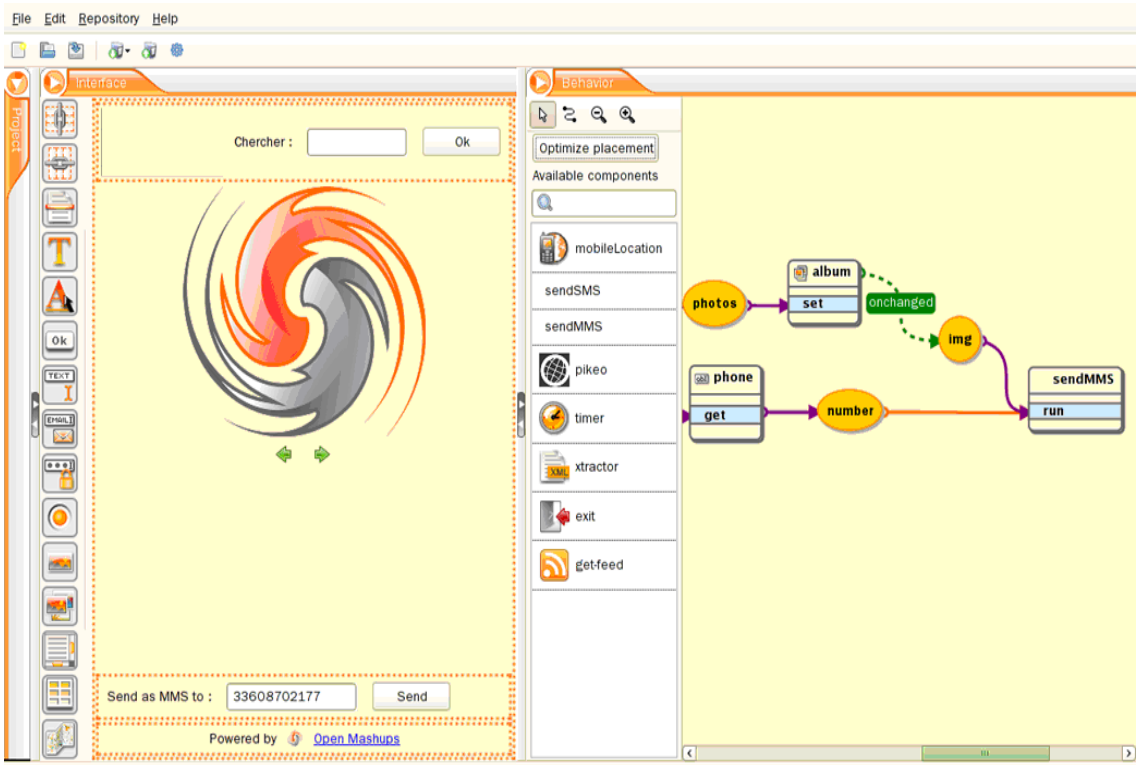

Figure 3.15: Open Mashup Studio.

Open Mashups studio offered some answers to end-user requirements by applying, in an open web context, rigorous formal techniques from the MDA/MDE (Model Driven

Architecture/Engineering) world. This is defined as a dedicated meta-model that represents an application from a very functional and declarative standpoint. This meta-model is the cornerstone of the tool approach: Open Mashup studio provided an easy to use graphical editing environment to depict application models (both for the graphical user interface and the behaviour of the application), and write code generators that produce executable code from these models. This approach allows hiding the complexity from the Mashup creator who only needs to understand and manipulate a limited number of concepts, relying on code generator implementers to take care of platform and device adaptation (Open Mashup studio [n.d.]). Open Mashup studio is not available any more.

3.1.3.11 Other Mashup creation tools

Creo (Faaborg and Lieberman 2006) augments web pages with additional links that can obtain additional information about items on a web page. Like Intel MashMaker, Creo will automatically make suggestions and can learn by example from things that users do with their data. Unlike MashMaker, Creo is limited to adding additional hyperlinks to web pages and cannot perform bulk data processing tasks (Ennals and Gay, 2007).

ClearSpring.com, **Widsets.com**, **WidgetBox.com**, and **Apple's Dashboard**, are Mashup design envirements that allow users to write small graphical web widgets and then lay them out together on a screen (Ennals and Gay, 2007).

DataMashups.com additionally allows users to connect these widgets together (e.g., the output of this widget is the input to that widget), but complex tasks require considerable programmer skills. Currently datamashups.com has been acquired by vtiger.com and is not any more available on the Internet, (Ennals and Gay, 2007).

HunterGatherer (Schraefel et al., 2002) and **Internet Scrapbook** (Sugiura and Koseki, 1998) allow users to extract parts of multiple websites and composite them together, but are not able to perform complex processing on these sites and extract collections of data. (Ennals and Gay, 2007).

3.1.4 Different Composition Approaches for Mashup Makers

Wajid et al. (Wajid et al, 2010) have defined three main approaches for Mashup Makers and web composition environments. The three approaches are: Control Flow Approach (CFA), Data Flow Approach (DFA), and Assisted Composition Approach (ACA). In a *Control Flow Approach,* the Mashup creation process is a matter of sequential composition of components where one task is required to be completed before the next task can be executed. In a *Data Flow Approach,* data is passed between multiple components without the requirement of a specific sequence; in such an approach the Mashup Maker allows the user to define how data flows from source to destination. An *Assisted Composition Approach* enables a user to choose among tasks; there is no need to define control or data flows among components and the sequences are managed automatically.

De Angeli et al. (De Angeli et al, 2011) have investigated a simplified approach for mashup composition/development that is called WIRE. The idea behind the WIRE naming comes from the most famous and existing Mashup development tools such as Yahoo! Pipes and Open Mashup Studio. Those Mashup Makers are providing environments in which the user needs to wire components on the design area in order to have Web application as a real output.

Picozzi et al. (Picozzi et al, 2010) have identified two basic approaches for mashup development: the first is the *manual approach* where the end-user is programming-skilled and therefore is able to write code to program components and their choreography. And the second is the *automatic approach* where the end-user is not programming-skilled and has only to integrate ready-to-use components that expert developers have previously programmed. S/he uses a tool that simplifies the composition of the Mashup.

In table 3.1 we present an overview of the characteristics of 10 well-known Mashup Makers for casual end-users. We have based our overview on the information we collected in the literature study and presented in section 3.1. The first criterion is the audience the Mashup Maker claims to target. The second criterion is the Mashup composition approach used (WIRE, cascading of multiple steps, or web page scrapping

(see subsection 3.1.4)), i.e. how an end-user needs to compose a Mashup. The third criterion is the data and control flow approach used by the Mashup Maker. This crietrion is related to definition of Wajid et al. (2010) (see subsection 3.1.4). The fourth criterion indicates if the Mashup maker is still available or not. And the last column indicates if help, documentations, and tutorials are provided by the Mashup maker.

Table 3.1: An overview of Mashup makers' characteristics

Mashup Maker	**Target audience**	**Mashup composition approach**	**Data & control approach**	**Current availability**	**Provides help, tutorials, API documentation**
Yahoo! pipes	Casual end-users and Web skilled people	WIRE	Data flow	Available	Yes
Microsoft Popfly	Casual end-users and Web skilled people	WIRE	Data flow	Not available anymore	Yes
Google Mashup tools	Diverse	Diverse	Diverse	Available	Diverse
Marmite	Casual end-users and Web skilled people	Browser plug-in, allows scrapping web pages, spreadsheet	Data flow	Not available anymore	Partially
Intel Mashmaker	Casual end-users and Web skilled	Browser plug-in, web pages scrapping, Browsing	Data flow	Not available anymore	Yes

	people	enrichment			
IBM QEDWiki	Casual End-user	WIRE, Browsing enrichment, Application on canvas,	Data flow	Not available anymore – directed to business as IBM MC	Yes
Dapper - Yahoo!	Casual end-user	Cascading multiple steps, web pages scrapping, wizard	Control flow	Available but bought by Yahoo!	Yes
OpenKapow	Casual end-users and Web skilled people	WIRE, web pages scrapping, application on canvas	Data flow	Payable and business oriented	Yes
Potluck	Casual end-user and Web skilled people	Faceted browsing, Web page scrapping	Assisted flow	Not available anymore	Partially
Open Mashup studio	Web skilled people and Web developers	WIRE, application on canvas	Data flow	Not available anymore	Partially

3.2 Usability Evaluation of Mashup makers

3.2.1 Definitions of Usability

As mentioned earlier in Chapter 1, this research concerns the investigation of the usability evaluation of Mashup Makers for end-user. Usability evaluation of software systems is one of the research tracks of Human-Computer Interaction (HCI). HCI is the research area that studies the interaction between people and computers. It involves the design, implementation and evaluation of interactive systems in the context of the user's task and work (Dix et al, 2003)

There are different definitions of the concept 'usability'. For example, usability as defined by ISO 9241 part 11 (ISO9241, Part11 [n.d.]) is "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use". Abran (Abran et al, 2003) extended this standard ISO definition to include Learnability and Security.

The usability concept is also defined in ISO/IEC 9126 (1998). This definition is widely accepted (Hornbaek, 2006). According to this definition, usability refers to "the capability of the (software) product to be understood, learned, used and be attractive to the user, when used under specified conditions." The definition is focusing on software systems in general.

Further usability definitions are found in the context of the user centric perspective like the one of Ham et al. (2006). Usability is a concept that needs to be evaluated from a user-centric point of view. User perception of usability is influenced by many design factors including visual appeal, hedonic qualities, logical task sequences, pleasure in use, as well as contextual factors including the users' environment (i.e. context of use) (Ham et al, 2006).

Nielsen (2003) indicated that usability is one of the most important attributes of any user interface and according to Nielsen it measures how easy the interface is to use. Others have indicated that: "Usability measures the quality of a user's experience when

interacting with a product or system, whether a website, a software application, mobile technology, or any user-operated device" (Usability.gov [n.d.]). Nielsen (2003; 1993) also stated that usability is not a single attribute; instead usability is defined in terms of five characteristics:

- **Learnability**: The system or product is easy to learn so that users can perform tasks the first time they interact with the interface.
- **Efficiency**: The system or product is efficient to use so that once users have learned the system, they will perform tasks quickly.
- **Memorability**: The system or product is easy to remember so that if users return to the system after a period of not using it, they can use it easily.
- **Errors**: The system or product has a low error rate so that users make few errors while interacting with it and they can easily recover from these errors.
- **Satisfaction**: The system or product is pleasant to use and users are subjectively satisfied while using it.

Alternatively, (Brinck et al, 2001) defined usability as "the degree to which users can perform a set of required tasks". They also indicated that usability is the product of several design goals, including the five attributes already indicated by (Nielsen, 2003; 1993), in addition to another goal named 'functionally correct'. This attribute means that the system or product provides the required functionality so that users can do what they need/want to do. (Brink et al, 2001) explained that the design goals of usability are sometimes in conflict and therefore the priority given to these design goals is determined with regard to the context of the design. (Sharp et al, 2007) added effectiveness and safety to the list of usability design goals/attributes; effectiveness means that the system or product is effective to use and good at doing what it is supposed to do so that users can carry out their work accurately and successfully; safety means that the system or product is safe to use so that it protects users from dangerous conditions and undesirable situations.

3.2.2 Usability Evaluation Methods

Usability evaluation methods are a set of methods used to evaluate the human computer interface provided by a product. They are aimed at identifying issues or areas of improvement in the interaction between the user and the system or in the interface in order to increase usability (Gray and Salzman, 1998). These methods are one of the hallmarks of User-Center Design (UCD) (Lazar, 2006). UCD is an approach and philosophy for designing and developing usable products and systems that place the user at the centre of the development process (Rubin, 1994). The UCD approach is based on receiving user feedback during each step of the design process (Rubin, 1994). Obtaining such feedback can be done by a variety of usability methods at each step of the design and development process (Pearrow, 2000), (Rubin, 1994).

Several usability evaluation methods and techniques have been developed to identify and analyse usability problems. There are different approaches for categorizing those usability evaluation methods. For example, Nielsen and Mack (1994) classify usability evaluation methods into four general categories: automatic (this involves the use of software to evaluate a user interface), empirical (involving real users who interact with a user interface), formal (incorporating the use of models to evaluate a user interface), and informal (where evaluators use rules in addition to their skills, knowledge and experience to evaluate an interface). Alternatively, Gray and Salzman (1998) divided evaluation methods into two main categories: analytic and empirical. The analytic methods include techniques such as heuristic evaluation and cognitive walkthrough, while empirical techniques include methods and procedures referred to as user experiments and testing.

In the following subsections we organise the most well-known usability evaluation methods into three categories in terms of how the usability problems were identified: by evaluators, by users, or by tools. This classification was introduced by Hasan et al. (2011). We found that this classification of usability evaluation methods is the most appropriate for our research work on usability evaluation of Mashup Makers.

3.2.2.1 Evaluator-Based Usability Evaluation Methods

This category includes usability methods that involve evaluators in the process of identifying usability problems. These methods are called usability inspection methods by Nielsen and Mack (1994), who defined these as a set of methods based on having evaluators inspecting or examining the usability aspects of a user interface. These methods are aimed at (1) finding usability problems that users might encounter while interacting with an interface and (2) making recommendations to improve the usability of the interface. The following are some of the most well-known methods in this category, which can be used to evaluate the usability of a user interface, including websites.

Heuristic Evaluation

Heuristic evaluation is a usability method developed by Nielsen and Molich (1990). This method involves having a number of evaluators assessing the user interface and judge whether it conforms to a set of usability principles ('heuristics') (Nielsen and Molich, 1990). In (Nielsen, 1994), Nielsen identified a set of 10 usability heuristics which were: visibility of system status, match between the system and the real world, user control and freedom, consistency and standards, error prevention, recognition rather than recall, flexibility and efficiency of use, aesthetic and minimal design, helping users to recognise, diagnose and recover from errors, and help and documentation.

Some researchers, however, indicated that the original set of heuristics developed by Nielsen were too general and too vague for evaluating new products such as web products because they were designed originally to evaluate screen-based products; they were also developed several years before the web was involved in user interface design (Sharp et al, 2007), (Pearrow, 2000), (Brinck et al, 2001). Consequently, new heuristics were developed specifically for evaluating websites. For example, in (Nielsen, 2000), Nielsen suggested the following heuristics which he called HOMERUN: high quality content, often updated, minimal download time, ease of use, relevant to users' needs, unique to the online medium, and adhering to net-centric corporate culture. However, despite the criticism on Nielsen's 10 heuristics, it is worth mentioning that researchers

advised including them as part of the design guidelines to evaluate usability of websites (Brinck et al, 2001), (Sharp et al, 2007). For example, in (Sharp et al, 2007), Sharp et al. advised evaluators who might wish to develop specific heuristics to evaluate websites to develop their own, by tailoring Nielsen's heuristics and by referring to other resources, such as design guidelines, market research and new research findings.

Pluralistic Walkthrough

The pluralistic walkthrough is a usability inspection method that involves a group of evaluators, including representative users, developers and usability experts, evaluating a user interface by "walking through" the steps of a task scenario (Hollingsed and Novick, 2007), (Nielsen and Mack, 1994). The group discusses the usability issues of an interface related to each step in a scenario (Nielsen and Mack, 1994). The scenarios are presented in the form of a number of screens which represents a single path through the interface (Sharp et al, 2007). As indicated by Hollingsed and Novick (2007), this method is defined by five characteristics: the involvement of various participants: representative users, developers and usability specialists; the interface screens are displayed during the evaluation in the same order in which they would be displayed in a web or computer interface; all the participants are asked to assume the role of a user; for each screen, participants write down what actions they, as users, would select in performing the task and add their feedback in detail; finally, during the discussion of each screen, the representative users are those who speak first. One of the benefits of the pluralistic walkthrough is related to the fact that it provides feedback from users who are directly involved in the evaluation (Hollingsed and Novick, 2007). Another benefit is that it focuses on users' tasks (Sharp et al, 2007). However, this method also has its limitations: for example, it is difficult to get all the participants together at the same moment and then work at the rate of the slowest (Sharp et al, 2007). Also, only a few scenarios, and therefore paths through the interface, can usually be investigated because of time constraints (Sharp et al, 2007), (Hollingsed and Novick, 2007). Research, which has investigated the use of this method, notes that this method is still used as a usability

expert/inspection approach although usability experts continue to perform users-only walkthrough without their involvement (Hollingsed and Novick, 2007).

Cognitive Walkthrough

Cognitive walkthrough is a usability inspection method that focuses on evaluating whether an interface is easy to learn through exploration (Wharton et al, 1994). This method still appears to be in continual use although it was developed in the early nineties, because of its effectiveness; it is used in the evaluation of different interfaces including web-based applications (Hollingsed and Novick, 2007). This method involves a team of evaluators who evaluate an interface by "walking through" one or more specific representative tasks and their related steps/actions, step-by-step. The team usually involves developers, designers and programmers (Fichter, 2004). For each step, the team attempts to offer a reasonable response or "story" to each of four questions determined by (Wharton et al, 1994) (see Table 3.2) explaining why users would choose the correct action to perform the task (Spencer, 2000), (Fichter, 2004). If the story cannot be told then suggestions for correcting the problems are noted (Fichter, 2004).

Table 3.2: Four questions from Wharon et al. (Wharton et al, 1994).

Will the user try to achieve the right effect?
Will the user notice that the correct action is available?
Will the user associate the correct action with the effect that user is trying to achieve?
If the correct action is performed, will the user see that progress is being made toward the solution of the task?

The cognitive walkthrough method is useful for obtaining a large number of design ideas from the team members who usually have different backgrounds and perspectives (Fichter, 2004). Also, this method focuses on users' problems in detail even though users do not need to be involved (Sharp et al, 2007). However, the major drawback of this method relates to the fact that it can be time consuming and tedious (Fichter, 2004),

(Holzinger, 2005). Furthermore, the selection of task scenarios can be difficult since, if the scenario is not appropriately described, then this results in an ineffective evaluation (Hollingsed and Novick, 2007). It is worth mentioning that Spencer (2000) suggested a modified cognitive walkthrough process called a streamlined cognitive walkthrough because he indicated that the original cognitive walkthrough method might be difficult to use in the evaluation of software in a large software development company. This is because of the social constraints faced by team members in the company such as time pressure, very long discussions concerning the design, and the fact that some team members might try to defend their design during the cognitive walkthrough process. Therefore, the suggested streamlined cognitive walkthrough can overcome such social constraints and provide useful data. This can be achieved by avoiding design discussion, defusing design defensiveness, and streamlining the method and data collection (Spencer, 2000). The streamlined cognitive walkthrough method uses only two questions, instead of the four questions suggested by (Wharton et al, 1994), in the evaluation of each step in the task analysis. See Table 3.3.

Table 3.3: Two questions from Spencer (Spencer, 2000)

Will the user know what to do at this step and if he/she has done the right thing?
Will the user know that he/she has done the right thing and is making progress towards his/her goal?

Guideline Reviews

This is a usability method which contains comprehensive guidelines and involves checking an interface for conformance with these usability guidelines. This method is similar to the heuristic evaluation method, except for the length and details of the guidelines used by evaluators; heuristic evaluators use a short list (of less than a dozen items) while guideline reviewers use a longer and more detailed list (with several dozen or more guidelines) (Lazar, 2006), (Gray and Salzman, 1998). Some organizations and companies have specific design guidelines (e.g., Microsoft design guidelines for Windows O.S) which can include hundreds of design rules (Lazar, 2006). Therefore, this

kind of review takes a long time to accomplish and hence is not commonly performed, in contrast to the heuristic review (Lazar, 2006).

Consistency Inspections

This is a usability method where an expert reviews all of the web pages on a site to ensure that its design is consistent in terms of layout, terminology and colour (Lazar, 2006). This method could also be used to inspect consistency across multiple sites, examining, for example, whether common functions look and work in the same way across these sites (University of Minnesota Duluth [n.d.]). The consistency of an interface is important because inconsistent interfaces could reduce users' performance and satisfaction, thereby increasing the error rate, as indicated by Lazar (Lazar, 2006).

Standards Inspection

Standards inspection is a usability method that involves an expert examining whether an interface complies with certain interface standards which are followed by other systems in the same market (Nielsen and Mack, 1994). The standards are usually written in formal language and therefore, in order to perform this type of inspection, an expert who is familiar with the standard and its language is required (Stone et al, 2005). An example of a usability standard that can be used as a reference is the ISO 9241 (ISO9241, Part11 [n.d.]), (Stone et al, 2005). ISO 9241 includes requirements and recommendations regarding the attributes of the hardware, software and the environment, which contribute to their usability and the ergonomic principles relating to them (Cost-Effective [n.d.]).

3.2.2.2 User-based Usability Evaluation Methods

This category includes a set of methods that involves users. These methods aim to record users' performance while interacting with an interface and/or users' preferences or satisfaction with the interface being tested. The most common method in this category relates to user testing. The other methods are either variations of a user testing approach or supplementary techniques that could be used with a user testing method. The following section presents the most common methods in this category which can be used to evaluate the usability of interfaces:

User Testing

The user testing method is considered to be the most important and useful approach since it provides direct information regarding how real users use the interface; it illustrates exactly what problems users' encounter in their interaction (Nielsen and Mack, 1994). In (Dumas and Redish, 1999), Dumas and Redish defined the user testing method as "a systematic way of observing actual users trying out a product and collecting information about the specific ways in which the product is easy or difficult for them". Different supplementary techniques have been suggested for use during a user testing session, such as making different types of observation (e.g., notes, audio, video, or interaction log file) to capture users' performance; questionnaires and interviews have also been suggested as ways of collecting data concerning users' satisfaction (Nielsen, 1993); (Sharp et al, 2007); (Dumas and Redish, 1999); (Rubin, 1994). Capturing user performance can be automated using tools such as Camtasia. Camtasia is a screen capture software package that has proved to be an effective tool for capturing website usability data (Goodwin, 2005). Such a tool records users' activities on screen (i.e. users' actions and movements that take place on the computer screen); it may also have the capability to record users' voices along with their actions if a microphone is used (Goodwin, 2005). This therefore helps to reduce the workload of the observer during the user testing session.

Think-Aloud Method

This is a user testing method with a condition: the condition of asking users to think aloud during their interaction with an interface (Lazar, 2006), (Nielsen, 1993). Nielsen (1993) indicated that having users verbalising their thoughts using this method allows an understanding of how users view or interpret an interface; it also facilitates the major misconceptions of users to be identified. Holzinger (2005) indicated that this method might be the most valuable usability testing method. However, the think-aloud method has some disadvantages related to the fact that the test setting, with an observer and recording equipment, will not represent a natural setting; this therefore will not encourage users to act and talk naturally (Van den Haak and De Jong, 2005).

Constructive Interaction (also known as co-discovery learning)

This method is a think-aloud method with one condition: the condition of having two users (instead of a single user) interacting with an interface together or working together to complete specific tasks (Holzinger, 2005), (Nielsen, 1993). The main advantage of employing this technique is that the test situation is much more natural in comparison with the think-aloud tests because people are used to verbalise their thoughts when trying to solve a problem together (Holzinger, 2005), (Nielsen, 1993). Therefore this technique is an appropriate usability testing method for testing an interface if the users are children because it is difficult for children to follow the standard think-aloud method (Nielsen, 1993). Holzinger (2005) indicated that by using the constructive interaction method, more comments might be obtained from users in comparison to the think-aloud method. This method is most suited in situations where it is easy to obtain a large number of users and where it is comparatively cheap for users to be recruited because it requires twice as many test users as the single-user thinking aloud technique (Nielsen, 1993). However, the unnatural settings which are associated with the think-aloud method also constitute one of the drawbacks of the constructive interaction method. It is worth mentioning that, despite the difference in the number of participants between the think-aloud and constructive interaction methods, research has found that these methods provided similar results in terms of the number and type of problems identified (Van den Haak et al, 2004) These results therefore would encourage the think-aloud method to be employed in preference to the constructive interaction approach since the latter incurs the cost of recruiting the second participant to obtain the same results (Van den Haak et al, 2004).

Retrospective Testing

This is a user testing method that involves video-recording users' sessions and then collecting their comments while reviewing the recording (Lazar, 2006), (Nielsen, 1993). While users are reviewing the tape, they may provide additional comprehensive comments in comparison to comments they made when working on the tasks; the experimenter can also stop the tape and ask users for more detailed information (Nielsen, 1993). This method has the advantage of gaining more information from each test user as indicated in (Nielsen, 1993). However, this method takes at least twice as long. Therefore

it is not suitable for use if the users are highly paid or perform critical work which means that they are unable to spend long on the activity (Nielsen, 1993).

Questionnaires and Interviews

Different types of questionnaire (i.e. closed or open) and interviews (i.e. unstructured, semi-structured or structured) are considered useful and simple techniques that collect data regarding users' satisfaction with, or preferences on, a user interface (Bidgoli, 2004); (Sharp et al, 2007); (Rubin, 1994). These could be used as supplementary techniques to the user testing method or they could be used alone. However, if these techniques are used alone then they are considered as indirect usability methods because they do not study the user interface directly; instead, they reflect users' opinions about that interface (Holzinger, 2005), (Nielsen, 1993). Dumas and Redish (1999) also indicated that surveys cannot be used to observe and record actual users' interactions with an interface but can be used to collect information regarding users' opinions, attitudes and preferences, as well as self reported data concerning behaviour. Therefore, data about users' actual behaviour should have precedence over users' preferences since users' statements cannot always be taken at face value (Holzinger, 2005). Furthermore, these techniques have other disadvantages: for example, a sufficient number of responses are needed to obtain significant results in the case of questionnaires (Holzinger, 2005). Interviews can also be very time consuming for both the interviewer and the participants, and the quality of the information that is collected depends on the interviewer's experience in performing interviews (Lazar, 2006). It is worth mentioning that using e-mail and online questionnaires allow preference data to be gathered quickly from small or large and/or dispersed users (Bidgoli, 2004), (Macro, 2000). However, the response rate for questionnaires is typically low (Bidgoli, 2004).

Focus Groups

This is an informal method for collecting in-depth information regarding the needs, judgments and feelings of typical users about an interface (Nielsen, 1993), (Rubin, 1994), (Dumas and Redish, 1999). In a focus group, about 6 to 9 users discuss selected topics, such as the different functions and features of an interface, with the assistance of a

moderator, and then identify issues during their interaction. This method allows diverse and relevant issues to be raised; it brings out users' spontaneous reactions, comments and ideas through their interaction (Sharp et al, 2007), (Nielsen, 1993). For example, it can provide information regarding what functions of the interface have problems or are undesirable; it also allows discussion concerning how these problems can be solved (Bidgoli, 2004). However, although this technique captures users' opinions and satisfaction, it does not measure users' actual interactions with an interface (Macro, 2000), (Nielsen, 1993), (Dumas and Redish, 1999). The focus group can also be conducted online and this can provide the same information as a face-to-face focus group (Macro, 2000). Online focus groups have the advantage of eliminating distance and travel costs for both participants and the moderator and enables information from participants from different geographical locations to be collected (Macro, 2000). However, participants must have computer access and a basic level of computer literacy; also, the moderator may not be able to observe the facial expressions and body language of the group participants (Macro, 2000).

3.2.2.3 Tool-based Usability Evaluation Methods

Rather than employing experts or users to evaluate the usability of an interface, software tools can be used to do this. The following section presents these methods.

Software Tools: Automatic Usability Evaluation

This method is related to tools that automatically assess whether a website conforms to a set of specific usability guidelines (Brinck et al, 2001). Most of these tools assess the quality of the HTML code of a website with regard to a number of guidelines. For example, they check if the images on a website's pages include the ALT attribute. Therefore, these tools are similar to the expert review/inspection methods (Lazar, 2006). Most tools focus on the accessibility of a site rather than its general usability (Lazar, 2006). One of the best known tools is Bobby (Stone et al, 2005). The original Bobby tool was a free public web accessibility testing tool provided by the Centre for Applied Special Technology (CAST). It examined the source of a site to check its compliance with accessibility guidelines including Section 508 of the U.S Rehabilitation Act and the

W3C'c Web Content Accessibility Guidelines. Later, in 2004, Bobby software was sold to Watchfire which provided the same free service in the WebXACT tool (Wikipedia [n.,d.]). However, Watchfire was then acquired by IBM in 2007 and consequently, in 2008, the Bobby tool was discontinued as a free tool or standalone product (Wikipedia [n.d.]; CAST [n.,d.]). It is now one of the tests included within the IBM Rational Policy Tester Accessibility Edition software (IBM [n.,d.]).

Software Tools: Transaction Log File and Web Analytics Tools

The transaction log file is related to tools that automatically collect statistics regarding the detailed use of systems, including websites. The server log file was developed originally to capture technical information concerning server performance (i.e. server error (404 error)) (Kaushik, 2007). This method is also considered as an indirect observation method which helps to analyse users' behaviour and which allows researchers to understand how users have worked on the tasks (Sharp et al, 2007). Researchers suggested that the log file could be used as a supplementary technique to the user testing method or it could be used alone to collect data concerning the usage of system for a specific period (Nielsen, 1993), (Sharp et al, 2007), (Dumas and Redish, 1999). However, as log files, specifically web server log files, started to get larger and non-technical people became interested in the data captured by such files, scripts were programmed that automatically analysed the large-sized log files and thus web analytics tools were officially born (Kaushik, 2007). The first documented log analyzer (GetSites) was written in June 1993 at Honolulu Community College (Website Measurement [n.d.]). The log file is one of the most common data sources of web analytics; however, there are other sources used by these tools such as page tagging (JavaScript tagging) and network based approaches.

3.2.3 Effectiveness of Usability Evaluation Methods for Mashup Makers

Usability evaluation is an essential activity for securing highly usable software products, which should be conducted during all the phases of design life cycle (Kangas and Kinnunen, 2005). Various usability evaluation methods and frameworks have been developed and can be classified into three types: usability testing, usability inquiry, and

usability inspection (Karat, 1997; Zhang, 2003). In section 3.2.2 of this chapter we presented the most known and famous usability evaluation methods. Although they have proven their usefulness, several comments can be made.

Earlier studies agreed that despite the fact that usability evaluation methods have a similar aim, which is to identify usability problems that prevent users from interacting easily with an interface, these methods varied with regard to the number and type of problems identified by them and the cost of employing these methods (Hasan, 2009).

These evaluation methods have resulted in the development of guidelines (Koyani et al, 2003) for desktop computing applications that have had a positive effect on the implementation of more usable systems. Guidelines allow developers to take the guesswork out of initial interface design.

As Mashup Makers are visual information systems, it worth's to mention what Scholtz (2006) highlighted regarding visual systems usability evaluation: "Evaluation aspects of visual analytical environments need to include usability, but it is necessary to go beyond basic usability". With this Scholtz means that it is necessary to get user's feedback of the ease of use to deeply investigate user's interactions with information systems.

Furthermore, the existing usability evaluation methods are not specific for Mashup Makers and therefore may neglect some important usability issues typically for Mashup Makers. In addition, a complete and comprehensive usability evaluation framework that ties together usability methods and techniques and that is specific dedicated to end-user development and more specific to Mashup Makers is lacking.

3.3 Related Work

While our literature research has found that there is no complete and comprehensive work about Web Mashup usability at the time of writing this thesis, our investigations have found two main research tracks dealing with usability of Web Mashup Makers. The first track is research work on the usability of Mashup Makers in general; the second

track deals with studies and usability experiments for a particular Mashup Maker. We start with an overview of the research in the first track.

Oleg Beletski (2008) reviewed, in an internal report, some Web Mashup programming environments and compares basic usability aspects of those environments. The report has summarized the usability aspects of the Web Mashup programming environments (tools) compared, by simply mentioning whether they are easy to use or not. The author has not mentioned any verification process for his measurements.

In (Grammel and Storey, 2008) and (Grammel and Storey, 2010), Grammel and Storey reviewed 6 Mashup Makers from the so-called End User Development perspective. The authors based their research methodology on some selected dimensions of the CD´s framework (Green et al, 1996), software engineering techniques and some concepts related to e-learning. We have tried to fairly examine their report regarding usability matters but we found it undetermined and in our opinion, it lacks specific usability review points.

In several papers (Zang and Rosson, 2008), (Zang et al, 2008), (Zang and Rosson, 2009), Nan Zang et al. present a Yahoo Pipes use survey and, in a sort of study with an end-user programming experiment, they try to explore key players between Mashup users as well as their activities. Also here, we found the research not yet mature from a usability perspective and needs more emphasis on usability and different Mashup tools.

Exploring Usability Guidelines for RIA (Gwardak and Påhlstorp, 2007) is a master thesis in which the authors used desktop usability guidelines and web usability guidelines as a basis to create an outline of Rich Internet Application (RIA) usability guidelines. Most of their work was focused on a comparative study of general usability guidelines. In conclusion they formulated some so-called start guidelines for developers in the field of RIAs. Although interesting and somewhat related, our research will focus on usability of Web Mashup Makers and will be based on a usability study, experiments and usability testing.

We now review the work in the second track, being studies and usability experiments on a particular Mashup Maker usually performed by the development team of the Mashup Maker.

Potluck (Huynh et al, 2007) is a project at the Computer Science and Artificial Intelligence Laboratory (MIT, USA). It aims at the development of an easy to use tool to Mashup data for casual users. They performed a usability evaluation study to ascertain whether people could learn how to use Potluck as well as to discover usability problems. Their study consisted two tasks: a structured task and an unstructured task. We have learned a lot from their experience and we will follow some of their notes related to the usability evaluation of Mashup tools.

Intel Mash Maker (Ennals and Gay, 2007) is a research project at Berkeley University (USA) funded by Intel. Mash Maker is a web-based tool to create Web Mashups by browsing around, without needing to type, or plan in advance what you want to do. The research team of Mash Maker has performed a usability evaluation of the tool following and using the Cognitive Dimension of notations (CDs) framework (Green et al, 1996). This evaluation (Zang and Rosson, 2008) has helped us directing our experiments on usability of web Mashups tools.

Marmite (Wong and Hong, 2007) is a research project of the HCI Institute of Carnegie Mellon University, USA. It is an end-user programming tool which lets users create Mashups that repurpose and combine existing web content and services. Marmite is targeting users with programming backgrounds and with spreadsheet skills. The development team of Marmite has performed a usability evaluation study which showed some difficulty for some users and the team intended to improve the usability aspects of the tool in the next versions. The Marmite usability evaluation study has helped us understanding a new way of evaluating Mashup development tools, which we will also consider in our usability study and experiments of Mashup development tools.

In several research tutorials (Namoun et al, 2010a), (Namoun et al, 2010b), (Namoun et al, 2010c), (Mehandjiev et al, 2010a), (Daniel et al, 2010), (Mehandjiev et al, 2010b),

(Mehandjievet et al, 2011), (Nestler et al, 2011), (Namoun et al, 2010d) and (Wajid et al, 2011), Namoun and other team members investigate service composition for non-programmers. After many users experimental studies on the ServFace Builder (ServFace [n.d.]) they draw attention to the fact that end-users do not realize that services can be connected together and do not easily understand that information can flow between services. Those findings as well the literatures presented in the tutorials helped us in our understanding and investigation of end-user composition of web applications and services.

3.4 Conclusion

Web Mashups combine information from multiple sources to produce a unified view of information to web-users. Web Mashups receive a lot of attention both from industry and researchers. Mashups promise to be the new way to "program" for the web. Different tools for creating Mashups exist and some especially target novice or casual users, i.e. users with little or no background in programming. Using these Mashup Makers it should be easy for casual users to quickly tailor and combine existing information for their own purpose. However, an important question is if indeed the available tools for Mashup creation are satisfying this promise, i.e. how high is their usability with respect to casual users. Furthermore, another following question is if existing usability evaluation methods could provide an answer to the first question.

Therefore, in this chapter, we have reviewed Mashups and some general-purpose Mashup Makers, their characteristics, and their composition approaches. We also reviewed the concept of usability as well as existing methods to evaluate the usability of products. We also presented our vision in regard to the effectiveness of the mentioned evaluation methods for Mashup Makers. Finally, we presented related work on the usability of Mashup Makers.

Part II: The empirical study and modelling the findings

CHAPTER 4: PILOT STUDY AND USER EXPERIMENT/STUDY

4.1 Introduction

The goal of this chapter is to report on the empirical studies (a pilot study and user experiment) performed in the context of our usability investigation. The purpose of performing these studies was to investigate the issues related to the usability of Mashup Makers for end-user. The results and feedback obtained from these studies have been used as input for the next research steps. Our ultimate goal is to come to a framework suitable to evaluate the usability of Mashup Makers. For this we need a good set of criteria for evaluating the usability of Mashup Makers. The studies described in this chapter has been set up in order to obtain the necessarily information to come to such a usability framework.

As set of usability evaluation criteria to start with, we have selected the Cognitive Dimensions (CDs) of notations framework (Blackwell et al., 2001), (Green et al., 1996).

The Cognitive Dimensions of Notations Framework provides 13 abstract criteria that can be used to evaluate the usability of visual languages. Those 13 dimensions are: Visibility, Abstract gradient, Closeness of mapping, Consistency, Diffuseness, Error-proneness, Hard mental operations, Hidden dependencies, Premature commitment, Progressive evaluation, Role-expressiveness, Secondary notation, and Viscosity. (A full description of the 13 dimension is provided in the context of research later on in sections 6.4.2, 6.4.3 and section 6.2). In addition to the aforementioned dimensions, new dimensions are sometimes proposed in the HCI research field, with different levels of adoption and refinement. These candidate dimensions are based on research outside the Cognitive Dimensions framework, and are adapted to it as a way to summarize that research into the hands-on approach encouraged by it (Blackwell et al., 2001), (Green et al., 1996).

For the empirical studies (both the pilot and user experiment) we performed and presented in this chapter, we have selected the Cognitive dimensions (CDs) framework because of its nature as task-specific, and concentrating on the process and activities. The framework also targets visual programming tasks, which makes it very suitable for Mashup Makers as these tools usually use a visual language. CDs framework provides a vocabulary that enumerates concepts important to variant users (skilled or not skilled) who are engaged in visual programming/design tasks. These concepts have been shown

over time to be important to human problem solving and it is important to consider each when designing a usable artefact or interface (Blackwell et al., 2001), (Green et al., 1996). However, note that usability evaluation against cognitive dimensions is subjective, and it is not a substitute for thorough user evaluation.

The chapter is organized as follows. In section 4.2, we describe the pilot study that we performed. In section 4.3, we explain the user experiment: the goal, the approach and methodology used; the design and performance; the results and an analysis of the results of the study. In section 4.4, we present a classification of the usability problems discussed. Section 4.5 concludes the chapter.

4.2 Pilot Study

The purpose of the pilot study that we present in this section is to follow up our usability investigation accomplished in the literature study and presented in chapter 3 by empirically investigating the usability of Mashup Makers by ourselves.

4.2.1 Pilot Study: Design and Performance

We have conducted a pilot study with 8 general purpose Mashup tools: Yahoo Pipes (YP) (Yahoo! pipes [n.d.]), Microsoft Popfly (MP) (Microsoft Popfly [n.d.]), Intel Mashmaker (IM) (Intel MashupMaker, [n.d.]), Openkapow Robomaker (OK) (OpenKapow [n.d.]), Jackbe (JB) (JackBe [n.d.]), IBM Mashup Center (IC) (IBM [n.d.]), Apatar (AP) (Apatar [n.d.]), and Dapper (DA) (Dapper [n.d.]). The pilot study was conducted by the author of this PhD thesis. For each Mashup Maker, five activities have been carried out (see Figure 4.1). The first activity was exploring the Mashup Maker from an end-user perspective taking in consideration the characteristics of these target users (casual users). The second, third and fourth activities are the three main steps of any Mashup creation process: aggregating data, manipulate data, and visualize data (Di Lorenzo et al, 2008). The fifth activity was the creation of a Mashup example. As mentioned earlier in the introduction of this chapter we adopted the CD's of notation framework as evaluation criteria, but we adapted its dimension to the context of our pilot study being Mashup

Makers. For each Mashup tool and for each activity performed (see table 4.1), we have given a qualitative evaluation for the different cognitive dimensions.

Table 4.1: Activities performed in the pilot study

Activity	1	2	3	4	5
	Discover Mashup maker	Collect 2 websites data	Preface Mashup creation	Perform Mashup creation	Run Mashup

4.2.2 Pilot Study: Results and Discussion

As we did the evaluation ourselves, it was not always possible to objectively give a mark to the different dimensions, as we are not casual users.

Table 4.2: CDs Evaluation for Mashup makers considered

Cognitive Dimension/ Mashup Maker	YP	MP	IM	OK	IC	JB	AP	DA
Abstraction Gradient	4	3	4	5	3	4	4	3
Closeness of mapping	4	4	3	3	3	5	5	4
Consistency	4	3	4	4	3	4	4	3
Diffuseness	4	4	3	4	5	4	4	4
Error-proneness	2	1	3	2	3	2	2	1
Hard mental operations	2	2	2	2	2	3	3	2
Hidden dependencies	3	4	3	4	2	4	4	4
Premature commitment	4	4	2	5	4	3	4	4
Progressive evaluation	1	2	3	3	2	3	3	2
Role-expressiveness	4	3	4	5	4	3	4	3
Secondary notation	2	3	3	3	2	2	3	3
Viscosity	4	4	3	5	3	2	2	4
Visibility	4	5	4	4	4	4	4	4

Therefore, the ranking provided should not be considered as some formal assessment. To obtain such a formal assessment it would be necessary to (re)do the evaluation with a representative number of members from the target users. Despite this limitation, the study itself was useful and interesting for different reasons. First of all, it provides us feedback on the use of the cognitive dimensions for the evaluation. We were able to detect which of the cognitive dimensions were useful to consider in further experiments and how to use those dimension. This also has given useful information for the definition of the usability criteria to be used. Table 4.2 presents the quantitative results we obtained. For every CDs dimension and Mashup Maker we have a rank from 1 to 5 that ranks how the Mashup Maker satisfies the CDs dimension (5 is the best and 1 is the worst). The numbers given in table 4.2 for the different CD's dimensions are the average of the scores obtained for each activity.

One of the findings from the pilot study is that it would be necessary to divide the target users further into groups based on their computer skills and background in English. This last issue turned out to be important because all considered Mashup Makers provide their interfaces in English. We also came to the conclusion that in order to allow for a better comparison between the different tools in further experiments, the experimentation environment should include a common example, and should also provide learning materials and some "know-how" tutorials.

More details of the pilot study are provided in Appendix 1.

4.3 User Experiment

4.3.1 User Experiment: Goal and Hypothesis

Our goal of the user experiment was to empirically evaluate the usability of some Mashup Makers by real end-users using the CD's framework evaluation dimensions. For this we have chosen to compare two groups, IT related and non-IT related people. This will allow us to use the paired t-test statistics method in order to show the difference between the two groups and to verify our hypothesis we have already set-up in the pilot

study we performed (Al Sarraj and De Troyer, 2009). Our research hypothesis of the user experiment is summarized as follows:

User experiment's research hypothesis:

"Mashup Makers should be more or less as usable for non-IT related people as for non-IT related users (casual users)".

Indeed, if Mashup Makers are constructed with casual users as their target audience, then the usability for this type of users should be rather good. It could even be possible that the tools are perceived as less usable (e.g., more frustrating) by IT-related people because these tools were actually not designed for them.

4.3.2 User Experiment: Approach and Methodology

To define our approach for the usability evaluation of Mashup makers, we adopted the three usability dimensions in the ISO 9241 definition (efficiency, effectiveness and satisfaction), but we also consider the learnability dimension from Abran et al. (Abran et al, 2003) and from Lew et al. (Lew et al, 2009).

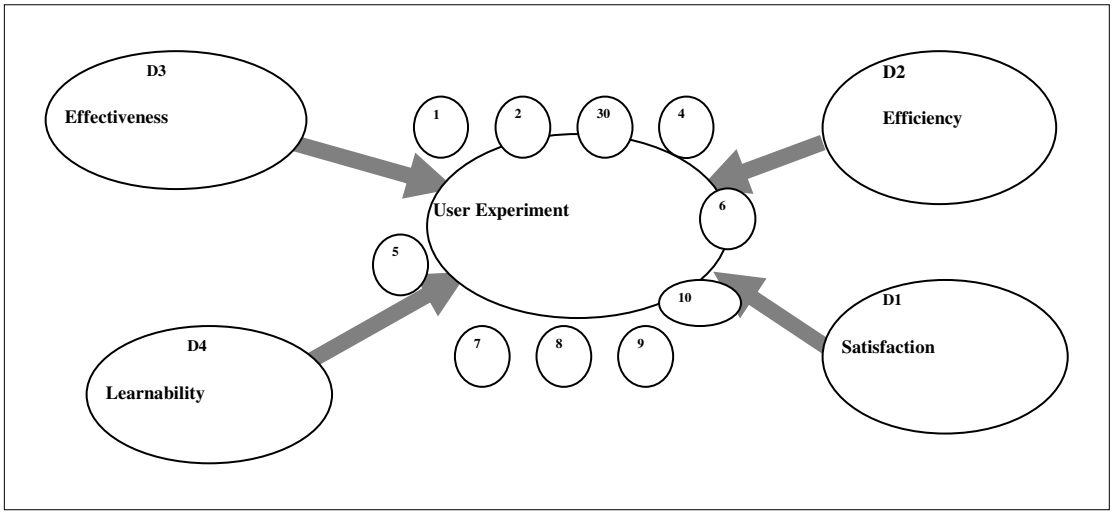

Figure 4.1: User Experiment approach

Actually, these dimensions are the theoretical and conceptual basis for our usability evaluation, but to make the different dimensions measurable we need criteria. Based on the results of the pilot study, we decided to used the following dimensions the CDs framework: visibility, gradient abstraction, hard mental operations, diffuseness, consistency, error-pronounce, role-expressiveness, progressive evaluation, viscosity, and provisionality. Below (see next section), we provide a brief description of these evaluation factors.

Figure 4.1 illustrates the approach taken in the user experiment approach for the evaluation of the usability of Mashup Makers. Four usability dimensions are considered; they are presented by means of the oval shapes at the four corners of the figure. The user experiment (the oval in the centre) is surrounded by several small numbered circles, representing the different evaluation factors (criteria) used (and adopted from the CD's of notation framework) for evaluating the usability of Mashup Makers.

In the pilot study that we performed first, it was possible to use 8 different Mashup Makers but unfortunately for the study described here, this was not possible any more: Microsoft Popfly was shutdown in August, 2009; Openkapow is not free anymore; Intel MashMaker is in standby mode as from the summer of 2009; Marmite is not available anymore as of the end of 2008; Serena is now more oriented to Business purposes; and IBM Mashup center became more complicated and needs lots of heavy software downloads and configuration. Although this is a pity, as we were not able to continue our research as planned, it doesn't mean that Mashup Markers are not important anymore. The fact that some tools disappear is a typical ICT phenomenon, as a new concept arise many different tools are developed but only a few survive time. So we have conducted our user experiment study with 3 Mashup Makers: Yahoo Pipes (YP), Open Mashups Studio (OMS), and Dapper (DA).

4.3.3 User Experiment: Design and Performance

Our user experiment was divided into two main phases, which will be described below.

Phase 1: in this phase we gathered the participant's backgrounds including:

- *Participant background and experiences*: We gathered information about the participant's age, background including his/her qualifications, the experience with the web and if any with Web Mashups. Also, we asked for the participant's experience with modelling and programming, if any.
- *Language level*: we asked for their knowledge of the English language and especially their knowledge of English words used on the web.
- *Participant's interests and motivations*: here we tried to collect information about the participant's interest in Web Mashups and web applications.

Phase 2: in this phase we performed the actual usability evaluation. As mentioned earlier, we adopted evaluation factors from the CD's framework and in particularly we used the adopted description and questions from (Blackwell and Green, 2000), (Green et al, 1996), which provide an improvement of the CD's framework. We give a brief definition of the evaluation factors adopted, their description in the context of our usability experiment, and a number of the questions asked during the experiments in the questionnaires or collected by means of the feedback sessions performed by the evaluator (more on this in section 4.3.4).

1. ***Visibility***: This factor refers to the ability to perceive components easily. In the case of the Mashup Makers, this means whether required Mashup components are visible without major cognitive work.

   Questions used: within the Mashup creation process, how easy is it to see or find the various parts of the application while it is being created or changed? What kinds of things are more difficult to see or find? If you need to compare or combine different parts, can you see them at the same time?

2. ***Hard mental operation***: This factor refers to the degree to which users need to resort to fingers or pencil annotations to keep track of the process, or experience difficulties during the process.

***Questions used*:** What kinds of things require the most mental effort? Do some things seem especially complex or difficult to work out in your head (e.g., when combining several things)? How easy is it to keep track of the Mashup design process?

3. ***Diffuseness*:** This factor refers to the verbosity of the Mashup Maker's interface and tools. In visual design environments as well as in Mashup Makers, some notations can be annoyingly long-winded, or occupy too much valuable "real-estate" within a display area, e.g., big icons and long words reduce the available working area.

   Questions used: What is the degree to which the Mashup Maker let you express what you want to express reasonably briefly (or is it rather long-winded)? What sorts of things take more space to describe? How high is the number of symbols or graphic entities required to express something?

4. ***Abstraction Gradient**:* This factor refers to the ability of grouping elements in order to be able to treat them as one element.

   ***Questions used**:* Does the Mashup Maker give way for defining new facilities or terms within the Mashup application so that you can extend it to describe new things or express your ideas more clearly or succinctly? What is the degree in which the Mashup Maker insists that a user start by defining new terms before he/she can do anything else?

5. ***Consistency*:** Consistency in the context of usability of user interface systems can be understood in the following sense: "Similar semantics are expressed by similar syntactic forms. Users often infer the structure of information artifacts from patterns in application". Applying this to Mashup Makers allow measuring the degree of consistency of Mashup Maker by the degree to which similar components are semantically expressed by similar syntactic or visual forms.

   ***Questions used**:* Where there different components of the Mashup Maker that meant similar things? Is the similarity clear from the way those components appear?

6. ***Error-pronounce***: This factor refers to the degree that within the Mashup design process, the Mashup Maker allows mistakes or prevents the user from making mistakes.

   Questions used*: Does the* Mashup Maker use an easy Mashup design process which makes it difficult to make mistakes? Did you often find yourself making big slips that were irritating or make yourself feel stupid? How easy is it to make mistakes?

7. ***Role-expressiveness***: This factor refers to the degree to which the Mashup design environment has semantically clear components and functions, i.e. how easy is it for a user to know the role of a components or part of a process.

   Questions used*:* When considering a function or an icon within the Mashup design process, is it easy to know its role in the overall Mashup design? Are most parts of the Mashup design process easy to understand? Do most parts mean what you expected?

8. ***Progressive Evaluation***: This factor refers to the degree to which work can be checked at any time. Evaluation is an important part of a design process, and Mashup Makers can or cannot facilitate progressive evaluation by allowing users to stop in the middle of their design to check work so far, find out how much progress has been made, or check what stage in the work they are up to.

   Questions used*:* Was it easy it to stop in the middle of creating a Mashup application and check your work so far? Could you do this any time you liked? Could you find out how much progress you had made, or check in what stage in the work you were? If not, why not? Could you try out partially completed versions of the Mashup application?

9. ***Viscosity*****:** This factor refers to the degree of resistance to local change.

   Questions used: When you needed to make changes to previous work, was it easy to make the change? Were there no or rare particular changes that were more difficult or especially difficult to make?

10. ***Provisionality and premature commitment:*** This factor refers to the degree of commitment to actions, or if there are hard constraints on the order of doing things. Does the Mashup Maker allow the user to fool around or make sketchy things?

    ***Questions used*:** Is it possible to sketch things out when you are playing around with ideas, or when you aren't sure which way to procede? Does the Mashup Maker provide features or notations to help you doing this?

For the experiment we used two separate groups, involving 24 participants in total and 12 participant for each group (all were graduate or postgraduate students at our university). The first group is the group of IT-related people who hold a university degree in some ICT related field (Computer Science, Computer Engineering, Information Technology, Electrical/Control/Communication Engineering, etc.). People in this group have studied at least some programming languages such as Basic, Fortran, Pascal, C, C++ or Java, and/or a computer modelling language or environment such as UML or Matlab. The second group consists of non-IT people, people without any programming backgrounds, and holding a university degree in a specialization other than ICT such as Arts, Literatures, Biology, Medical Science, Law, Economy, Accounting, etc. Table 4.5 shows a resume of the participant's background.

We have split the user experiment into 2 main phases as shown in table 4.3. We explain them here:

Table 4.3 User experiment phases

Phase	Description
A	- Welcome of the participant and fill-in of bio-data questionnaire. - Presentation with videos showing an introduction to the Mashup Creation Tools.
B	- User activity including user tasks performance and evaluation - Closing of the user experiment with an oral feedback.

Phase A: First the participant has been welcomed and asked to fill in a bio-data questionnaire including questions on gender, age, mother tongue, English level, university degree and specialization, Internet use and other background like: how many year he/she already used the Internet; frequency of using the Internet; purpose; browsers used, chat tools used, social networking websites used; knowledge of Mashups and in which domain (if any). The questionnaire was followed by a brief presentation introducing Web Mashup Makers and 3 short videos about respectively Yahoo Pipes (YP), Open Mashups Studio (OMS), and Dapper (DA). Then, there was a five minutes session to allow the participant to ask questions. As output of phase A, we have the bio-data of each participant (resumed in table 4.5) and a sheet of notes and questions.

Phase B: Because Mashup Makers differ in their purposes and functionalities, it was not possible to have a common task for the three Mashup Makers. Therefore, we asked the participant to perform a different task for each Mashup Maker. This also avoided a learning effect between the three successive tasks with the three Mashup Makers. Every user was asked to perform the three tasks randomly.

> ***Task 1***: Yahoo Pipes (figure 4.2): the participant was asked to create a pipe to fetch feeds on the Flickr website with Brussels in the title, then to filter the results to the Brussels area having the zip code '1050', and display the results as spots on a Yahoo map.
>
> ***Task 2***: Open Mashups Studio (figure 4.3): the participant was asked to create a Mashup to search for feeds of a picture of Brussels and sending the first feed as MMS to a given mobile number and saving the number and picture for future use.
>
> ***Task 3***: Dapper (figure 4.4): the participant was asked to create a dapp (Dapper Mashup) of a list of RSS feeds news from two Belgian or European newspapers in English.

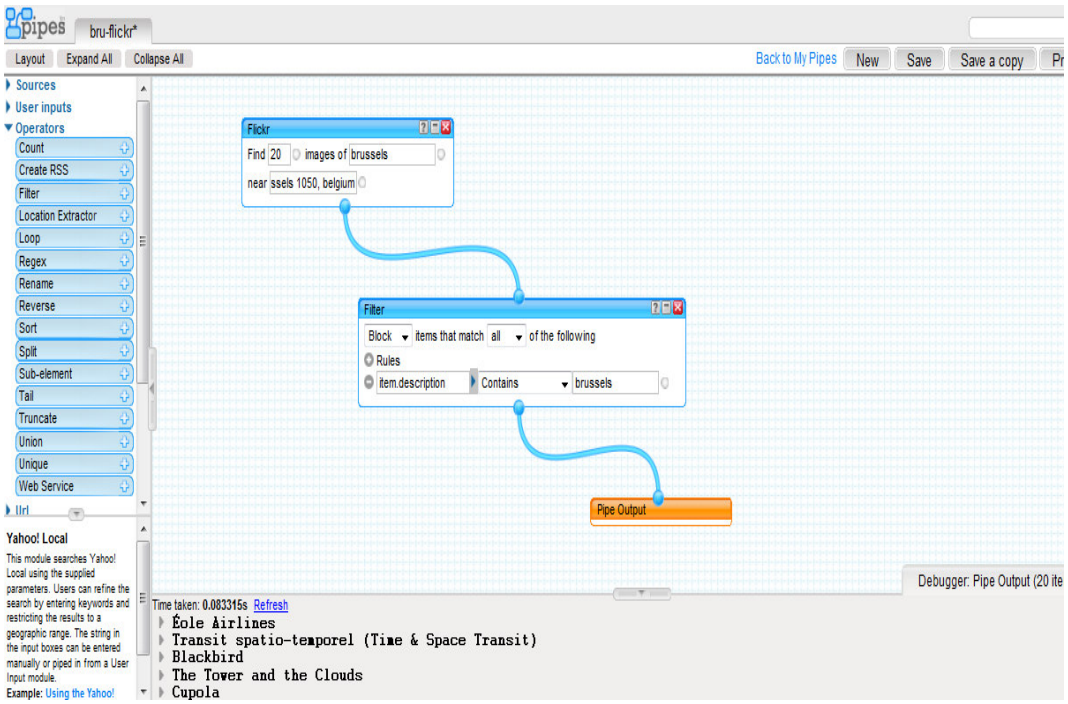

Figure 4.2: A snapshot of Yahoo Pipes task requested

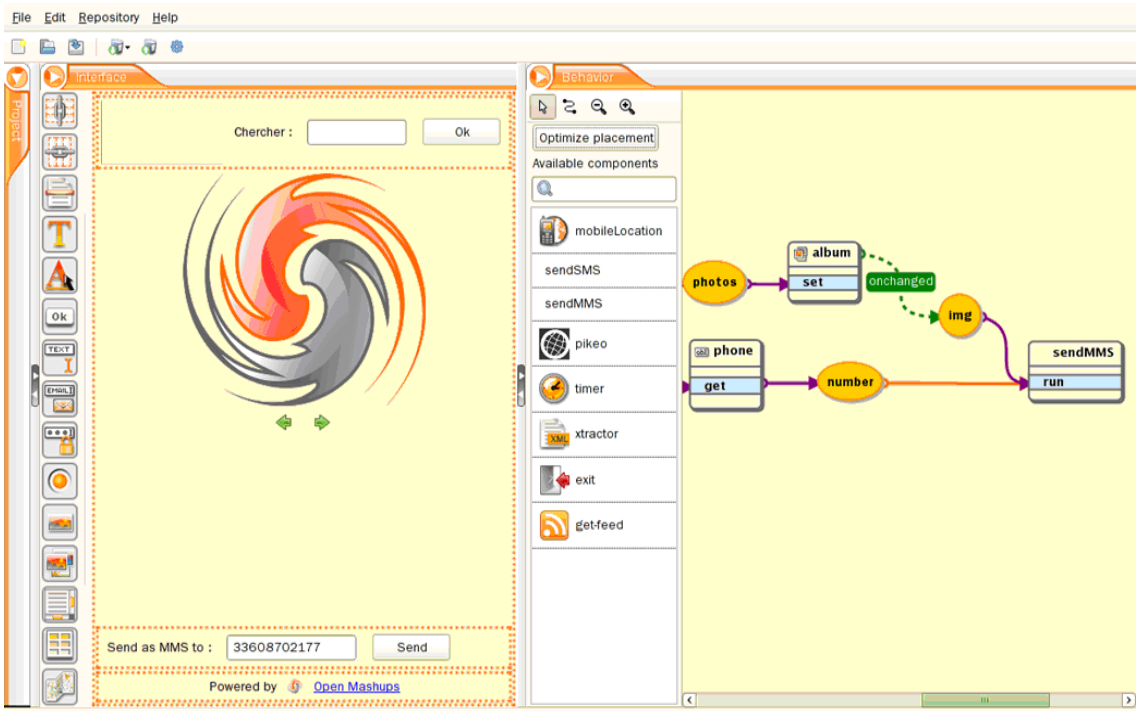

Figure 4.3: A snapshot of Open Mashups Studio task requested

While the participant performed the tasks we asked him/her to think aloud, we made notes of his/her actions, and filled in our prepared evaluation sheets with time, task performance, errors, frustration rate, and other evaluation factors. At the end of a task the participant was asked to fill in a mini questionnaire (figure 4.5), which took about three minutes, to express his opinion about the Mashup Maker in relation to the task performed. As output of the phase B we have for each participant three sheets of evaluation notes and three mini questionnaires, one for each Mashup Maker considered.

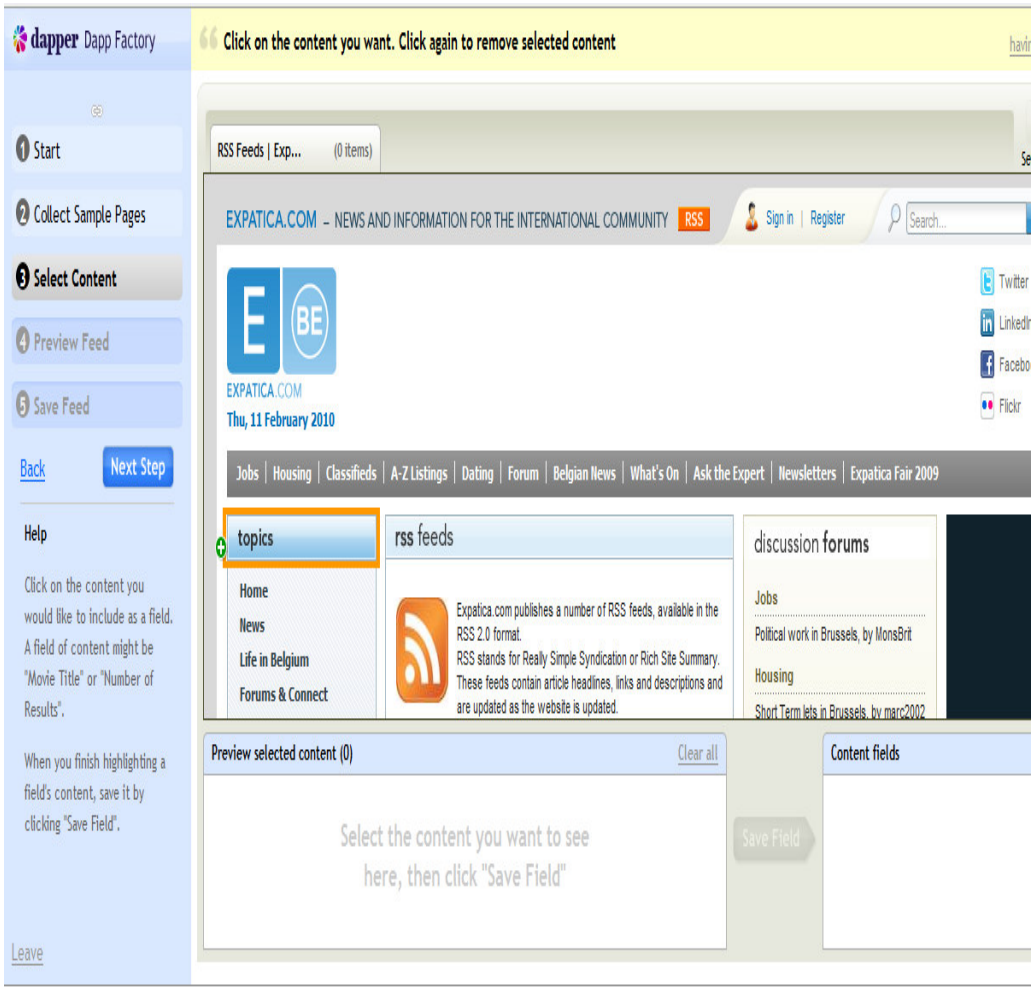

Figure 4.4: A snapshot of a requested dapper task

And finally at the end of the phase B the participant was invited for a coffee and asked to give orally his/her feedbacks about the experiment and his/her opinion about the Mashup

Makers they used. This has been done to be able to check if there were some contradictions between the written and oral feedbacks.

The purpose of the experiment was to assign a qualitative evaluation to each evaluation factor and for each activity. For the qualitative evaluation, we have rated our observations and also the feedback by means of five levels. The score 5 is used for 'Very high level', 4 for 'High level', 3 for 'Moderate level', 2 for 'Low level', and 1 if we observe that the factor was very low in the tool during the considered activity. The ranking is summarized in Table 4.4.

	Question	**disagree**				**agree**
		1	2	3	4	5
1	Mashup Maker allows to find components easily.	❑	❑	❑	❑	❑
2	Mashup Maker provides a straightforward design process.	❑	❑	❑	❑	❑
3	The number of graphical entities provided by the Mashup Maker fits my expectations/needs.	❑	❑	❑	❑	❑
4	Names of components provided by the Mashup Maker clearly indicate their functionality.	❑	❑	❑	❑	❑
5	Components provided by the Mashup Maker are relevant.	❑	❑	❑	❑	❑
6	The workflow for creating the Mashup is self-explaining and clear.	❑	❑	❑	❑	❑
8	Mashup Maker provides methods to review completed or semi completed tasks.	❑	❑	❑	❑	❑
9	Mashup Maker provides the ability to change components during the Mashup design.	❑	❑	❑	❑	❑
10	Mashup Maker provides alternatives components in case of misunderstanding	❑	❑	❑	❑	❑
11	The most negative aspect of this Mashup Maker is ……					

12	The most positive aspect of this Mashup Maker is ……...
13	Overall suggestion to increase the usability: ……..….

Figure 4.5: Mini Questionnaire after every task of phase B

The final rate for a Mashup Maker/Evaluation factor for a subject is calculated as the average for the whole gathered ranked activities and from all related questionnaires and evaluator's feedback sheets. We also calculate the standard deviations. The results are explained in the next section.

Table 4.4 Ranking activity by factor level

Factor level	**Very high**	**High**	**Moderate**	**Low**	**Very low**
Rank	5	4	3	2	1

4.3.4 User Experiment: Results

The results of the study explained in the previous sections are summarized in three ways: table 4.5 summarizes the background of the participants. IT-related participant's specialization varies between Computer Science, Information Technology, and (Computer) Engineering, but all have at least studied one programming language and/or modelling course in their bachelor degree. The non-IT participant's background ranges from Literature, over Arts, to more Natural Science-oriented educations. The average age is respectively 30 and 28 years; and there are 6 males and 6 females. Table 4.5 also summarizes the average daily use of Internet, as well as the English level. IT-related people reported more experience with the Internet and better English skills (especially for English Internet terms) than the non IT-related people. (Note that the participants were not native English speakers.)

The results of the user experiment are summarized in a descriptive statistical way in table 4.6. The results for the three Mashup Makers are grouped in three columns: Yahoo! Pipes (label 'YP'), Open Mashup Studio (label 'OMS'), and Dapper (label 'DA'). For each Mashup Maker the results are given for the IT-related (columns marked with 'IT') and for the non-IT related (columns marked with 'Non-IT') participants. For each usability factor the average (AV) and standard deviation (SD) of the individual rates obtained for each usability factor are given for the two different groups and the different Mashup Makers. We can observe a noticeable difference for all the usability evaluation factors between the two groups.

Table 4.7 summarizes the results of the paired sample t-tests on the overall usability evaluation obtained using SPSS. In table 4.7, we see the differences between the two groups. There is a significant difference between the IT and Non-IT group for Yahoo Pipes (t is 15.224 and the Sig is ,000), there is also a difference for Open Mashups Studio (t is 7.327 and sig is ,000) and for Dapper (t is 6.749 and sig is ,000) but less than for Yahoo Pipes. Full details of the SPSS results statistics are provided in appendix 2.

Table 4.5: Resume of Participants backgrounds.

Partici -pant	**#**	**Age Average**	**Female/ Male**	**Average Years of Internet experience**	**Average Daily Internet Use (in Hours)**	**Average English level**	**Qualification = University degree**
IT-related	12	30	6F/ 6M	8.5 SD=2.37	4.66 SD=1.82	4.16 SD=0.83	3 Computer Science, 3 Information Technology, 2 Computer Engineering, 3 Communication & control Engineering, 1 Electrical Engineering
Non-IT related	12	28	6F/ 6M	6.08 SD=2.46	2.91 SD=1.67	3.4 SD=1.02	2 Literature, 2 Arts, 1 Pharmacy, 1 Law, 1 Biology, 2 Accounting, 1 Geology, 1 Geography, 1 Chemistry

Table 4.6: User Experiment Results: Mashup Makers/Evaluation Factors

No	Evaluation Factor	YP				OMS				DA			
		IT		Non-IT		IT		Non-IT		IT		Non-IT	
		AV	SD	AV	SD	AV	SD	AV	SD	AV	SD	AV	SD
1	Visibility	4.46	0.91	2.08	0.91	3.64	1.43	1.44	0.83	4.51	0.94	3.51	1.14
2	Hard mental operation	4.09	0.82	2.01	1.24	3.76	0.78	1.05	0.98	4.37	0.83	2.07	0.93
3	Diffuseness	4.01	0.97	2.19	1.34	3.07	0.92	1.41	0.76	4.19	0.68	2.78	1.21
4	Abstraction gradient	4.18	1.19	1.87	1.11	3.98	0.75	1.71	1.25	4.56	0.68	2.28	1.18
5	Consistency	4.14	1.51	1.49	0.91	3.81	1.33	1.33	0.63	4.62	0.75	3.61	1.05
6	Error-pronounce	3.96	0.87	1.16	1.37	3.73	1.27	3.73	1.27	4.36	1.12	2.18	1.11
7	Role-expressiveness	3.89	0.71	1.99	1.41	3.45	0.98	1.44	1.28	4.15	1.53	2.54	0.93
8	Progressive Evaluation	4.32	0.92	3.05	0.92	4.11	1.42	2.86	1.02	4.52	0.69	3.49	0.99
9	Viscosity	4.07	0.98	2.09	1.08	3.84	0.81	1.79	0.91	4.65	0.78	2.65	0.88
10	Provisionally	4.35	1.42	2.35	0.82	4.06	1.17	1.16	0.77	4.69	1.25	2.69	1.35

Table 4.7: Paired T test results in SPSS.

Pair user group	Mean	t	df	Sig (2-tailed)
YP: IT - Non-IT	2,119	15,224	9	,000
OMS: IT - Non-IT	1,953	7,372	9	,000
DA: IT - Non-IT	1,578	6,749	9	,000

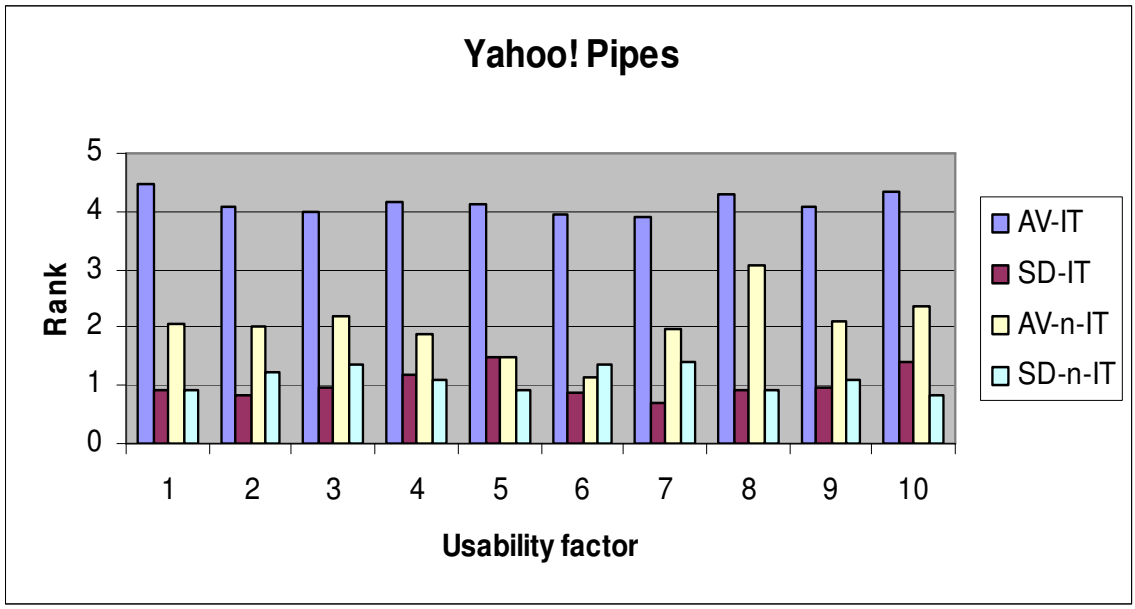

Figure 4.6: User Experiment results: Yahoo Pipes

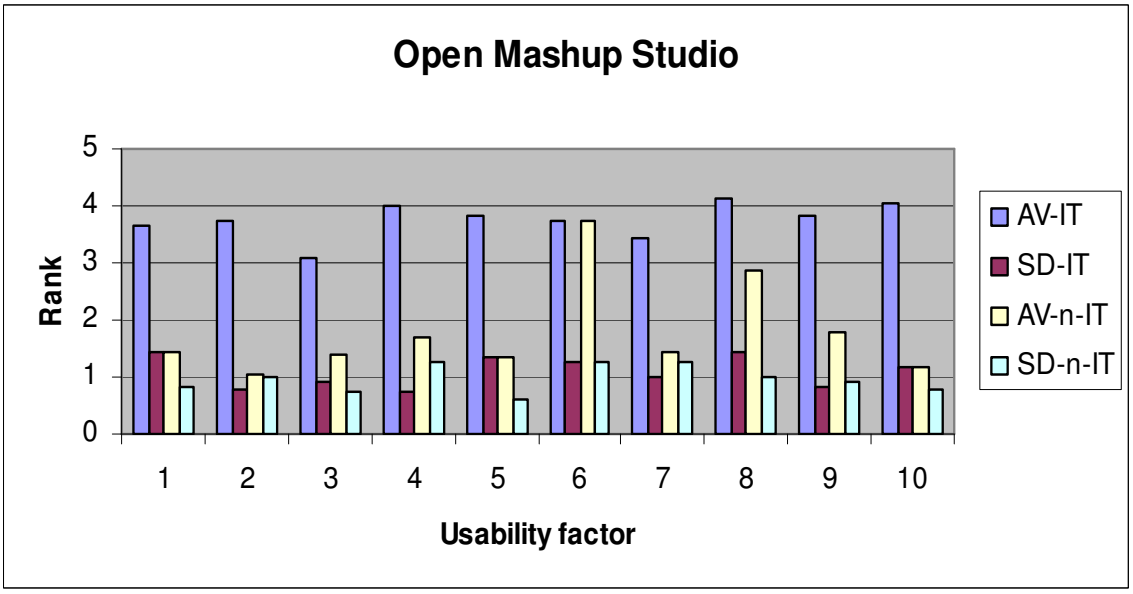

Figure 4.7: User Experiment results: Open Mashup Studio

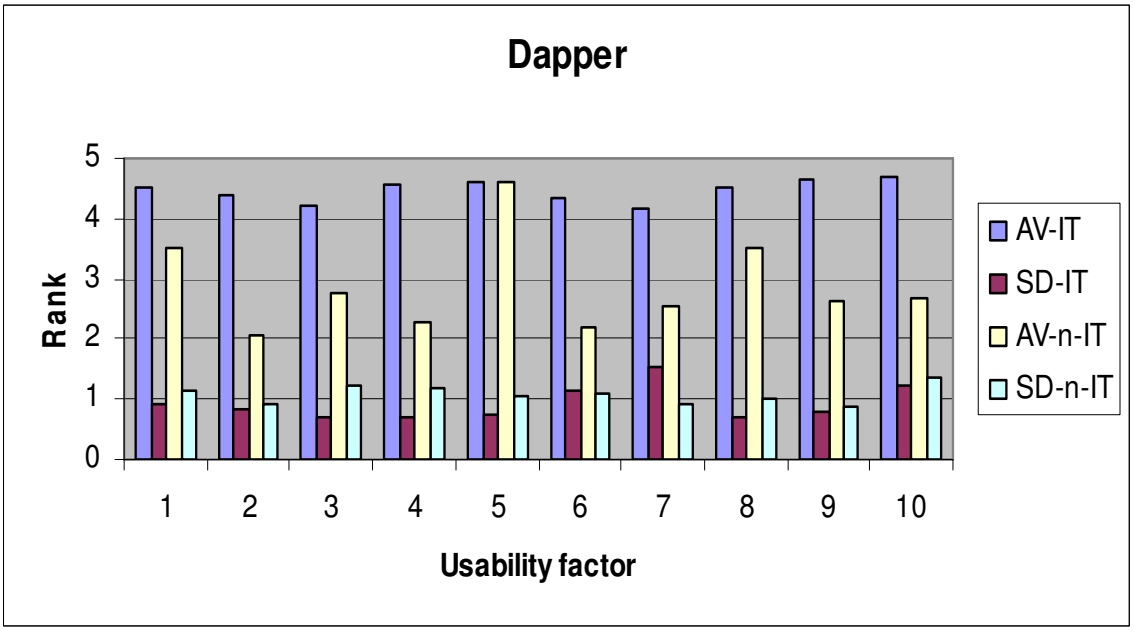

Figure 4.8: User Experiment results: Dapper

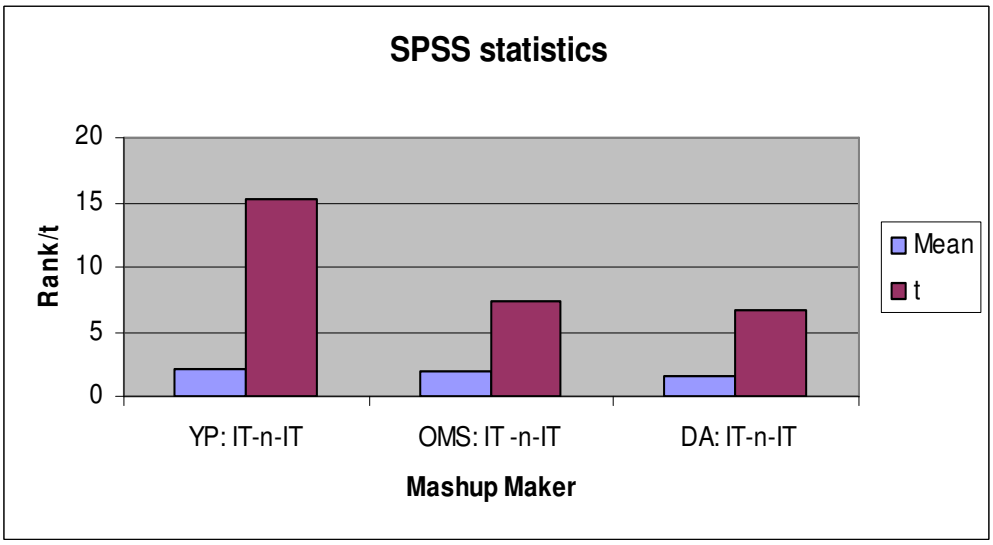

Figure 4.9: Paired T test Results in SPSS

4.3.5 User Experiment: Discussion

Herewith we analyse and discuss the results presented in table 4.6 and in table 4.7 for the different evaluation factors and from a statistical perspective:

1. *Visibility*: IT-related participants gave remarkable higher scores compared to non-IT related participants. This may be due to the fact that IT-related people have more experience with similar tools.

2. *Hard mental operation*: in contrast to the IT-related people who have shown less difficulties with this factor, non-IT related people have shown high difficulties and they needed papers and pens to resort and to keep track of the process.

3. *Diffuseness*: the comparison between the two groups in this factor is really noticeable; almost all the IT-related group accomplished the requested tasks in a time close to the planned time. In contrast to this, non-IT related group showed a large delay, in some cases doubled the planed time.

4. *Abstraction gradient*: for this factor, IT related participants were able to understand the questions asked and gave their answers/feedbacks in a self-evident way. Instead, the non-IT related participants needed to ask for explanation. For this factor, the results in table 4.6 show the significant difference between the two groups for all the Mashup Makers considered in this user experiment. However, is not clear whether this is due to the fact that non-IT people had problems in understanding the concept or if they really perceived the tools as being low in abstraction gradient.

5. *Consistency*: IT-related participants gave a rationale explanation both in their thinking aloud and in the feedbacks they gave. In contrast to this, non-IT users were in general confused when we were asking the questions related to this factor. Therefore, we tried to get their feedbacks by giving simple examples. Nevertheless, they gave a low rate for this factor. So, also here we can make the same remark as for the previous factor.

6. *Error-pronounce*: The non-IT users made more mistakes and repeated their mistakes more than the IT-related users.

7. *Role-expressiveness*: Also here we noticed a significant difference between the results obtained in the two groups. IT-related participants rated this well while non-IT related participants showed an uncertainty.

8. *Progressive Evaluation*: for this factor, there is a significant difference between the two groups. IT related users were much more able to distinguish their progressive steps when designing the Mashups.

9. *Viscosity*: IT-related users have shown fewer efforts when performing the tasks than the non-IT users. In our opinion, this factor is highly important for casual users.

10. *Provisionality and premature commitment*: results obtained show an important difference between IT-related and non-IT related users who have given a very low rate on this factor.

The conclusion is that there is quite an important difference between the rates of the IT-related users and the non-IT related users. This could lead to the conclusion that compared to IT-related people, the Mashup Makers are less usable for casual users (non-IT related people). However, the difference may be due to the fact that IT-related people are already familiar with many of the concepts used in Mashup Makers and these concepts may be completely new for non-IT related users. It is possible that the rates become better after the non-IT users have been able to practice more with the tools. This has to do with learnability and should be investigated further. Maybe these tools become as usable for non-IT people as for IT-related people after some learning period. However, also for criteria which are less influenced by the familiarity with the concepts, such as consistency, abstraction gradient, and progressive evaluation there is a big difference between the two groups, except for Dapper where the differences are small for consistency and progressive evaluation. Maybe this is due to the fact that Dapper is using successive steps in the Mashup design environment. The differences in this kind of evaluation criteria may be a first indication that Yahoo Pipes and Open Mashups Studio, and to a lesser extent Dapper are not very usable for casual users. However, further research is needed to confirm this and to pinpoint the exact reasons for this. However,

note that it is not the purpose of our research work to further investigate the usability of particular Mashup Makers; our research goal is to come up with a usability framework for Mashup Makers for end-users. Therefore, we will not perform more experiments in the context of the usability of Yahoo Pipes, Open Mashups Studio, or Dapper, but rather use the results of this experiment to establish our goal.

In conclusion we can state that the results of this experiment do not allow us to accept the original hypothesis. However, it is also not possible to reject the hypothesis without further investigation.

4.4 Classification of the Usability Problems

Examination of the reported usability problems in association with task analysis results can provide a useful reference point for identifying and organizing usability factors to be considered in a usability evaluation (Ham et al, 2006).

We collected about 90 specific usability problems from non-IT users during the user experiment. Those problems are distributed over the following six main categories:

(1) User interface perception: this includes many visibility problems, as well as many cases that led to end-user frustration.
(2) Design metaphors perception: problems related to understanding the meaning of components and confusion in combining components (not) compatible with each other.
(3) Logical design hindrances: problems related to lack of technical modelling skills especially in using steps cascading and the composing of many components.
(4) Modelling problems: problems related to the complexity of Mashup functionality and to misunderstanding the steps of the creation process.
(5) Lack of programming insight: misunderstanding in identifying parameters for operators, as well as problems with understanding loops, if-else-then, strings, comparison, and so on.
(6) Difficulties in scripting and code handling: in some cases, the user needs to understand html code as well as syntax of YQL (Yahoo! Query).

These six categories can be clustered into three main areas. Firstly, those related to the tool's *user interface perception*, secondly those related to the *user's interaction* with the tool, and finally those related to *the goal of creating a Mashup* within a Mashup Maker (functional requirements of creating a Mashup). In the next subsections, we will give a detailed description of the coding process we performed for the usability problems.

4.4.1 Coding of the Usability Problems

We performed a long process for coding the usability problems during and after the user experiment. Each subject's interaction was reviewed twice. This part of the analysis consisted of annotating the question/answer sheets of the subjects, as well as our own notes taken during the thinking aloud sessions, with codes indicating problems encountered in using and interacting with the Mashup Maker. The coding scheme was based on the one described by Kushniruk et al (1996). As explained earlier in this section, the problems were distributed over three main areas. Firstly, there were the problems related to the user interface of the Mashup Maker - this includes categories of identifying problems with: data entry, display visibility, navigation, locating information, following procedures, typing, speed, and attention. Secondly, there are the usability problems related to the user interaction with the Mashup Maker - this was reviewed and coded in the approach of where discrepancies occurred between what the user was asked to do or should do and the actual action done. Here we developed a coding scheme to identify users' interaction usability problems. Thirdly, and the last area is that related to Mashup functional problems - this includes categories of problems with the content of Mashup tool, also problems occurring due to an incorrect default selection of components or Mashup items. In the next subsections we will give examples of usability problems in the three areas mentioned.

4.4.2 Usability Problems with respect to the User Interface Perception

In this subsection, we present the usability problems obtained during the user experiment, specifically related to difficulties with user interface perception and the disability to

perceive user interface components. The examples are listed in Table 4.8 (the full list of usability problems in this area is provided in appendix 3.1).

Table 4.8: Examples of user interface problems.

Problem	Description
Unclear design areas	End-users were unaware or misunderstood the design areas of the Mashup Maker, including colours, spaces, shapes, etc.
Unclear design components	End-users were unaware or misunderstood the design components of the Mashup Maker. End-users are not able to match a Mashup Maker components with its real world equivalent.
Confusing design metaphors	End-users were confused by the metaphors used and sometimes complained that those metaphors mislead him/her. End-users wondered whether different words, situations, or actions mean the same thing.

4.4.3 Usability Problems with respect to User Interaction

Table 4.9 shows examples of the main usability problems related to user interaction (the full list of usability problems in this area is provided in appendix 3.2).

Table 4.9: Examples of user interaction problems.

Problems	Description
Slowing and/or stopping during design process of a Mashup.	End-users face difficulties in understanding the working of the Mashup Maker while interacting with it. He/she expects other actions, and he/she wrongly reacts to the unexpected action. This causes slowing down the design process and in many cases the users had to stop and ask for help.
Inability to memorize components and steps during the creation of a Mashup	End-users must rely on their memory to recall components rather than to recognize components.
Uncertainty and fears of making errors during	End-users are in doubt on how to interact with the Mashup Maker. Users often make mistakes and are

steps cascading or components composing.	looking for a clearly marked "emergency exit" as well as an undo and redo functionality.

4.4.4 Usability Problems with respect to Functionality

Table 4.10 shows examples of the main usability problems related to the functional area (the full list of usability problems in this area is provided in appendix 3.3).

Table 4.10: Examples of functional problems.

Problems	Description
Unawareness and/or misunderstanding of component's meaning and functionality	End-users were unaware or misunderstood the meaning, goal and functionality of different Mashup creation components as well as operators and parameters.
Unawareness of consequences of components composition actions.	End-users had difficulties in composing the single components required for creation of a Mashup and didn't know how to order them.
Unawareness of consequences of cascading steps.	End-users had difficulties in defining the individual steps required for creation of a Mashup and how to specify the order in which these steps should be cascading or executed.
Ambiguity and/or uncertainty of using and building structures that represent models and reuse of designs.	End-users showed very low understanding with respect to using and reusing of designs that build models. End-users also complained about incompatibility of components and designs.

4.5 Conclusion

In this chapter, we have presented the pilot study and the user experiment that we performed to evaluate the usability of three mainstream Mashup Makers. The results of the user experiment could not confirm a good usability for casual users. Rather, the

results suggest that these tools are actually targeting more IT-oriented people. We observed that casual web users faced many difficulties when using the tools.

Based on the finding of the user experiment we have classified the usability problems detected into three categories. Those categories will be the basis for our conceptual evaluation model that we will present in the next chapter.

In the next chapter we will further investigate usability impact factors for Mashup Makers and we build further on the usability problems identified in this chapter. We need to investigate more in detail which aspects are causing the difficulties that end-users experience when using the tools. The result will be captured by means of a conceptual model for evaluating the usability of Mashup Makers.

CHAPTER 5: THE CONCEPTUAL EVALUATION MODEL

At the end of the previous chapter, we presented three different categories of usability problems collected. These three categories form the basis of our Conceptual Evaluation Model that we present in this Chapter. The main purpose of the Conceptual Evaluation Model is to provide a conceptual framework (i.e. model) for identifying the usability indicators for Mashup Makers for end-users. This chapter prepares for the next chapter where we present the actual Usability Evaluation Framework.

To indicate the difference between the Conceptual Evaluation Model described in this chapter and the Usability Evaluation Framework described in the next chapter, we refer to the work of Vicente (1999), who states that a conceptual model should prescribe features or requirements that need to be represented in a model. In contrast, a methodological framework should prescribe how to develop a model in a proceduralized way. Furthermore, the term conceptual model may be used to refer to models which are represented by concepts or related concepts which are formed after a conceptualization process in the mind. Conceptual models represent human intentions or semantics. Conceptualization from observation of physical existence and conceptual modelling are the necessary means human employ to think and solve problems (Duan and Cruz, 2011).

The components in the Conceptual Evaluation Model are based on the usability evident investigations and factors for Mashup Makers which are represented in the usability problems resulted from the user experiment and concluded in section 4.6. Further on, we justify and motivate the different components of the model by means of findings from the literature. However, before doing so, we first discus the concept of usability factor.

5.1 Usability Factors

To evaluate usability in a more systematic way, many studies examined factors or dimensions constituting usability (Bevan, 1999). For example, and as already indicated, (ISO/IEC 9241, 1998) defines three dimensions: effectiveness, efficiency, and satisfaction. Another example is the set described in (Nielsen, 1993): learnability, efficiency of use, memorability, errors, and satisfaction. These dimensions can be classified into two main groups: objective and subjective dimensions. An objective

dimension generally measures how well the users' tasks are supported by applying task performance measures like task completion time and the number of errors. Objective dimensions do not always predict the user's assessment of usability because it does not reflect users' feeling or satisfaction. Subjective dimensions therefore also need to be assessed to provide a holistic and complete usability measurement (Ham et al, 2006).

For any usability evaluation method it is important to identify different kinds of usability factors in a systematic way. A usability evaluation factor can be defined as: "an entity, resource or a unit of information which refer to or provide meaning of an evaluation of the usability of certain object (Karwowski et al, 2011)". Several researches and studies have identified various kinds of usability factors to characterize the usability of different software system artefacts (e.g., (Frøkjaeer et al, 2000), (Klockar et al, 2003), (Folmer and Bosch, 2004), (Folmer et al, 2003), (Hornbaek, 2006)). In addition, usability practitioners need a structured model that can help them understanding the relationships among different usability factors (Ham et al, 2006). Additionally, usability evaluation methods should help usability practitioners to identify critical usability problems systematically and generate better design ideas (Blandford et al, 2004) (Lee et al, 2006). In this regard, one critical role of an evaluation method is to lead usability practitioners to consider various usability factors from multiple points of view (Hartson et al, 2003).

Usability factors may also differ in their degree of abstraction. For instance, the major factors composing the conceptual model that we will present are of a very high level of abstraction (i.e. user interface, user interaction, and functional support of the Mashup Maker). Therefore, we will consider them as *indicators* of usability. Such high-level factors could be composed of sub-factors of middle level of abstraction. Examples of sub-factors could be visibility, consistency, affordance, and feedback. In general, these sub-factors are still too abstract to be measured directly. Therefore, they are in general measured using e.g., questionnaires or analysing raw data collected during the usability evaluation process.

5.2 The conceptual Evaluation Model

As already indicated, the purpose of the Conceptual Evaluation Model is to serve as a meaningful and useful framework or model for identifying the specific usability impact factors of Mashup Makers.

Based on the discussion in previous sections and on the usability problems areas that we identified and presented in section 4.4, we developed this Conceptual Evaluation Model. A high level representation is presented in figure 5.1.

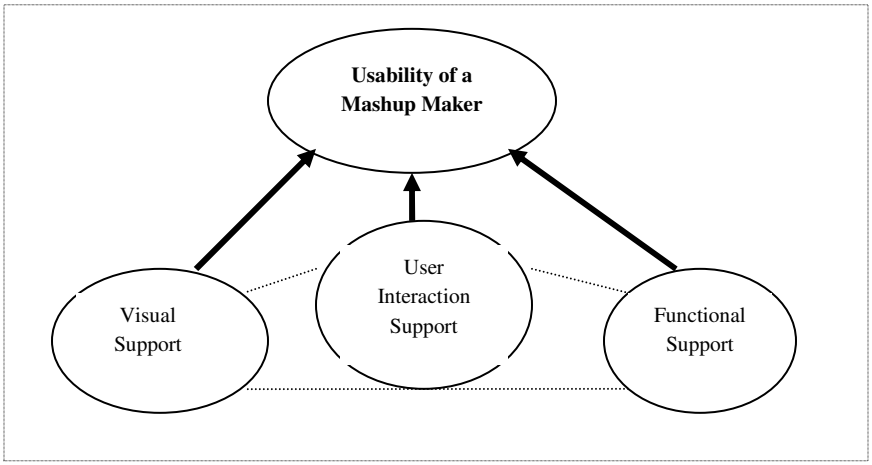

Figure 5.1: Conceptual Evaluation Model

Our Conceptual Evaluation Model consists of three main parts, which represent the three main usability aspects of Mashup Makers. These three parts are: *Visual Support*, *User Interaction Support*, and *Functional Support*.

The *User Interaction Support* part addresses the usability of the Mashup Maker from a user interaction perspective. It groups usability aspects such as cognitive and intuitive interaction support. The *Visual Support* part is concerned with the usability of the user interface of the Mashup Maker: layout of components, size, colour, metaphors, etc. The *Functional Support* part considers how the users' functional requirements are supported by the Mashup Maker. It should be noted that these three parts are not completely

disjoint. There are concepts that can be considered in two or more parts. Actually, the three parts should be considered as different viewpoint on usability. As such, the same concept can be considered in different parts depending on the point of view used.

In the next subsections, we clarify and motivate the relevance of each part of the model by means of findings from the literature.

5.2.1 Visual Support

Visual notations play an important role in communicating with end-users and customers, as they are believed to convey information more effectively to non-technical people than text (Avison and Fitzgerald, 2003). The importance of using visual notations and representations come from the fact that visual representation are effective because they tap into the capability of the powerful and highly parallel human visual system. We like receiving information in visual form and can process it very efficiently: around a quarter of our brain is devoted to vision, more than all other senses combined (Kosslyn, 1985). It is worthily important here that diagrams can convey information more concisely (De Marco, 1978). In software engineering, it is common that visual languages are more precise than ordinary textual (natural) languages (Blandford et al, 2004) (Larkin and Simon, 1987). Information represented visually is also more likely to be remembered due to the picture superiority effect (Goolkasian, 2000).

Visual language is one of the oldest forms of knowledge representation (Tufte, 2001). There are two main approaches that make the difference between visual languages and textual languages on how they encode information and how they are processed by the human mind (Moody, 2009). The first approach is concerned with the fact that visual languages encode information using special arrangement of graphic (and textual) elements. They are two-dimensional (spatial) representations (Larkin and Simon, 1987). The second approach is the fact that visual representations are processed differently: according to the dual channel theory (Mayer and Moreno, 2003) visual representations are processed in parallel by the visual system, while textual representation are processed serially by the auditory system (Bertin, 1993).

These differences mean that fundamentally different principles are required for evaluating and designing visual languages than for evaluating textual languages. However, such principles are far less developed than those available for textual languages (Moody, 2009), (Gurr, 1999), (Winn, 1990).

Pinker (1984) describes *visual cognition* as the process that allows us to determine on the basis of retinal input what particular shapes, configuration of shapes, objects, scenes and their properties are before us. He also added that, visual cognition is to focus on the recognition of shapes and representations of objects and spatial relations in perception and imagery; it is also known as no less than language or logic, maybe a talent that is central to our understanding of human intelligence.

In (Freitas et al, 2002), Freitas et al. suggest that usability evaluations of visualizations involve three issues: the presentation of the data, the interaction with the data, and the usability of the data itself.

5.2.2 Interaction Support

The interaction support usability aspects of Mashup Makers can be divided into two main categories, the *cognitive interaction support* and the *intuitive interaction support.*

5.2.2.1 Cognitive interaction support

Cognition is the science of the human mind, the science of humans as information processors; it could also be defined as the behavioural science, nothing more than cognitive psychology. While there is no unified cognition science definition, the definitions share the same fabric, i.e. human mind (Boring, 2002).

Various cognitive science studies are done or still done in many research fields; some are related to machine learning, some are related to artificial intelligent and other are related to human computer interaction (HCI). The last aforementioned research field (HCI) is the one involved in our research.

Before continuing, different terms and hypotheses should be defined and clarified.

The first term is *cognitive effectiveness*, which is defined as the speed, ease and accuracy with which a representation can be processed by the human mind (Larkin and Simon, 1987). Cognitive effectiveness determines the ability of a visual notation (Mashup Maker in our case) to both communicate with business stakeholders and support design and problem solving by software engineers. Moody (2009) states that it is also important to mention that the cognitive effectiveness is defined as a primary-dependent variable for evaluating and comparing visual notations and the primary design goal in constructing them, this variable is operationally defined and can therefore be empirically evaluated.

Cognitive integration is also important in this context. In software engineering, problems are typically represented by systems of diagrams rather than single diagrams; it applies equally to diagrams of the same type (homogenous integration) (for example data flow diagrams (DFT)) or of different types (heterogeneous integration). Multiple diagrams place additional demand on the reader to mentally integrate them (Moody, 2009).

Cognitive task analysis is concerned with characterizing the decision making and reasoning skills of subjects as they perform activities involving the processing of complex information (Preece et al, 1994).

5.2.2.2 Intuitive interaction support

Oxford English dictionary defines the word *intuition* as "knowledge or mental perception that consists in immediate apprehension without the intervention of any reasoning process".

Intuition is an immediate pre-reflective experience and only possible in duration. Duration is understanding of time that prolongs "the past into a present which is really blenching into the future". Intuition is also a kind of experience; perception is an external, material experience and intuition a vital, inner experience. Intuition is a mode of contemplation that postpones bodily actions; it requires reversing the customary direction of thought towards utility, it open space for creating changes or difference, ideas …. Intuition is finding only the old in the new (Bergson, 2002).

A technical system is *intuitively usable* if the user's unconscious application of prior knowledge leads to effective interaction (Mohs et al, 2006), or as state by Raskin (1994), intuitiveness stands for "readily transferred existing skills" or familiarity.

Intuitiveness supports usability aims for issues such as effectiveness and efficiency. The more intuitive an interface is the faster the users can reach their goals and the better a system performs in terms of user traffic throughput. Despite drawing on unconscious processes, intuitiveness becomes both quantifiable and a commodity that can be tested (Kaltenbacher, 1999).

5.3.3 Functional Support

The functionality provided by a system is in general an important issue that influences the satisfaction of the users. As satisfaction is in general considered as an aspect of usability, functionality also influences usability.

When talking about the functional support of a Mashup Maker it directly means talking about the way it satisfies the functional requirements of its end-users. In particular, a Mashup Marker's functional requirements are specific targeted towards casual users and not towards programmers.

The functional requirements of a system indicate what the system should do (Lausen, 2003). The functional requirements of a system are usually described in terms of the system's input, output and the relation between the two (Lausen, 2003). Many studies and research works are describing and handling the functional requirement of systems (e.g., (Van Lamsweerde and Letier, 2000), (Castro et al, 2001), (Lausen, 2003)). However traditional functional requirements specify the system's role but ignore the system context (Lausen, 2003). In our usability research of Mashup Makers, we found that the appropriate specifications for describing the functional requirements of a Mashup Maker are those that do not ignore the system context. An example of such a method of functional requirement specification is given by Lausen (2003) and is called the *Task and Support method*. The Tasks and Support method uses annotated task descriptions. They specify what the computer and user shall accomplish together without indicating which

actor performs which parts of the tasks. This method produces higher-quality requirements that are faster to produce and easy to verify and validate (Lausen, 2003). The Task and Support method is inspired by Alistair Cockburn's use case, for which the use case tends to describe what the system does and how it interacts with the user (Cockburn, 1997; 1997; 2001).

5.4 Conclusions

In this chapter, we presented a Conceptual Evaluation Model for Mashup Makers for end-users. We also justified each component of this conceptual model. The purpose of this conceptual model is to identify the key indicators for the usability of Mashup Makers. As already indicated, the three usability aspects identified are overlapping and intersecting. For instance, the reader may find concepts within the cognitive interaction support (described in subsection 5.2.2.1) also oriented towards visual support. This is because of their nature as impact factor of usability. As our research objectives is to investigate the usability of Mashup makers we find it essential to ground our research on what we found in the literature.

In the next chapter, this Conceptual Evaluation Model will be used as the basis for our usability evaluation framework.

Part III: The usability evaluation framework and its evaluation

---

CHAPTER 6: THE MASHUP MAKER USABILITY EVALUATION FRAMEWORK: MUEF

In the previous chapter, we developed a conceptual model for usability factors for Mashup Makers for end-users. Further, we explained the different components of that Conceptual Evaluation Model. In this chapter, we address the development of an analytical framework for usability evaluation of Mashup Makers for end-users.

This chapter is organized as follow: Section 6.1 presents the approach used for the usability evaluation framework and discusses the requirements that should be satisfied in such a framework. Section 6.2 presents the architecture of the proposed framework, its layers and components and the relation among the layers and components. Section 6.3 presents the usability quantification process associated with the proposed framework. Finally, section 6.4 concludes the chapter.

6.1 The Approach of the Framework

The framework for evaluating the usability of Mashup Makers as Web application development environments for end-users is developed based of the set of collected usability problems obtained from our literature review and the empirical studies described in Chapter 4 (the pilot study and the user experiment/study), and the Conceptual Evaluation Model, presented in Chapter 5, which identifies 3 different aspects of the usability evaluation of Mashup Makers. In addition, we adopted the general architecture of the "Quality in Use Integrated Measurement" (QUIM) model of Donyaee et al. (2002) (see figure 6.1). This QUIM model (Donyaee et al, 2002) (Seffah et al, 2006) is a hierarchical model of usability measurement that unifies various usability models into one single consolidated model. QUIM is hierarchical and multilayered in that it decomposes usability into factors, then into criteria, and finally into specific metrics. As shown in figure 6.1, the QUIM model is presented as a triangle that consists of 5 layers. The upper layer in dedicated to the quality in use, where the usability of the evaluated object or product is identified. The second upper layer is the factor level, where the usability is determined by abstract factors. Then the third upper layer is the criteria level where the factors are less abstract and the usability could be measured by metrics derived from the criteria. The next layer is the metrics layer, where the actual usability evaluation factors exist. The lowest layer of the QUIM model is the data layer. The data

layer presents the raw usability data that relate to the layer presented in figure 6.1 as primary and secondary artefacts. There are also two arrows at both sides of the triangle, the right side arrow represents the testing and prediction in the usability evaluation process and the left side arrow represents the specification process of every component in every layer of the QUIM model.

The layered approach of the QUIM model, where usability is decomposed into factors, criteria and so on is very interesting and useful as it brings more structure in the evaluation process. Therefore, we adopted this multilayer structure, as well as the goal-means relationship between the layers. However, we have tailored the different layers to the context of the usability evaluation of Mashup Makers and Web application composition environments for end-users. In this way, the framework provides an approach to quantify usability of Mashup Makers; several usability aspects are collectively measured to give a single score. More details on how this framework works are provided in sections 6.2 and 6.3.

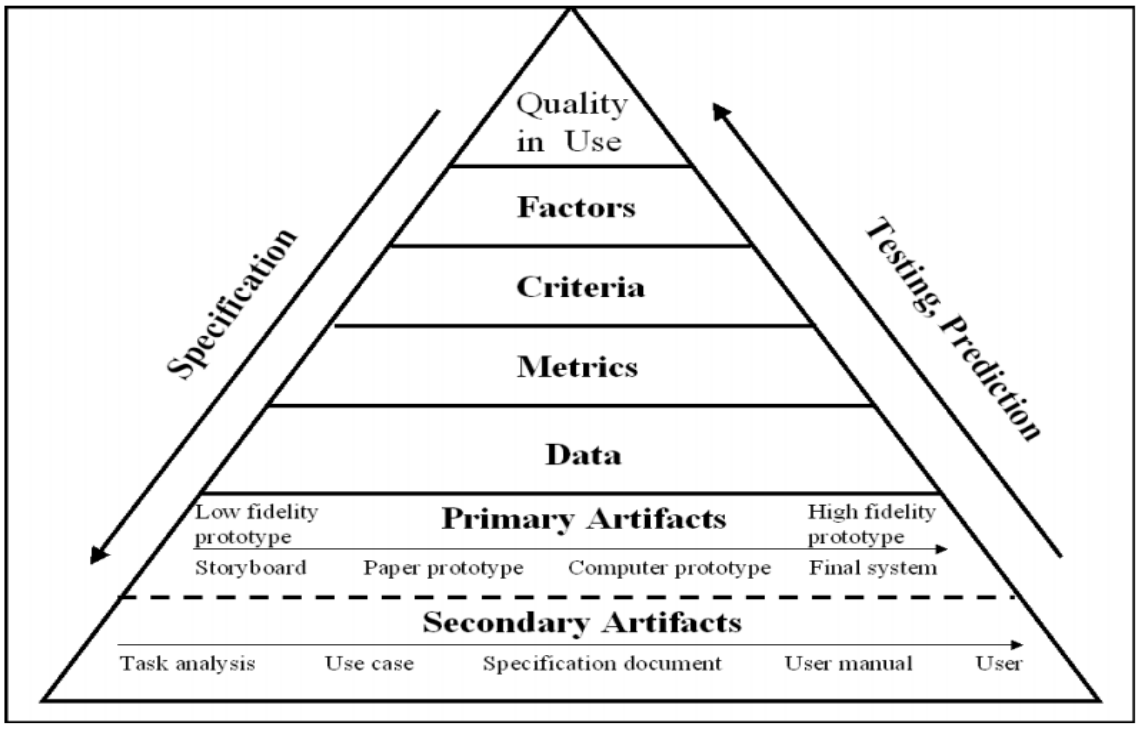

Figure 6.1: QUIM model (Donyaee et al, 2002)

6.1.1 Requirements for MUEF

To guaranty a workable usability evaluation framework, we selected the more relevant eight requirements that a usability evaluation framework needs to satisfy. Those requirements were derived from a literature study on usability framework requirements. The requirements are as follows:

(1) *Fact-based approach*: the proposed framework should be based on a set of facts. In our case, those facts are theoretical and empirical investigations, as well as the usability problems that need to be taken into consideration to establish a usability evaluation framework for Mashup Makers as they might be standards (benchmarks) usability problem lists (Hartson et al, 2003).

(2) *Hierarchal approach*: the framework needs to take into consideration the fact that usability factors and problems have different abstraction levels (Dillon, 1999), (Dillon, 2001).

(3) *Modularization*: the proposed framework needs to allow usability practitioners to take in account only some parts of the user interface, which can then be redesigned with no influence of others parts of the user interface (Treu, 1994), (Carliner, 2003), (Hartson, 2003).

(4) *Optimization*: the proposed framework needs to facilitate inference in an optimized way for economically evaluating usability issues in terms of what user interface features to be examined and what types of tasks to be performed in the design process of a Mashup (Kwahk and Han, 2002).

(5) *User-oriented*: the proposed framework should provide easy to use elements for usability practitioners, to easily allow them to learn and use the framework and practice it (Hertzum and Jacobsen, 2001).

(6) *Context of use-based*: the proposed framework should help usability practitioners understanding the context of use of the usability evaluation process within the framework

(Bautsch et al, 2001). Specifying the context of use of a methodological framework is defined by "Identifying the people who will use the methodological framework, what they will use it for, and under what conditions they will use it" (Rochford, 2009).

(7) *Design-oriented*: the proposed framework needs to provide mechanisms and elements to help creating better Mashup design environments after indentifying potential usability problems (Hartson et al, 2003), (Kadyte and Tetard, 2004).

(8) *Implementable*: The proposed framework should provide an easy way to be implemented through a detail set of guidelines (checklists) and/or usability benchmarks/standards (Zhang and Adipat, 2005).

6.2 MUEF Architecture

The architecture of MUEF is shown in figure 6.2. In the following subsections we describe the features of the framework, the layers (levels), components and the relationships between the layers, as well as the method for the usability quantification.

6.2.1 Indicator Level

While usability cannot be accurately and fully evaluated in any way, it can be estimated or evaluated by some Usability Indicators that provide a basis for decision making (Ham et al, 2006). Based on the Conceptual Evaluation Model presented in chapter 5, we propose to estimate and evaluate the usability of Mashup Makers by three main usability indicators: *Visual Support*, *Functional Support*, and *Interaction support*.

The top level of the framework hierarchy (see figure 6.2) contains these usability indicators or usability factors, sometimes also called usability views. Usability indicators are abstract conceptual constructs that cannot directly be measured but aim to connect observable and measurable usability criteria at the criteria level (the level below in the hierarchy). The different usability indicators are explained in detail in the Conceptual Evaluation Model (see section 5.2 in chapter 5).

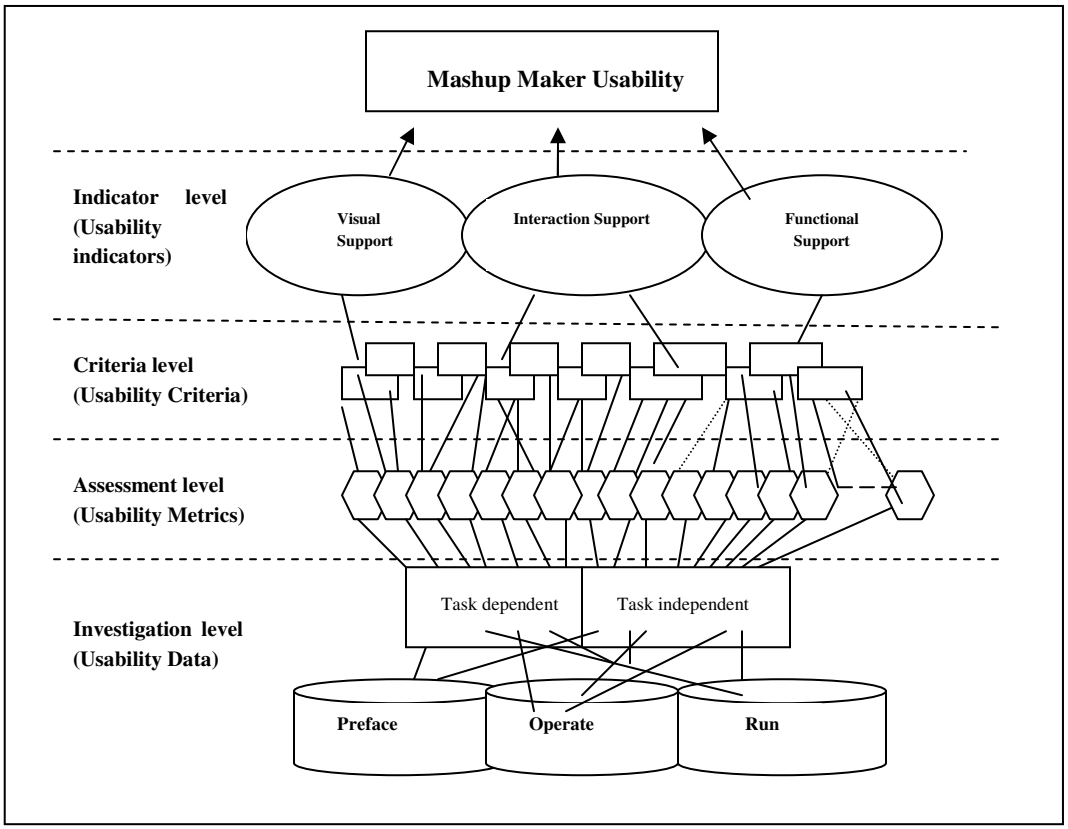

Figure 6.2: Mashup Maker Usability Evaluation Framework (MUEF)

6.2.2 Criteria Level

The next level in the MUEF is the Usability Criteria level. In figure 6.2, usability criteria are represented by rectangle shapes. There are 14 criteria. Usability criteria can be directly measured through at least one specific usability metric (usability metrics will be described in next level, the assessment level). Different usability criteria can contribute to different usability indicators. Every usability indicator in the indicators level should have a relationship with at least one usability criterion in the criteria level.

The usability criteria were identified taken into account the literature, the pilot study, and the user experiment. We adopted the 14 CD framework dimensions (Blackwell and Green, 2000; 2002) as usability criteria but redefined them for the context of the Mashup Makers. In table 6.1 we present and describe the usability criteria used in the MUEF framework. The usability criteria are mapped to the usability indicators in the upper level.

The relationships between the usability criteria and the usability indicators in the upper level (the indicator level) are shown in table 6.2. Other or more usability criteria could be adopted, but we found that those of the CD framework are most relevant for Mashup Makers and they showed to be useful and appropriate in our user experiment presented in Chapter 4.

Table 6.1: Usability criteria.

Usability criterion	**Description**
Visibility	The ability to perceive components easily. In the case of the Mashup Makers, this means whether the required Mashup components are visible without major cognitive work.
Hard mental operation	The degree to which users need to resort to fingers or pencil annotations to keep track of the process, or experience difficulties during the process.
Diffuseness	The verbosity of the Mashup Maker's interface and tools. In visual design environments as well as in Mashup Makers, some notations can be annoyingly long-winded, or occupy too much valuable "real-estate" within a display area, e.g., big icons and long words reduce the available working area.
Abstraction gradient	The ability of grouping elements in order to be able to treat them as one element.
Consistency	The degree to which similar components are semantically expressed by similar syntactic or visual forms
Viscosity: resistance to	A viscous system (in our case a Mashup Maker) needs many user actions to accomplish one goal.

change.	E.g., changing all headings to upper case may need one action per heading. Environments containing suitable abstractions can reduce viscosity.
Premature commitment: constraints on the order of doing things.	Self-explanatory. Examples: being forced to declare identifiers too soon; choosing a search path down a decision tree; user has to select components before he/she choose relations and links.
Hidden dependencies: important links between entities are not visible.	If one component cites another component, which in turn cites a third one, changing the third component may have unexpected repercussions. Examples: style definitions; complex recursions (i.e. changing in loop parameters has unexpected effect on the whole program).
Role-expressiveness: the purpose of an entity is readily inferred.	Role-expressive notations make it easy to discover why the composer has built the structure in a particular way; in some notations each component looks much the same and discovering their relationships is difficult. Assessing role-expressiveness requires a reasonable conjecture about cognitive representations (see the user interaction explanation in of section 5.2 in chapter 5) but does not require the analyst to develop his/her own cognitive model or analysis.
Error-proneness:	A system is error-prone if it invites slips, errors and mistakes and gives little protection for errors. Prevention (e.g., check digits, declarations of identifiers, etc) can redeem the problem.
Secondary notation: extra	Users often need to record things that have not been anticipated by the designer. Rather than

information in means other than formal syntax.	anticipating every possible user requirement, many systems support secondary notations that can be used whenever the user likes. One example is marginal comments (beside the design canvas of the Mashup malker user interface); another is the use of colours or format choices to indicate information additional to the content of text.
Closeness of mapping: closeness of representation to domain.	How closely related is the Mashup component to the result it is describing? Within the Mashup creation process, the Mashup creator should understand the relationship between components he/she uses to create the Mashup and the output he aims for.
Progressive evaluation	The degree to which work can be checked at any time. Evaluation is an important part of a design process, and Mashup Maker can or cannot facilitate progressive evaluation by allowing users to stop in the middle of their design to check work so far, find out how much progress has been made, or check what stage in the work they are up to.
Provisionality: degree of commitment to actions or marks.	Premature commitment refers to hard constraints on the order of doing tasks during the Mashup creation process, but whether or not hard constraints exist, it can be useful to make provisional actions – recording potential design options, sketching, or playing "what-if" games. (i.e. in some cases defining a component in a Mashup creation process before link this component to other is a premature commitment)

Our approach to map usability criteria to usability indicators is based on a rational analysis of each usability indicator and usability criteria. This rational analysis was inspired by the research work achieved on QUIM and usability investigation by Donyaee et al. (2002) and Seffah et al. (2006). Also, the rational analysis is derived from the literature and the experimental studies we performed and presented in chapter 3 and chapter 4.

Table 6.2 summarizes the relationships between the three usability indicators and the fourteen usability criteria. This relationship is established by means of providing mappings between usability criteria and usability indicators. These mappings are derived from the rational analysis and are highlighted and explained below case by case.

Visibility: Clearly visible objects, components and mechanisms that allow end-users to perceive the Mashup maker's functionality and interaction will contribute to the usability with respect to visual support and interaction support.

Hard mental operations: If there are mechanisms that facilitate the design process of a Mashup (and avoid that users have to use additional tools), then this will contribute to the usability with respect to visual support, interaction support and functional support.

Diffuseness: The proper use of diffuseness (i.e. use of space in the user interface and design area) will facilitate the perception of objects, components and interaction mechanisms and therefore contribute to the usability with respect to visual support and interaction support.

Abstraction gradient: If there are objects and mechanisms that facilitate the grouping of elements, this will simplify and speed up manipulations and therefore contribute to the usability with respect to visual support and functional support.

Consistency: It is generally accepted that consistency of an interface will contribute to the usability of an interface, so consistency will contribute to the usability with respect to visual support and interaction support.

Viscosity: If there are mechanisms to prevent or decrease resistance to change this will contribute to the usability with respect to interaction support and functional support.

Premature commitment: Constraints on the order of doing things will make the system less flexible and therefore it will have an influence on the usability with respect to interaction support and functional support.

Hidden dependencies: If there are mechanisms to uncover dependencies between elements and that facilitate the perception of such dependencies or if no such dependencies exist, then this will contribute to usability with respect to visual support, interaction support and functional support.

Role-expressiveness: If the role of design elements and process is obvious or there exist mechanisms to reveal their role, then this will contribute to usability with respect to visual support, interaction support, and functional support.

Error-proneness: If there are mechanisms to prevent errors or undo errors, then this will contribute to the usability with respect to interaction support and to functional support.

Secondary notation: The availability of secondary notations can contribute to the usability with respect to visual support, interaction support, as well as functional support. For instance, the use of colour to annotate or highlight certain information will be useful for the visual support, while the possibility to make annotations or notes will be useful for the interaction support and the functional support

Closeness of mapping: Interfaces that stay close to the mental model of the user and to the domain under consideration have a higher usability as the interface and the concepts used are more familiar to the user. Therefore, closeness of mapping will contribute to the usability with respect to visual support (better understanding of the interface) and functional support (better knowing what to do).

Progressive evaluation: If there are mechanisms to allow checking work progress at any time (or regularly) during the design process, this will contribute to the usability with

respect to interaction support and functional support, as it will provide early feedback on what has already been done.

Provisionality: Provisional actions will increase the flexibility of the system and allow to experiment with different options, which will contribute to the usability with respect to interaction support and functional support.

Table 6.2: The relationships between usability indicators and usability criteria

Usability indicator	Usability criteria contributing
Visual support	Visibility, Consistency, Abstraction Gradient, Hard mental operation, Hidden dependencies, Closeness of mapping, Secondary notation, Diffuseness, Role-expressiveness
User Interaction support	Visibility, Error proneness, Viscosity, Provisionality, Consistency, Role-expressiveness, Premature commitment, Hard mental operation, Secondary notation, Diffuseness, Hidden dependencies, Progressive Evaluation.
Functional support	Hidden dependencies, Role-expressiveness, Progressive Evaluation, Error proneness, Provisionality, Premature commitment, Abstraction Gradient, Closeness of mapping, Secondary notation, Hard mental operation, Viscosity

It should be clear that the contribution of the different usability criteria to a usability indicator could be different. Some criteria may have a higher impact than others and this will also depend on the Mashup maker being evaluated. Therefore, the evaluators are asked to give a weight to the different criteria per usability indicator, which is then used

to calculate the score for the usability indicator. This score is done automatically by filling-in a Microsoft Excel sheet prepared for this purpose (see appendix 4). This will be described in more detail in section 6.4, the usability evaluation procedure and quantification with MUEF.

6.2.3 Assessment Level (Usability Metrics)

The IEEE metrics standard (Paul et al, 1999) defines a software metrics as "a function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software processes a given attribute that affects its quality" (Boring, 2002). In the usability indicators context adopted in our research and implemented in MUEF, the usability metrics' output could be defined as quantitative and qualitative values that summarize the status of specific criteria. In figure 6.2, usability metrics are represented by a hexagon.

We call this level the Assessment level, because the usability metrics given in this level allow the assessment of the usability criteria.

For the assessment level, we have defined 42 usability metrics, i.e. 3 questions or statements per criteria; table 6.3 gives the usability criteria and their related usability metrics. Those questions/statements can be used as metrics (e.g., in questionnaires). Those questions/statements are actually derived from the definition of the usability criteria (see table 6.1) and from the questionnaire adopted from the CD of notations framework (Blackwell and Green, 2000; 2002) but adapted to the context of our research on Mashup Makers.

Table 6.3: Usability criteria and their related usability metrics (questions/statements).

#	**Usability criteria**	**Usability Metrics – questions/statements**
1	Visibility	Is the visual interface of the Mashup Maker clear?
		Does the Mashup Maker have easy to understand and perceive user interface components?

		Does the Mashup Maker allow to find user interface components easily and to see related components at the same time?
2	Hard mental operations	Is it easy to keep track of tasks performed and components used during the creation process of the Mashup?
		Is it easier to make frequent tasks (e.g., in fewer steps) than non-frequent tasks? Does the difficulty of the tasks match their frequency?
		Is the design of the Mashup easy and needs few intuitive thoughts?
3	Diffuseness	Does the Mashup Maker provide sufficient design space?
		Do the visual interface components have proper size and shape?
		Does the Mashup Maker provide space friendly component linking mechanisms?
4	Abstraction gradient	Does the Mashup maker give you any way of defining new facilities or terms within the notation, so that you can extend it to describe new things or to express your ideas more clearly or succinctly?
		Is it easy to group components and can such a group be used as one element?
		Is it easy to group tasks and can such a group be used as one element?
5	Consistency	Are similar graphical entities (i.e. icons and menus) provided by the Mashup Maker laid out the same way?
		Are semantically related components expressed by similar visual forms?

		The Mashup Maker provides a consistent user interface (consistent naming of components, similar elements have similar meaning, etc)
6	Viscosity	Are only few actions needed to accomplish tasks during the Mashup design process?
		Does the Mashup Maker provide suitable/understandable components that make it easy to change things after a design has been completed?
		Is it easy to distinguish process cascading during the Mashup design process?
7	Premature commitment	In case of cascading multiple steps, a step usually doesn't require premature commitment from the previous step.
		Does the Mashup Maker provide the ability to change components or to redefine them during the Mashup design?
		Are you able to select the order you liked when doing tasks during Mashup design process?
8	Hidden dependencies	Are dependencies, if any, clearly visible?
		Do changes in one part of the visual interface affect other parts?
		In case there is cascading of multiple steps, are dependency between steps clear?
9	Role-expressiveness	Do components have semantically meaningful names?
		Are most parts of the visual interface of the Mashup Makers easy to interpret?
		In case of steps cascading, the achievement of a step doesn't require understanding of steps still to

		be achieved?
10	Error-proneness	It is difficult to make mistakes?
		Does the Mashup Maker prevent making mistakes?
		When making mistakes, is it easy to undo the mistakes?
11	Secondary notation	Does the Mashup Maker provide facilities and spaces to leave comments and notes (marks) or mechanism for brainstorming?
		Does the Mashup Maker provide alternative notations?
		Does the Mashup maker provide easy and user friendly help tutorials?
12	Closeness of mapping	Are all components provided by the Mashup Maker relevant?
		Is the workflow for creating the Mashup self-explaining and clear?
		The visual interface of the Mashup maker and its components are closely related to the process of creating Mashups.
13	Progressive evaluation	Is it easy to stop in the middle of the design process and check your work so far?
		Does the Mashup Maker provide methods to review completed or semi completed tasks?
		Is it possible to try out partially completed Mashups?
14	Provisionality	Is it possible to sketch things out when you were playing around with ideas during the design process of the Mashup?
		Does the Mashup Maker provide methods and components to provisionally re-perform completed

		or semi completed tasks?
		Does the Mashup maker provide ways to do "what if games" during the design process of the Mashup?

It is worth mentioning here that we agree on the fact that the more metrics we have the more accurate and precise usability evaluation can be done. However, it would be very time consuming for the user of MUEF to give scores for more than 50 questions, given the fact that there are other activities in the evaluation process that also consume time. So, we have limited the number of usability metrics as to optimize the time needed for using MUEF by usability practitioners.

6.2.4 Investigation Level (Usability Data)

The lowest level in MUEF is the data level or what is called the Investigation Level as shown in figure 6.2. Usability data in the context of MUEF is the data required for the usability metrics. Usability data can be qualitative or quantitative. Usability data can be collected from different sources including users and usability specialists, questionnaires, surveys, user documentation, design artefact, task analysis, video or other information material, user needs, etc (Donyaee et al, 2002). As applied in our investigation of the usability of some Mashup Makers (see Chapter 4), we have reduced the process of the collection of usability data into the process of creating a Mashup as done by an end-user. The process consists of three main steps: Firstly, the *preface* step that includes understanding the Mashup design environment, resources to be used in the design, and preparation for the design. Secondly, the *operate* step that represents the actual design process of the Mashup including the choosing of components, linking, defining operation and controls if needed and manipulating the structure of the Mashup within the design area of the Mashup Maker. Finally, the *run* step that represents the execution of the Mashup and delivering the output of the Mashup as Web page having the requested information. Collected information for the usability data level can be classified into two main categories *task dependent* and *task independent;* this is to facilitate the interrelation

with the usability metrics in the upper level, the assessment level, where the evaluator (MUEF user) should give scores for the questions (metrics). Also this classification is considered as one of the qualitative results of the usability evaluation using the MUEF.

In subsections 6.2.1, 6.2.2, 6.2.3 and 6.2.4, we presented in details the relationship between MUEF layers as well as the mapping between the usability evaluation factors (indicators to criteria and criteria to metrics).

6.3 Usability Evaluation and Quantification with MUEF

It is worth re-mentioning (as already explained in the research objectives in chapter 1) that MUEF targets usability practitioners (usability experts) and Mashup Maker designers. In this section, we use the term 'evaluator' to indicate the user of MUEF. An evaluator has to have good knowledge and experience in web interface design and evaluation, and web usability evaluation and testing concepts and practices.

Furthermore, in this section, we use the term 'observer' to indicate the person who supervises the usability evaluation process of a Mashup Maker using MUEF. As MUEF is a context of use-based framework (as mentioned in subsection 6.1.1 and section 6.4), it should be used by an evaluator but the presence of an observer may provide help as well as organizing materials and resources and reduce the time of the evaluation process.

In this section, we first present the usability evaluation procedure to be used in order to apply MUEF in evaluating the usability of a Mashup Maker. Then we present and clarify the usability quantification methodology used in MUEF.

6.3.1 Usability Evaluation Procedure for MUEF

To use MUEF for evaluating the usability of a Mashup Maker, the evaluator should follow the stages shown in table 6.4. Table 6.4 summarizes the usability evaluation procedure of MUEF. Seven main stages are required to be followed by the evaluators. The first stage is the introduction of MUEF and the preparation for the usability evaluation of the Mashup Maker under consideration. In this stage, an introduction of the

MUEF philosophy, layers and methodology is presented. A preparation of the Mashup Maker usability evaluation is presented too. This includes indentify the Mashup to be created, its main steps, components and the output that is to be obtained.

The second stage is the creation of the Mashup using the Mashup Maker under consideration. In this stage the evaluator uses his/her skills as well the materials and resources needed to perform the Mashup creation task. This is to let the evaluator be involved in an actual creation task as should be done by an end-user. The third stage after finish creating the Mashup is to collect the usability data. To facilitate this task we prepared a sheet with two simple columns where the evaluator needs to indentify task dependent and task independent activities of those subtasks he/she has done during the creation of the Mashup. The fourth stage is dedicated to filling-in the metrics sheet where the evaluator needs to give scores to the questions in the questions list we provide. In the fifth stage, the evaluator needs to fill in the factor/weight sheet, by this he/she give his score (weight) to every metric, criteria and indicator. In the sixth stage, the evaluator needs to fill-in the feedback (questionnaire) sheet where he/she answers direct questions about the evaluation process he/she has done as well as to give feedback on the usefulness and effectiveness of the MUEF. The feedback questionnaire is optional and it is made available to obtain feedback from MUEF users in order to improve MUEF. The last stage is to finalize the usability evaluation by reviewing together with the observer the sheets and giving the final quantification as well the qualitative evaluation of the usability of the Mashup Maker considered. If no observer is available, the evaluator himself collects the final usability quantifications and feedbacks of the evaluated Mashup Maker.

All the sheets resources, materials used in the evaluation procedure (process) are provided in appendices 4 and 5. It is worth mentioning that the evaluator should only give his/her score of metrics and his/her weight of usability criteria and usability indicators and the computation of the averages of metric scores and the averages of weights and the mapping between layers is done automatically by providing a Microsoft Excel sheet to be filled in by the MUEF user (see appendix 4).

Table 6.4: Usability evaluation procedure for MUEF

Stage	Description
1	Introduction to MUEF and preparation for the usability evaluation of the Mashup maker under consideration
2	Performing the Mashup creation
3	Identify usability data of the investigation layer as explained in subsection 6.2.4 by indentifying dependent and independent tasks
4	Filling-in the questions and answer sheet to give scores for the usability metrics in the assessment layer
5	Give ranking (weight) for usability factors (metrics, criteria and indicators) by filling-in the factor/weight sheet
6	Fill-in feedback (questionnaire) sheet
7	Reviewing sheets (possible together with the observer) if needed and having the usability of Mashup Maker considered by the quantitative and qualitative means

6.3.2 Usability quantification within MUEF

In principle, the quantification of the usability in MUEF starts by identifying the usability data in the lower level of MUEF, but practically it starts by answering the questions listed in the metric level. We use a 5-point Likert-scale (Wuensch, 2005) for scoring every question to be answered by the evaluator in the question lists (5 for strongly agree, 4 for agree, 3 for neither agree nor disagree, 2 for disagree, or 1 for strongly disagree). Then all scores are collected for all the metrics and mapped to the relevant usability criteria in the upper level. As mentioned in the MUEF layer explanations in section 6.2, the mappings between the usability indicators and usability criteria and between the usability criteria and usability metrics is provided by MUEF (presented in section 6.2) but the evaluator should give a weight based on the importance of the metrics/criterion for the particular Mashup Maker. The computation of the averages of metric scores and the averages of

weights and the mapping between layers is done automatically by providing a Microsoft Excel sheet to be filled in by the MUEF user (see appendix 4).

Table 6.5: Likert-scale score used for metrics

	Strongly agree	**Agree**	**Neither agree nor disagree**	**disagree**	**Strongly disagree**
Question/metric	5	4	3	2	1

Finally the criteria are weighted and mapped to the relevant usability indicators in the upper level. Then by calculating the average of every usability indicator and the standard deviation we have a quantitative value for the three main usability indicators of Mashup Makers. Collecting the ranks and calculating the averages can be done using a Microsoft excel sheet (see Appendix 5). A MUEF user may also leave his/her comments by filling out a mini questionnaire (on a sheet) in every stage of the evaluation process. Those comments and feedback are collected and resumed as the qualitative results of the usability evaluation process of a Mashup Maker.

Remember that MUEF targets usability practitioners, usability experts and Mashup Makers designers. Those people have enough knowledge and experience to follow our usability evaluation procedure for evaluating Mashup Makers by following the quantification approach we presented. Nevertheless, we offer as much as possible easy and user-friendly methods such as filling-in some paper sheets and Microsoft excel sheets as mentioned in subsection 6.3.1 (see Appendix 4).

6.4 MUEF: Satisfying the Requirements?

As mentioned in subsection 6.1.1 there are several condition that should be satisfied by a useful and effective usability evaluation framework. Herewith we present how the

proposed framework MUEF satisfies the eight conditions that we formulated for a successful usability evaluation framework. This is clarified in the following:

1- MUEF is a fact-based approach framework: our framework is based on a set of usability investigations both theoretical, as in chapter 3 and 5, and empirical, as those in chapter 4.

2- MUEF has a hierarchal approach: this is clearly shown in its architecture as multilayer hierarchal framework based on usability evaluation impact factor of Mashup Makers for end-users.

3- MUEF provide a high level of modularization: this is also evident in MUEF as any usability evaluator of a Mashup Maker may follow the steps of evaluation by focusing on only a part of the user interface of a Mashup Maker without being needed to consider every part of the user interface of the Mashup Maker.

4- MUEF provides high level of optimization as it provides an optimized way for economically evaluating the usability of Mashup makers by minimizing the time and work and by means of available computational resources (a computer, Internet connection, Web browser, follow-up sheets …etc). A usability evaluation of a Mashup maker using MUEF first needs to recruit a usability practitioner. Providing him/her by an introduction of MUEF (if needed) and the Mashup Maker to be evaluated, the needed computational resources and the needed time. From our validation exercise performed (see Chapter 7), we found that in general 3 hours at most is needed for the actual evaluation.

5- MUEF is a user-oriented: as described in table 6.4 (usability evaluation procedure of MUEF) and chapter 7 (MUEF evaluation), the MUEF is an easy to use framework for usability evaluation as it provides a user friendly step by step evaluation plan (see table 6.4 and figure 7.1) and usability practitioners only need to fill-out some sheets and answer a limited number of questions in order to benefit from the framework to evaluate the usability of a Mashup Maker.

6- MUEF is a context of use-based framework: MUEF is a usability evaluation framework specific for Mashup Makers for end-users and is developed based on both theoretical and empirical investigations of usability of Mashup Makers; all questions to be answered by the evaluators are adapted to the context of use i.e. the process of evaluation of Mashup Makers by usability practitioners.

7- MUEF is design-oriented as it considers the process of designing a Mashup in the evaluation process. Also, the evaluation process provides Mashup Maker designers with practical and potential usability guidelines and usability aspects that should be taken in consideration (the user interface aspect, the user interaction aspect, and the functional aspect).

8- MUEF is implementable as it provides a set of step by step procedure to evaluate the usability of Mashup Makers. In addition, a number of question and answer lists to be fill-out by usability practitioners are provided.

6.5 List of Usability Guidelines for Mashup Maker Designers.

Based on the finding with our usability studies and experiments and the development of MUEF, we have compiled a list of usability guidelines that can be used by Mashup maker designers to improve their products or to be taking into consideration when designing new products. Herewith we present these usability guidelines for Mashup maker designers.

1) The Mashup maker should provide a clear user interface with clear components and easy to perceive and understand functions (e.g., using metaphors).
2) The visual interface of the Mashup maker and its components should be closely related to the process of creating Mashups (e.g., by using proper metaphors).
3) The Mashup maker should provide sufficient design space, visual interface components should have proper size and shape, and the components linking mechanisms should be space friendly.
4) The Mashup maker should use consistent layouts for similar graphical elements (i.e. icons and menus) at different places. Also semantically related components

should be expressed by similar visual elements and similar elements should have similar meaning.

5) The Mashup maker should make dependencies between components and cascading tasks clearly visible.
6) The Mashup maker should provide well-defined components that let a user keep track of the tasks performance sequence.
7) The Mashup maker should provide a design process consisting of a few intuitive design tasks.
8) The Mashup maker should provide design tasks that need only a few actions to be accomplished and provide suitable/understandable actions that make it easy to change things after a design has been made.
9) The Mashup maker should provide the ability to select the order of actions when doing tasks. Within the design process it should be easy to see the order in which the different tasks need to be performed (e.g., task cascading).
10) The Mashup maker should only provide components relevant to the creation of Mashups.
11) The Mashup maker should provide the user with the ability to define new facilities or terms within the notation that help in extending it to describe new things or to express users ideas.
12) The Mashup makers should provide mechanisms that make it easy to group components and tasks and to use such a group as a single element.
13) The Mashup maker should provide ways and mechanisms that reduce the need for premature commitments between design steps.
14) In case of step cascading the Mashup maker should provide mechanisms that reduce the need that performing a step requires understanding the next step(s) to be achieved.
15) The Mashup maker should provide mechanisms that prevent making mistakes as much as possible and in case of mistakes the Mashup maker should provide easy way(s) to undo mistakes.
16) The Mashup maker should provide facilities and spaces that allow a user to make/leave comments and notes (marks) and mechanisms for brainstorming.

17) The Mashup maker should provide mechanisms that allow a user to stop in the middle of the design and check his/her work so far, i.e. provide mechanisms to review completed or semi completed tasks/designs. In addition, the Mashup maker should provide ways to try out partially completed Mashups.
18) The Mashup maker should provide methods and components that allow to provisionally re-perform completed or semi completed tasks. Also the Mashup maker should provide ways to do "what if games" during the design process of Mashup.
19) The Mashup maker should provide an easy to use help facility, user-friendly tutorials and API documentations.
20) The Mashup maker should clearly mention the requirements (e.g., minimal Internet connection, browser, OS, etc.) that need to be satisfied to use the tool.

The aforementioned usability guidelines could be categorized into three main categories according to the three main usability aspects presented in chapter 5 (the visual support, interaction support, and functional support). However, note that there is an overlap between the three categories and some usability guidelines could be considered under two or even under all categories. This is already explained in chapter 6, section 6.3 where we explained that the usability criteria in MUEF are overlapping and can be linked to different usability indicators. The usability guidelines 1, 2, 3, 4, 5, 6, 9 could be consider under the first category, the visual support. The usability guidelines 4, 5, 7, 8, 9, 11, 12, 13, 14, 15, 16 could be consider under the second category, the interaction support. And the usability guidelines 2, 10, 11, 13, 14, 15, 16, 17, 18, 19, and 20 could be considered under the third category, the functional support.

6.6 Conclusions

In this chapter we have presented a usability evaluation framework for Mashup Makers for end-users. The proposed framework consists of different abstraction layers of usability factors related to each other by goal-means relations. An associated usability evaluation procedure consisting of several stages is also presented. The quantification

process of the usability metrics is also presented. The next chapter presents the evaluation and validation of the proposed framework.

CHAPTER 7: MUEF EVALUATION

The previous chapter presented MUEF as a multi-layers usability evaluation framework for Mashup Makers for end-users and showed how to use it in for the usability evaluation of Mashup Makers.

In this chapter, we consider the evaluation and validation of MUEF. The chapter is structured as follows. Section 7.1 explains the approach used for the validation and evaluation of MUEF. Section 7.2 presents the evaluation process that we performed for MUEF. Section 7.3 presents the results of the evaluation and section 7.4 discusses how this evaluation has/will influenced MUEF. Finally, the conclusions of the chapter are presented in section 7.5.

7.1 The Approach

Different approaches can be used to validate and evaluate MUEF. One possibility is to conduct a comparative evaluation with similar or related frameworks. However, to the best of our knowledge there are no similar or related frameworks for the usability evaluation of Mashup Makers for end-users. However, in the field of HCI, and as Sommerville and Dewsbury (2007) pointed out, it is practically unrealistic to conduct comparative evaluations of any design methods and frameworks. This problem holds for usability evaluation methods and frameworks as well (Ham et al, 2009). Furthermore, it would be very hard to absolutely prove the effectiveness of our framework. But it could be possible to examine the effectiveness of our framework in terms of qualitative and quantitative measures such as the number of identified usability problems. However, the nature of identified usability problems and the linking between the usability problems and their relevant design features it would be more important to assess the value of usability evaluation methods (Molich et al, 2004). The evaluation of the proposed usability framework would be more effective and consistent if it is done by some experts in the field of Computer science and specifically in HCI and usability, and of course with special knowledge and skills needed for investigating the usability of Mashup Makers. For this reason, we conducted an evaluation exercise with some usability practitioners to examine the usefulness of the framework.

7.2 Evaluation: Design and Setup

7.2.1 Design

We prepared a step-by-step method (see table 7.1) for usability practitioners (experts) to guide them using the MUEF framework in a usability evaluation process.

Table 7.1: Step by step evaluation method for evaluating MUEF

Step	Action
1	Preparation for the Mashup creation
2	Performing the Mashup design using the Mashup maker considered
3	Identify Mashups Maker Composition Approach (identify pre-usability data steps: Preface, Operate and Run)
4	Identify task dependent and task independent Usability Data
5	Give ranking for usability metrics by filling the Microsoft excel sheet
6	Identify weight of every usability factor (attribute/criteria/metric) associated
7	Fill-in feedback questionnaire and give oral feedbacks
8	Usability of Mashup maker considered by sort of qualitative and quantitative means

The evaluation process itself was composed of three main phases (see table 7.2): Phase 1 was the planning and preparation phase, where an introduction of the MUEF is presented together with a presentation of the Mahsup Maker to be evaluated. Phase 2 was the actual evaluation conduction, where the usability practitioner has to follow our step-by-step method given in table 7.1 and fill-in the question-list sheets to identify the relevant usability data, give ranking for usability metrics, map those metrics to their relevant usability criteria and finally to map those criteria to their related indicators in the indicators level. We prepared a Microsoft Excel sheet to help evaluators in the quantification of the usability evaluation of Mashup makers using MUEF. All the

materials, sheets and resources are provided in Appendix 5 (as they are the same as for the usability evaluation procedure of MUEF - table 6.4 and in section 6.3 of chapter 6). Finally, Phase 3 was the closing of the evaluation process, where the evaluator was been thanked for his participation in the evaluation and was asked to fill-in a simple questionnaire about his opinion of the usefulness of MUEF followed by an oral feedback discussion.

Table 7.2: Phases of MUEF evaluation process

Phase	Description
1	- Introduction to MUEF and a presentation of the Mashup Maker considered.
2	- Evaluator activity including tasks performance and MUEF evaluation (following the step-by-step method given in figure 7.1)
3	- Closing of the evaluation process with a questionnaire and an oral feedback.

The step-by step-method to be followed by the evaluator was as follow:

- In step 1, the evaluator is prepared for the mashup creation by introducing him to the Mashup Maker considered, the tasks to be performed, and the way to design the Mashup using the considered Mashup Maker.
- In step 2, the evaluator creates the mashup using the Mashup Maker considered.
- In step 3, the evaluator is asked to collect the usability data (pre-usability data in form of the three steps Preface, Operate and Run). As mentioned in table 6.4 (Usability evaluation with MUEF).
- In step 4, the evaluator is requested to identify the task dependent and task independent usability data. As mentioned in table 6.4 (Usability evaluation with MUEF).

- In step 5, the evaluator is requested to give ranking of usability metrics by filling in the Microsoft excel sheet as mentioned in table 6.4 (Usability evaluation with MUEF). (see Appendix 4).
- In step 6, the evaluator is requested to give weight for usability factors considered in MUEF (metrics, criterions and indicators) as mentioned in table 6.4 (Usability evaluation using MUEF). We also asked the experts to give their opinion and comments on the metrics, the criteria and indicators.
- In step 7, the usability evaluator is requested to fill in a questionnaire where he gives comments and feedbacks; also oral feedback is considered in this step.
- In the last step, step 8, the usability evaluation of the Mashup Maker is completed and could be shown qualitative as well as quantitative.

7.2.2 Setup

12 evaluators (experts) were recruited for this evaluation; most of them were HCI professionals, Web user interface design experts or usability practitioners with at least 5 years of working experience in academia and/or industry. Those who have had academic experience have taught a HCI course at least three times and supervised undergraduate final study projects also in the field of Web Information Systems. All of the evaluators had industrial experience of no less than 3 years and have been involved in evaluating the user interfaces of Web sites and Web information systems.

For the evaluation exercise, two Mashup Makers were selected: Yahoo! pipes and Dapper. As already explained, the evaluation and validation process using the framework consisted of three main phases: The first phase was dedicated to the planning and preparation. The second phase was dedicated to the conduction of the evaluation of the selected Mashup Makers using the MUEF framework. The last phase of the evaluation/validation process was to get a comprehensive feedback from evaluators. This was achieved by asking the evaluators to fill in a questionnaire and by means of an oral discussion.

Next, we used the results obtained from the evaluators in diagnosing the usability issues of the framework itself, and for defining area's of improvement. Full materials of this evaluation are provided in appendix 5.

7.3 Results

The results of the evaluation can be presented in two ways: firstly, as a quantitative measurement of the evaluator's opinion on the framework, secondly as a summary of usability issues and recommendations formulated by the evaluators. The main question asked at the end of the evaluation was if they agree that the MUEF is useful to evaluate the usability of Mashup Makers. Table 7.3 shows the evaluator's opinion on the usefulness of MUEF framework. All the participants agreed on the fact that MUEF is a useful tool for evaluating the usability of Mashup Makers. Three strongly agreed, eight agreed and only one neither agreed nor disagreed and nor strongly disagreed.

Table 7.3: Evaluator's opinion on the usefulness of MUEF

	Strongly agree	**Agree**	**Neither agree nor disagree**	**Disagree**	**Strongly disagree**
# of participants	3	8	1	0	0

The usability issues and recommendations received from the participants are summarized in the following points:

- Firstly, some terms and expressions used to evaluate usability, as well as some questions in the question lists to evaluate the metrics, and the criteria themselves are too abstract to be directly measured.
- Secondly, sometimes and in some places it was difficult to give ranking for some activities or items because of different reasons, as well as because of difficulties in understanding the terms used.

- Thirdly, in some cases in the question lists there was a repetition of the terms and expressions for the same evaluated task.

These opinions, observations and findings are very important and we took those points into consideration for the improvement of MUEF. Those issues are presented in the next subsection.

7.4 Enhancements to MUEF

It was noted that the use of the three main areas of usability evaluation factors that were identified in chapter 4 after the user experiment, and which is the basis of the evaluation framework, was useful and facilitates the weighting of usability evaluation factors (especially those presented as usability evaluation criteria and further the usability evaluation metrics in the question list in the assessment level of the MUEF) to be employed by the usability evaluators. However, it was inferred from the qualitative feedback from the evaluators that there is a need to provide more explanation about the 42 metrics and further about the usability evaluation factors used with regard to their importance. This would help the usability practitioners and the Mashup Maker designers prioritize the ranking of the metrics and in general the weighting of evaluation factors for their usability evaluation of the Mashup Maker they are occupied with. This was inferred when the experts were asked to give their opinion on the types of specific metrics to be used in the evaluation and the method that they would prefer to employ. Therefore, it was suggested that the number of usability evaluation factors (metrics, criteria and indicators) identified in the levels of the usability evaluation framework MUEF should be classified according to their severity level (major and minor). For instance, we noted that during the evaluation process, most of the evaluators were suggesting to replace the weighting of the usability criteria by something more easily to score, such as minor and major. Major could be used to indicate that they find the usability criteria more relevant to the usability evaluation of the Mashup Maker, otherwise minor could be used. Further, we also noted that most of the evaluators gave high weights for usability criteria such as visibility, consistency, hard mental operations, error-pronounce, viscosity, secondary notation and closeness of mapping.

To make the suggested enhancements to the framework, the usability evaluation factors that were identified in MUEF were examined and classified by their weight as mentioned above. Herewith, it is worth mentioning that the usability evaluation factors weights, generated by user testing, were obtained by referring to the performance data, the observation notes, the notes generated from reviewing in the literature review and users' comments both from the user experiment and the expert case study, and the final questionnaire.

Based on the previous discussion and results of the evaluation process that related to the number and severity level of each specific usability evaluation factor, especially in the assessment level of MUEF, and further in the usability indication area identified by the proposed conceptual model of chapter 6, a minor change to the framework is suggested. The proposal is to divide the usability evaluation factors into two main groups depending on their severity level, i.e. major or minor.

7.5 Conclusions

In this chapter, we reported on the evaluation and validation of MUEF. For this, we performed an evaluation exercise with 12 experts and we found several interesting points to enhance MUEF. Those points can be summarized as follows:

Firstly, usability quantification (giving scores, weights, etc.) within MUEF is an important but subjective issue and requires some expertise. This needs to be explained to evaluators every time the framework is used; we think it could be an interesting research topic to refine the usability quantification within MUEF. Secondly, as mentioned in section 7.4, usability evaluation factors are better to be grouped depending on their severity (minor or major). Finally, the usability evaluation terms and expressions used in criteria, metrics and questions in the framework's evaluation process should be improved and explained better. The framework should always be updated with new terms used in the field of Mashups and End-user design environments.

CHAPTER 8: CONCLUSIONS

This chapter presents and discusses the conclusions that have been drawn from conducting this research. First, we present the contributions of the work and explain how the aims and objectives of this research have been achieved. Next, we discuss the limitations of this research. Finally, the chapter provides recommendations for future work.

8.1 Contributions and Achievement of the Objectives

8.1.1 Achievement of the Objectives

The aim of this research was to develop a methodological and analytical framework which could comprehensively and effectively be used by usability practitioners and Mashup Maker designers to investigate usability problem areas of Mashup Makers. The development of this framework also wants to raise awareness to usability and usability evaluation methods in the field of HCI, usability and especially usability evaluation of Web applications development for end-users. The main aim was achieved by meeting the three specific objectives formulated for the research. Actually, we performed 9 steps (divided in 3 phases), as specified in the research method of chapter 2. This section summarises how the objectives of this research have been achieved.

Objective 1:

(1) To discover the main issues related to Web Mashup Makers, Web Mashup usability evaluation approaches, and to have a concrete understanding of the usability of Mashup Makers for end-users.

This objective was met by performing a deep literature study about Mashup Makers and usability evaluation of such environments and also by performing a pilot study of a selection of the most well-known Mashup Makers. The literature was investigated to find out what would be the most appropriate approaches that could be used to evaluate the usability of Mashup Makers. For instance, in Chapter 4 we explain the reasons behind the

selection of CD's framework to be used as evaluation criteria, together with their usefulness for Mashup Maker's usability evaluation.

Objective 2:

(2) To deeply investigate usability issues of Mashup Makers for end-users by performing empirical studies (pilot studies and user experiments), and to draw on the findings of the empirical studies in establishing a consolidated usability evaluation model for Mashup Makers for end-users.

This objective was met by performing both a pilot study and a complete user experiment both described in chapter 4 and by the development of the Conceptual Evaluation Model described in chapter 5.

An investigation of 6 of the most well-known Mashup Makers was the first step. In order to evaluate the usability of the 6 selected Mashup Makers, an evaluation method was designed and the author, as evaluator, conducted the tasks in the evaluation. Section 4.2 of Chapter 4 explains the complete evaluation process, the full details are provided in appendix 1.

Next, the empirical study was done by selecting appropriate usability evaluation criteria as well as designing the user experiment. For this user experiment, three Mashup Makers were selected; the tasks were performed, and data was collecting using the identified methods.

Also for this objective, in order to evaluate the usability of the three selected Mashup Makers, research tools were designed and the evaluators (our self) and participants were recruited. Chapter 4 explains the evaluation approach developed to evaluate the usability of the Mashup Makers and the user testing materials created to conduct the user experiment. The procedure that was undertaken to collect findings is explained in Chapter 4 and Appendix 2.

The importance of this step was the identification of usability problems that we categorize into three main areas. The usability problems areas resulted from the user experiment were analysed into more details. Each usability problem area employed a kind of view/aspect of usability. Chapter 5 explains the usability problems areas identified for Mashup Makers and presents them as a Conceptual Evaluation Model for the usability of Mashup Makers.

Objective 3:

(3) To develop a usability evaluation framework for Mashup Makers for end-users which will support usability experts and Mashups Maker's designers evaluating the usability of their Mashup Makers and to validate the framework developed.

To achieve the objective of developing an evaluation framework for the usability of Mashup Makers, we developed MUEF as a multi-layers analytical framework. MUEF consists of different levels of usability factors, where each level is a refinement of the previous level. MUEF is oriented to usability practitioners and Mashup Maker designers. An evaluation exercise of MUEF was also performed by a number of experts in the domain. This evaluation exercise showed that MUEF is a useful framework that can be the basis of the enhancement of end-user development environments of web applications and other end-user development.

8.1.2 Contributions

In this subsection, we summarize the major contributions of this dissertation. The major contributions of the research are presented in three-fold:

- In our opinion, a first contribution of this research is the fact that we were able to uncover and distinguish the importance of Mashup Makers as end-user Web application development environments and to draw the attention to the usability as a key issue to the success of such tools.
- The second major contribution is the development of a conceptual model that could was used for usability evaluation of Mashup makers for end-user and. This

model identifies three major area's of usability: visual support, interaction support, and functional support.

- The final and the most important contribution of this dissertation is the development of the usability evaluation framework for Mashup makers for end-users, MUEF as a multi-layer (hierarchal) methodological framework which targets usability practitioners and Mashup Makers' designers. MUEF is conceived as an easy to use framework with user-friendly materials. It is worth highlighting that MUEF was evaluated by experts in the domain by means of an experimental study.

8.2 Limitation of this Work

As any research work and study, a number of limitations were found while conducting this research; those limitations could have influenced the findings obtained. In the following paragraphs we present those limitations and explain the main issues related to them:

• In the user experiment, the use of the three Mashup Makers considered and the tasks performed could have influenced the results. These Mashup Makers were selected on the basis of their availability, frequent use, and not on the basis of having the largest number of functionality or usability problems. This could have influenced the types of problems discovered, and therefore covered in the framework proposed, and may not be representative for all Mashup Makers.

• The time period used for this research, which was covering the last three years, was moderate when compared to other studies in the field of software development. For instance, while we were preparing our approach for the empirical studies the market of Mashup Makers was considerably changing, i.e. some Mashup Makers changed their approach, while others disappeared. It is very well possible that in the future other developments may occur in the domain of Mashup Makers that could influence our findings.

• The suggested framework was evaluated and tested by a number of experts. An assessment of whether the framework would reduce the time/cost of performing a usability evaluation was not undertaken.

8.3 Recommendation for the Future

In order to address the limitations identified in section 8.2, a number of recommendations are suggested for future work.

1. Further research on user sampling the empirical study could be undertaken to investigate the relationship between the sample size of the users, evaluators, experts and the number of problems identified by them. The same for the relationship between the number of tasks requested to be performed by the evaluators and the number of usability problems identified by them. The results of such investigations could then be compared to the current results to decide whether the number of tasks and/or evaluators required can be reduced while identifying the same number of usability problems.
2. Research could be undertaken regarding the effect of learning on the usability of Mashup Makers. It is indeed possible that the usability of Mashup Makers will be perceived differently when the end-users could spend some time on learning or practice with the tools. It is also possible that in that case other types of usability issues popup that are currently not considered in the framework. This could require re-considering the areas of usability problems derived and re-inspecting the framework's efficiency and usefulness by other evaluators (experts).
3. Research could be undertaken for new Mashup Makers as well as for the Mashup Makers which have changed their user interfaces or their interaction based on the recommendations offered by this research. The research could involve a comparison between the Web usability metrics and criteria values obtained before and after the evaluation of the Mashup Maker. This comparison could reveal best ways of end-user development of Web applications.

4. The developed conceptual model could be used as the base for usability evaluation of different end-user products and software systems. Further research could also be undertaken to investigate if the conceptual model indeed justifies this hypothesis.
5. Finally, the proposed framework needs to be updated over time to follow new developments in the field of Mashup making as well as in user interfaces and usability in general.

REFERENCES

Abran, Khelifi, and Suryn, (2003). Usability meanings and interpretations in ISO standards, Software Quality Journal, 325-333.

Al Sarraj. W., and De Troyer, O., (2009). Usability Framework for Web Mashup Makers for Casual User, Proceedings of the ICWE 2009, Publ. CEUR-WS, ISBN 1613-0073, Spain.

Al Sarraj. W., and De Troyer, O., (2010). Web Mashup Makers for Casual Users: A User Experiment, Proceeding of the (iiWAS2010), ISBN 978-1-4503-0421-4, Paris, France.

Ankolekar, A., et al., (2007). The Two Cultures: Mashing up Web 2.0 and the Semantic Web, Proc. Int'l Conf. World Wide Web, ACM Press, , pp. 825–834.

Apatar, [n.d.] <http://www.apatar.com>, [accessed 22.09.2009].

Avison, D.E., and Fitzgerald, G., (2003). Information Systems Development: Methodologies, Techniques and Tools, third ed. Blackwell Scientific.

Balasubramaniam, S.; Lewis, G.A.; Simanta, S.; and Smith, D.B.; (2008). Situated Software: Concepts, Motivation, Technology, and the Future, Published by Software, IEEE , Nov.-Dec. 2008, Volume: 25 Issue: 6, on page(s): 50 – 55, ISSN: 0740-7459

Bastick, T., (2003). Intuition: evaluating the construct and its impact on creative thinking. Kinsgton, Jamaica: Stoneman & Lang.

Bautsch, H., Granger, J., Karnjate, T., Kahn, F., Leveston, Z., Niehus, G., Ward, T., (2001). An Investigation of Mobile Phone Use: A Socio-Technical Approach. Socio-Technical Systems in Industry, Madison.

Beemer, B., A., and Gregg, D., (2009). Mashups: A Literature Review and Classification Framework. Future Internet 1(1): 59-87.

Beletski, O., (2008). End User Mashup Programming Environment. Available at: http://www.tml.tkk.fi/Opinnot/T111.5550/2008/End%20User%20Mashup%20Programming%20Environments_p.pdf

Bertin, J. (1993). Semiology of Graphics: Diagrams, Networks, Map Univ. of Wisconsin Press.

Bergson, H., (2002). The creative mind: An introduction to metaphysics. New York: Kensington.

Berners-Lee, T. Originator of the Web and director of the World Wide Web Consortium talks about where we've come, and about the challenges and opportunities ahead. DeveloperWorks Interviews, Available at: http://www128.ibm.com/developerworks/podcast/dwi/cmint082206.txt, [accessed 16.12.2009].

Bevan, N., (1999). Quality in use: Meeting user needs for quality. The Journal of Systems and Software 49, 89-96.

Bidgoli, H., (2004). The Internet encyclopaedia, John Wiley and Sons.

Blackwell, A. F., Britton, C., Cox, A., Dautenhahn, K., Green, T. R. G., Gurr, C., Jones, S., Kadoda, G., Kutar, M. S., Loomes, M., Nehaniv, C. L., Petre, M., Roast, C., Roe, C., Russ, S., A., W., & Young, R. M., (2001). Cognitive Dimensions of Notations: Design Tools for Cognitive Technology. In M. Benyon & C. L. Nehaniv & K. Dautenhahn (Eds.), Cognitive Technology (pp. 325-341), Springer-Verlag.

Blackwell, A., & Green, T., (2000). A cognitive dimensions Questionnaire Optimized for Users, 12th Workshop of the Psychology of Programming Interest Group, Italy.

Blackwell, A., & Green, T., (2002). Notational Systems – the cognitive Dimensions of Notations Famework, HCI Models, Theories and Frameworks.

Blandford, A. E., Hyde, J. K., Connell, I., Green, T. R. G., (2004). Scoping analytical usability evaluation methods: a case study. Technical report of University College Interaction Centre (<http://www.uclic.ucl.ac.uk/annb/CASSM>).

Boott, R., Haklay, M., Heppell, K., and Morley, J., 2001. The use of GIS in brownfield redevelopment. In Innovations in GIS 8: Spatial Information and the Environment, edited by P. Halls (London: Taylor and Francis), pp. 241–258.

BORING, R. L. Human-computer interaction as cognitive science. In: ANNUAL MEETING OF THE HUMAN FACTORS AND ERGONOMICS SOCIETY, 46, Baltimore, 2002. Proceedings. MD, 2002.

Brancheau, J.C., Brown, C.V., (1993). The Management of End-User Computing: Status and Directions. ACM Computing Surveys, 25(4).

Brinck, T., Gergle, D. & Wood S.D., (2001). Usability for the web: designing websites that work, Morgan Kaufmann Publishers.

Bryman, A. (2008). Social research methods. Oxford University Press.

Burmeister, B., Arnold, M., Copaciu, F., Rimassa, G., (2008). BDI-Agents for Agile Goal-Oriented Business Processes, Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008) – Industry and Applications Track, Berger, Burg, Nishiyama (eds.), May, 12-16., 2008, Estoril, Portugal, pp. 37-44.

Cappiello, C., Daniel, F., Matera, M., Picozzi, M., and Michael Weiss. (2011). Enabling End User Development through Mashups: Requirements, Abstractions and Innovation Toolkits. Proceedings of IS-EUD 2011, June 2011, Pages 9-24.

Card, S.K., Moran, T.P., & Newell, A. (1980). The keystroke-level model for user performance time with interactive systems, Communications of the ACM, 23(7), 396-410.

Carliner, S., (2003). Physical, cognitive, and affective: A three-part framework for information design. In: Albers, M., Mazur, B. (Eds.), New Jersey; ISBN 0-8058-4140-7.

Castro, J., M. Kolp, J. Mylopoulos, (2001). A requirements-driven development methodology, in: Proceedings of the 13th International Conference on Advanced Information Systems Engineering, CAiSE'01, pp. 108–123, Interlaken, Switzerland.

Chung, Lawrence, and Julio Cesar Sampaio do Prado Leite (2009). On Non-Functional Requirements in Software Engineering, Lecture Notes in Computer Science, Volume 5600/2009, 363-379, DOI: 10.1007/978-3-642-02463-4_19.

Cockburn, A., (1997). Structuring Use Cases with Goals" (Part 1), J. Object-Oriented Programming, Sept.–Oct., pp. 35–40; available at: http://members.aol.com/acockburn/papers/usecases.htm.

Cockburn, A., (1997). Structuring Use Cases with Goals," (Part 2), J. Object-Oriented Programming, Nov.–Dec., pp. 56–62; available at: http://members.aol.com/acockburn/papers/usecases.htm.

Cockburn, A., (2001). Writing Effective Use Cases, Addison-Wesley, Boston, Mass.

Collis, J. and Hussey, R., (2003). Business research: a practical guide to undergraduate and postgraduate students, 2nd ed. Palgrave Macmillan.

COSTABILE M.F., FOGLI D., MUSSIO P., AND PICCINNO A. (2006). End-user development: The software shaping workshop approach. In End-User Development, H. Lieberman, F. Paterno, and V. Wulf (eds.) Springer, 183-205.

Cost-Effective User Centred Design, (2001). User centred design standards, <http://www.usabilitynet.org/trump/resources/standards.htm>, [accessed 10.04.09].

Creswell, J., (1994). Research design: qualitative and quantitative approaches, Sage Publications.

Creswell, J., 2003. Research design: qualitative, quantitative, and mixed method approaches. SAGE.

Daniel, F., Koschmider, A., Nestler, T., Roy, M., and Namoun, A., (2010). Toward process mashups: key ingredients and open research challenges, Proceedings of the 3rd and 4th International Workshop on Web APIs and Services Mashups, ACM New York, NY, USA.

Dapper [n.d.], <http://www.dapper.net>, [accessed 30.09.2009].

Dapper – Yahoo! < http://advertising.yahoo.com/article/dapper.html >, [accessed 30.12.2011].

De Angeli, A., Battocchi, A., Chowdhury, S. R., Rodriguez, C., Daniel, F., Casati, F., (2011). End-User Requirements for Wisdom-Aware EUD" Proceedings of IS-EUD2011.

Di Lorenzo, G., Hacid, H., Paik, H., Benatallah, B., (2008). Mashups for Data Integration: An Analysis, Technical Report at UNSW-CSE-TR-0810, available at: ftp://ftp.cse.unsw.edu.au/pub/doc/papers/UNSW/0810.pdf.

De Marco, T. (1978). Structured Analysis and System Specification, Yourdon Press.

Dillon, A., (1999). TIME – a multi-levelled framework for evaluating and designing digital libraries. International Journal of Digital Libraries 2 (2–3), 170–177.

Dillon, A., (2001). Beyond usability: process, outcome and affect in human–computer interactions. Canadian Journal of Information Science 26 (4), 57–69.

Donyaee, M, Seffah, A., Kline, R., and Striva, J. (2002). An integrated framework for usability measurement. In Proc. 12th International Conference on Software Quality 1-10.

Duan, Y., and Cruz C., (2011), Formalizing Semantic of Natural Language through Conceptualization from Existence. International Journal of Innovation, Management and Technology (2011) 2 (1), pp. 37-42.

Dumas, J.S., & Redish, J.C., (1994). A practical guide to usability testing, New Jersey: Ablex Publishing.

Dumas, J. S. & Redish, J. C., (1999). A practical guide to usability testing. Second Intellect Ltd, Rev Sub edition.

Dix, A., Finlay, J.E., Abowd, G.D., Beale, R., (2003). Human-Computer Interaction 3rd edn. Prentice-Hall, Inc., Upper Saddle River.

Easterby-Smith, M., Thorpe, R. and Lowe, A., (1991). Management research: an introduction, Sage Publications.

Ennals, R., Gay, D., (2007). User-Friendly Functional Programming for Web Mashups, ICFP' available at: http://portal.acm.org/citation.cfm?id=1291187.

Fichter, D., (2004). Heuristic and cognitive walk-through evaluations, Online, 28(3).

Flashwidgetz [n.d.] < http://www.flashwidgetz.com/> [accessed 30.12.2011].

Frøkjaeer, E., Hertzum, M., Hornb_ek, K., (2000). Measuring usability: are effectiveness, efficiency, and satisfaction really correlated? In: Proceedings of the ACM CHI 2000. Conference on Human Factors in Computing Systems, The Hague, The Netherlands, pp. 345–352.

Folmer, E., and Bosch, J., (2004). Architecting for Usability: a Survey. Journal of Systems and Software, 70(1-2):61–78,

Folmer, E., Van Gurp, J., and Bosch, J. (2003). A framework for capturing the relationship between usability and software architecture. Software Process Improvement and Practice 8, 2, 67-87.

Freitas, C. Luzzardi, P. Cava R., Pimenta, M. Winckler, A. and Nedel, L. (2002). "Evaluating Usability of Information Visualization Techniques," Proc. Advanced Visual Interfaces Conf. (AVI '02), pp. 373-374.

Goolkasian, P., (2000). Pictures, Word and Sounds: From Which Format Are We Best Able to reason?" J. General Psychology, vol. 127, no. 4 pp. 439-459.

Gurr, C.A., (1999). Effective Diagrammatic Communication: Syntactic, Semantic and Pragmatic Issues," J. Visual Languages and Computing, vol. 10, pp. 317-342.

Goodwin, S., (2005). Using screen capture software for website usability and redesign buy-in, Library Hi Tech, 23(4).

Google Gadget [n.d.] < http://www.google.com/webmasters/gadgets/> [accessed 30.12.2011].

Google Mashup Editor [n.d.], <http://code.google.com>, [accessed 15.10.2008].

Gray, W. & Salzman, C., (1998). Damaged merchandise. A review of experiments that compare usability evaluation methods, Human-Computer Interaction, 13, 203-261.

Grammel, L., and Storey, M., (2008). An end-user perspective on Mashup Makers, Available at http://lars.grammel.googlepages.com/paper_Mashup_makers.pdf,

Grammel, L., and Storey, M., (2010). A Survey of Mashup Development Environments. Springer LS in CS, Volume 6400/2010, 137-151, DOI: 10.1007/978-3-642-16599-3_10.

Green, T.R.G. and M. Petre, (1996). Usability analysis of visual programming environments: a 'cognitive dimensions' approach. Journal of Visual Languages and Computing, 7(2): p. 131-174.. <http://www.cl.cam.ac.uk/~afb21> (retrieved 30.12.2008).

Gummesson, E. (2000). Qualitative Methods in Management Research, 2nd ed., London: Sage Publications.

Gwardak, L., Påhlstorp, L., (2007). Exploring Usability Guidelines for Rich Internet Applications, Master thesis Lund University, available at: http://biblioteket.ehl.lu.se/olle/papers/0002774.pdf.

Haklay, M., and Harrison, C. M., 2002. The potential for public participation GIS in UK environmental planning: appraisals by active publics, International Journal of Environmental Planning and Management (JEPM), 45, 841–863.

Ham, D.-H., Heo, J., Fossick, P., Wong, W., Park, S., Song, C., , Bradley, M., (2006). Conceptual Framework and Models for Identifying and Organizing Usability Impact Factors of Mobile Phones", Published by ACM OZCHI 2006, , Sydney, Australia.

Hammersley, M. (1996). The Relationship between Qualitative and Quantitative Research: Paradigm Loyalty versus Methodological Eclectism. In: Richardson, J.T.E. eds. Handbook of Research Method in Psychology and the Social Science. Leicester: BPS Books.

Hartson, H.R., Andre, T.S., and Williges, R.C., (2003). Criteria for evaluating usability evaluation methods. Human-Computer interaction, 15(1), pp. 145-181.

Hartson, H.R., Andre, T.S., and Williges, R.C., (2003). Criteria for evaluating usability evaluation methods. International Journal of Human–Computer Interaction 15 (1), 145–181.

Hasan, L., (2009). PhD thesis: Usability Evaluation Framework for E-commerce Websites in Developing Countries, Loughborough University, UK, available at: https://dspace.lboro.ac.uk/.

Hasan L., Morris, A., Probets, S., (2011). A comparison of usability evaluation methods for evaluating e-commerce websites, DOI:10.1080/0144929X.2011.596996, The online platform for Taylor & Francis Group content.

Hertzum, M., Jacobsen, N.E., (2001). The evaluator effect: a chilling fact about usability evaluation methods. International Journal of Human–Computer Interaction 13 (4), 421–443.

Hix, D., Hartson, H.R., (1993). Developing User Interfaces: Ensuring Usability through Product and Process. John Wiley & Sons, New York, USA.

Hollingsed, T. & Novick, D. (2007). Usability inspection methods after 15 years of research and practice" In SIGDOC '07: Proceedings of the 25th annual ACM international conference on design of communication, 249-255, New York, NY, USA.

Hornbaek, K., (2006). Current practice in measuring usability: challenges to usability study and research. International Journal of Human-Computer Studies 64, 79-102. 2.

Holzinger, A., (2005). Usability engineering methods for software developers, Communications of the ACM, 48(1).

Hussey, J. and Hussey, R. (1997). Business Research: A Practical Guide for Undergraduate and Postgraduate Students, Macmillan Press, London.

Huynh, D., Miller, R., Karger, D., (2007). Potluck: Data Mash-up Tool for Casual Users, ISWC'available at: http://people.csail.mit.edu/dfhuynh/research/papers/iswc2007-potluck.pdf

IEEE std. 1064, (1998). Software Quality Metrics Methodology.

IBM Mashup center, [n.d.] <http://services.alphaworks.ibm.com/graduated/Mashupcenter.html> [accessed 12.12.2008].

iCalendar – Google, [n.d.] < http://support.google.com/calendar/bin/answer.py?hl=en&answer=37108>, [accessed 30.12.2011].

Intel MashMaker [n.d.] <http://Mashmaker.intel.com> [accessed 25.02.2009]

Internet world stats [n.d.] <http://www.internetworldstats.com/stats.htm> [accessed 10.11.2011]

ISO 9241 Part 11, (1998). International Standard for Ergonomic requirements for office work with visual display terminals (VDTs) – Guidance on Usability, available at:

http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16883,

JackBe, [n.d.] <http://www.jackbe.com> [accessed 14.12.2008].

Jacko, J.A., Sears, A., (2003). The Handbook for Human Computer Interaction. Lawrence Erlbaum & Associates, Mahwah.

Jhingran, A., (2006). Enterprise Information Mashups: Integrating Information, Simply. In: 32nd International Conference on Very Large Data Bases, 3--4, VLDB Endowment.

Kadyte, V., Tétard, F., (2004). The role of usability evaluation and usability testing techniques in the development of a mobile system, NordiCHI 2004, Aalborg University, Denmark.

Kaltenbacher, B., (2008). PhD thesis: Intuitive Interaction - Steps towards an integral understanding of the user experience in interaction design.

Kangas, E., Kinnunen, T., (2005). User-centred design to mobile application development. Communication of the ACM 48 (7), 55–59.

Karat, J., (1997). User-centred software evaluation methodologies. In: Helander, M.G., Landauer, T.K., Prabhu, P.V. (Eds.), Handbook of Human–Computer Interaction. Elsevier Science, The Netherlands.

Karwowski W, Soares M M, Stanton, N A. (2011) Human Factors and Ergonomics in Consumer Product Design: Methods and Techniques (Handbook of Human Factors in Consumer Product Design): Needs Analysis: Or, How Do You Capture, Represent, and Validate User Requirements in a Formal Manner/Notation before Design" (Chapter 26 by K Tara Smith) , CRC Press.

Kaushik, A., (2007). Web analytics, an hour a day, Wiley Publishing, Inc.

Klockar, T., Carr, D.A., Hedman, A., Johansson, T., Bengtsson, F., (2003). Usability of mobile phones. In: Proceedings of the 19th International Symposium on Human Factors in Telecommunication, Berlin, Germany, pp. 197–204.

Koschmider, A., Hoyer, V. Giessmann, A., (2010). Quality Metrics for Mashups" SAICSIT'10 Bela Bela, South Africa, ACM 978-60558-950-3.

Kosslyn, S.M., (1985). Graphics and Human Information Processing, J. Am Statistical Assoc., vol. 80, no. 391, pp. 499-512.

Koyani, S., Bailey, R. & Nall, J., (2003). Research-based web design & usability guidelines, [Retrieved 01.08. 2011, from http://usability.gov/pdfs/guidelines_book.pdf.

Kushniruk, A., Patel, V.L., Cimino, J.J., Barrows, R., (1996). Cognitive evaluation of the user interface and vocabulary of an outpatient information system, in: Proceedings of the AMIA Annual Fall Symposium, vol. 1996, pp. 22—26.

Kulathuramaiyer, N., (2007). Mashups: Emerging Application Development Paradigm for a Digital Journal. In: Journal of Universal Computer Science 13(4), 531—542.

Kwahk, J., Han, S.H., (2002). A methodology for evaluating the usability of audiovisual consumer electronic products. Applied Ergonomics 33 (5), 419–431.

Larkin J.H., and Simon, H.A., (1987). Why a Diagram Is (Sometimes) worth Ten Thousand Words, Cognitive Science, vol. 11. no.1, pp.65-100.

Lausen, S., (2003). Task descriptions as functional requirements, IEEE software, -computer.org.

Lazar, J., (2006). Web usability: a user-centered design approach, Pearson/Addison-Wesley.

Lee, Y.S., Hong, S.W., Smith-Jackson, T.L., Nussbaum, M.A., Tomioka, K., (2006). Systematic evaluation methodology for cell phone user interfaces. Interacting with Computers 18 (2), 304–325.

Lewis, C., Polson, P., Wharton, C., & Rieman, J., (1990). Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces", Proceedings of the Conference on Human Factors in Computing Systems, 235-242.

Lieberman, H., et al., (2006). End-User Development: An Emerging Paradigm, Pub: Springer available at: http://www.springerlink.com/content/h371591g75621w53/fulltext.pdf.

Lew, P., Zhang, L., and Wang, S., (2009). Model and Measurement for Web Application Usability from an End User Perspective, ICWE2009 – QAW, Eds., Spain,

Lidwell, W., K. Holden, and J. Bulter, (2003). Universal Principles of Design: A Cross-Disciplinary, Reference, Rockport Publishers.

Love, S., (2005). Understanding Mobile Human–Computer Interaction. Elsevier Ltd., Oxford, UK.

Liu, X., Hui, Y., Sun, W., Liang, H., (2007). Towards Service Composition Based on Mashup, Published at IEEE Congress on services, ISBN: 978-0-7695-2926-4, Salt lake City, UT.

Macro, O., (2000). Usability literature review, Report submitted to Centers for Disease Control and Prevention Epidemiology Program Office, Atlanta, Georgia.

Mayer R.E., and Moreno, R., (2003). Nine Ways to Reduce Cognitive Load in Multimedia Learning," Educational Psychologist, vol. 38, no. 1, pp. 43-52.

Merrill, D., (2006). Mashups: The new breed of Web app--An introduction to mashups, available at: http://www.webremix.org/readings/merrill.pdf

Mehandjiev, N., Lécué, F., Wajid, U., and Namoun, A., (2010). Assisted Service Composition for End Users, 2010 Eighth IEEE European Conference on Web Services, ISBN: 978-0-7695-4310-9.

Mehandjiev, N., Lécué, F., Wajid, U., Kleanthous, G., and Namoun, A., (2010). Service Compositions for All, Lecture Notes in Computer Science, 2010, Volume 6481/2010, 197-198, DOI: 10.1007/978-3-642-17694-4_21.

Mehandjiev, N., De Angeli, A., Wajid, U., Namoun, A., and Battocchi, A., (2011). Empowering End-Users to Develop Service-Based Applications, Lecture Notes in Computer Science, 2011, Volume 6654/2011, 413-418, DOI: 10.1007/978-3-642-21530-8_53.

Microsoft Popfly, [n.d.] <http://www.popfly.com> [accessed 16.12.2008].

Mohs, C., Hurtienne, J., Israel, J. H., Naumann, A., Kindsmüller, M. C., Meyer, H., A., & Pohlmeyer, A. (a). (2006). IUUI - Intuitive Use of User Interfaces. In: T. Bosenick, M. Hassenzahl, M. Müller-Prove & M. Peissner (eds.), Usability Professionals, 130-133. Stuttgart: German Chapter der Usability Professionals' Association.

Molich, R., Ede, M.R., Kaasgaard, K., Karyukin, B., (2004). Comparative usability evaluation. Behaviour & Information Technology 23 (1), 65–74.

Moody, Daniel, L., (2009). The “Physics” of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering, Published by the IEEE Computer Society.

Namoun, A., Wajid, U., and Mehandjiev, N., (2010a). A Comparative Study: Service-Based Application Development by Ordinary End Users and IT Professionals, Lecture Notes in Computer Science, 2010, Volume 6481/2010, 163-174, DOI: 10.1007/978-3-642-17694-4_14.

Namoun, A., Nestler, T., and De Angeli, A., (2010b). Conceptual and Usability Issues in the Composable Web of Software Services, Lecture Notes in Computer Science, 2010, Volume 6385/2010, 396-407, DOI: 10.1007/978-3-642-16985-4_35.

Namoun, A., Nestler, T., and De Angeli, A., (2010c). Service Composition for Non-programmers: Prospects, Problems, and Design Recommendations, 2010 Eighth IEEE European Conference on Web Services, ISBN: 978-0-7695-4310-9.

Namoun, A., Wajid, U., and Mehandjiev, N., (2010d). Service Composition for Everyone: A Study of Risks and Benefits, Lecture Notes in Computer Science, 2010, Volume 6275/2010, 550-559, DOI: 10.1007/978-3-642-16132-2_52.

Näslund, D. (2002). Logistics needs qualitative research – especially action research. International Journal of Physical Distribution & Logistics Management. 32(5), pp. 321-338.

Netvibes [n.d.] < http://eco.netvibes.com/widgets> [accessed 30.12.2011].

Nestler, T., Namoun, A., Schill, A., (2011). End-user development of service-based interactive web applications at the presentation layer, Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems, ISBN: 978-1-4503-0670-6, ACM New York, NY, USA.

Nielsen, J., (1993a). Usability Engineering. Academic Press, Oxford, UK.

Nielsen, J. (1993b). Heuristic evaluation, In J. Nielsen & R.I. Mack (Eds.), Usability Inspection Methods. (pp. 25-62). New York: John Wiley & Sons.

Nielsen, J. & Mack, R. L. (1994). Usability inspection methods, (Eds.) John Wiley & Sons, New York.

Nielsen J. & Molich, R., (1990). Heuristic evaluation of user interfaces, CHI'90 Proceedings, ACM, 249-256.

Nielsen, J. (1992) Finding usability problems through heuristic evaluation. In Proceedings of the SIGCHI conference on Human Factors and Computer Systems, pages 373–380, ISBN:0 89791-513-5

Nielsen, J., (1994). Heuristic evaluation. In Nielsen, J. & Mack, R. L. (Eds.), Usability Inspection Methods, John Wiley & Sons, New York, 25-64.

Nielsen, J., (2000). Designing web usa'bility: the practice of simplicity, New Riders Publishing.

Nielsen, J., (1993c). Usability engineering, London: Aademic Press.

Nielsen, J., (2003). Usability 101: Introduction to usability. Useit.com. <http://www.useit.com/alertbox/20030825.html>, [accessed 14.02.2009].

Nicholas W., (2006). Social research methods. SAGE.

Olsen, W. K. 2004. Triangulation in social research: qualitative and quantitative methods can really be mixed, in Holborn, M. (ed.), Developments in Sociology: An Annual Review, Ormskirk, Lancs, UK, Causeway Press.

OpenKapow, [n.d.] <http://openkapow.com/Default.aspx> [accessed 02.03.2009].

Open-Mashups Studio [n.d.] <http://www.open-Mashups.org> [accessed 12.10.2009]

O'Reilly, T. What-is-web-2.0. Available at: http://oreilly.com/web2/archive/what-is-web-20.html, [accessed 15.09.2009].

Ort, E., Brydon, S., and Basler, M., (2007). Mashup Styles, available at: http://java.sun.com/developer/technicalArticles/J2EE/Mashup_1/

Pantazi, S., Kushniruk, A., R. Moehr, J., (2006). The usability axiom of medical information systems" International Journal of Medical Informatics Volume 75, Issue 12, December 2006, Pages 829-839 available at: www.intl.elsevierhealth.com/journals/ijmi.

Paul, R.A.; Kunii, T.L.; Shinagawa, Y.; Khan, M.F.; (1999). Software metrics knowledge and databases for project management. appears in: Knowledge and Data Engineering, IEEE Transactions on Issue Date: Jan/Feb 1999, Volume: 11 Issue:1, page(s): 255 – 264, ISSN: 1041-4347

Pearrow, M., (2000). Website usability handbook, Charles River Media.

PICOZZI, M., SPREGA, G., MATERA, M., CAPPIELLO, C., (2010). Master thesis: DashMash: a Mashup Environment for End-User Development, POLITECNICO DI MILANO, Italy.

Pinker, S., (1984). Visual cognition: An Introduction, MIT.

Programmable Web, [n.d.] <http://www.programmableweb.com> [accessed 25.05.2011].

Preece, J., Roger, Y., Sharp, H., Benyon, D, Holland, S., Cary, T., (1994). Human-computer Interaction Addison-Wisley Publishing company.

Preece, J., Rogers, Y., Sharp, H., (2002). Interaction Design: Beyond Human–Computer Interaction, John Wiley & Sons. New Jersey, USA.

Raskin, Jef (1994): Holes in History. In Interactions, 1 (3) pp. 11-16.

Research validity - [n.d.], http://linguistics.byu.edu/faculty/henrichsenl/ResearchMethods/RM_2_18.html, [accessed 15.07.2011].

Research reliability – [n.d], http://krpb.pbworks.com/f/RELIABILITY%26VALIDITY.pdf, [accessed 15.07.2011].

Riehle, Dirk (2000), Framework Design: A Role Modeling Approach, Swiss Federal Institute of Technology.

Rochford, M., (2009). “Talk: Understanding contexts of use” at Interaction2009, Vancouver 8 February 2009. available at: http://www.slideshare.net/rochford/understanding-contexts-of-use [accessed 20.11.2011].

Rubin, J., (1994). Handbook of usability testing: how to plan, design, and conduct effective tests, Wiley.

Saunders, M., Lewis, P., and Thornhill, A., (2007a). Understanding research philosophies and approaches, Book: Research Methods for Business, 5th Edition, Published by: Pearson Education Limited, ISBN: 9780273701484.

Saunders, M., Lewis, P. and Thornhill, A. (2007b). Research Method for Business Students. 4th ed. London: Prentice Hall.

Scholtz, J., (2006). Beyond Usability: Evaluation Aspects of Visual Analytic Environments, IEEE Symposium on Visual Analytics Science and Technology 2006 IEEE Symposium on Visual Analytics Science and Technology, October 31 - November 2, 2006 Baltimore, MD, USA, ISBN: 1-4244-0592-0.

Seffah, A., Donyaee, et., al., (2006). Usability measurement and metrics: A consolidated model”, Software Quality Journal, Vol. 14 (2), pp. 159-178.

Seliger, H. & Shohamy, E. 1989. Second Language Research Methods. Oxford University Press.

Seliger, H. & Shohamy, E. 1995. “Language Testing: Matching Assessment Procedures with Language Knowledge”. 142-160 in Birenbaum M. & Dochy, F., (eds.) Alternatives in Assessment of Achievements, Learning Processes and Prior Knowledge. Kluwer Academic Publishing, Boston, MA.

Serena, [n.d.] <http://www.serena.com> [accessed 25.01.2009].

ServFace project: Service Annotations for user interface composition, [n.d.] <http://www.servface.eu/> [accessed 06.011.2010]

Sharp, H., Rogers, Y. & Preece, J., (2007). Interaction Design: Beyond Human-Computer Interaction, Wiley; Second Edition.

Simon Peyton Jones, (2003). Wearing the hair shirt: a retrospective on Haskell (invited talk). In ACM SIGPLAN Conference on Principles of Programming Languages (POPL'03),

Sommerville, I., Dewsbury, G., (2007). Dependable domestic systems design: A sociotechnical approach. Interacting with Computers 19 (4), 438–456.

Soren Lauesen, (2003). Task Description as Functional Requirements, IEEE Software, Published by the IEEE Computer Society ISBN 0740-7459/03.

Spencer, R., (2000). The streamlined cognitive walkthrough method, working around social constraints encountered in a software development company, CHI 2000 Proceedings, 2(1), 353–359.

Stone, D., Jarrett, C., Woodroffe, M. & Minocha S., (2005). User interface design and evaluation, The Open University, Morgan Kaufmann.

Terziyan, V., (2007). Talk: Semantic and Agent Technologies in Developing Distributed Applications, IT2007, Jyvaskyla, Finland, 01.11.2007, Available at: www.cs.jyu.fi/ai/Terziyan_Klymova.ppt.

Theofanos, M., and Scholtz, J., (2005). A Diner's Guide to Evaluating a Framework for Ubiquitous Computing Applications, HCI International 2005, Las Vegas, NV July.

Thomas, P., and Macredie, R. D., 2002, Introduction to the new usability. ACM Transactions on Computer-Human Interaction, 9, 69–73.

Treu, S., (1994). User Interface Evaluation: A Structured Approach. Plenum Press, New York, USA.

Tufte, E.R., (2001). The Visual Display of Quantitative Information, second ed. Graphics Press.

Usability.gov, U.S. Department of Health and Human Services [n.d.] <http://www.usability.gov>, [accessed 15.10.2008].

Van Lamsweerde, A., and E. Letier, (2000). Handling Obstacles in Goal-Oriented Requirements Engineering, IEEE Transactions on Software Engineering vol. 26(10), pp. 978-1005.

Van den Haak, M.J., de Jong, M.D.T & Schellens, P.J., (2004). Employing thinkaloud protocols and constructive interaction to test the usability of online library catalogues: a methodological comparison, Interacting with Computers, 16 (2004) 1153–1170.

Van den Haak, M.J. & de Jong, M.D.T., (2005). Analyzing the interaction between facilitator and participants in two variants of the think-aloud method, IEEE International Professional Communication Conference Proceedings.

Vicente, K.J. (1999). Cognitive Work Analysis. Lawrence Erlbaum Associates.

Voss, C., Tsikritktsis, N. and Frohlich, M. (2002). Case research in operations management. International Journal of Operations and Production Management 22(2), pp.195-219.

Wajid, U., Namoun, A., and Mehandjiev, N., (2011). Alternative Representations for End User Composition of Service-Based Systems, Lecture Notes in Computer Science, 2011, Volume 6654/2011, 53-66, DOI: 10.1007/978-3-642-21530-8_6.

Wajid. U., Namoune A., and Mehandjiev N., (2010). A comparison of three service composition approaches for end users¨ Proceedings of the International Conference on Advanced Visual Interfaces, ACM New York, NY, USA, ISBN: 978-1-4503-0076-6.

Wharton, C., Rieman, J., Lewis, C. & Polson, P., (1994). The cognitive walkthrough method: a practitioner's guide, In Nielsen, J., and Mack, R. L. (Eds.), Usability inspection methods, 105-140. New York, NY: John Wiley & Sons.

Wikipedia; Likert_scale, [n.d.] <http://en.wikipedia.org/wiki/Likert_scale> [accessed 22.02.2011].

Wikipedia, Web Mashup, [n.d.] http://en.wikipedia.org [accessed 25.11.2008].

Winn, W.D., (1990). Encoding and Retrieval of Information in Maps and Diagrams, IEEE Trans. Professional Comm., vol. 33, no. 3, pp. 103-107.

World Wide Web Consortium, [n.d.] <http://www.w3c.org.> [accessed 28.10.2008].

World Wide Web Consortium - XML [n.d.] <http://www.w3c.org.> [accessed 28.10.2008]

World Wide Web Consortium – HTML [n.d.] <http://www.w3c.org.> [accessed 28.10.2008]

World Wide Web Consortium - Javascript [n.d.] <http://www.w3c.org.> [accessed 28.10.2008]

World Wide Web Consortium – CSS [n.d.] <http://www.w3c.org.> [accessed 28.10.2008]

World Wide Web Consortium – RSS [n.d.] <http://www.w3c.org.> [accessed 28.10.2008]

World Wide Web Consortium – JSON [n.d.] <http://www.w3c.org.> [accessed 28.10.2008]

Wong, J., and Hong, J., I., (2007). Making Mashups with marmite: towards end-user programming for the web. In CHI ’07: Proc. of the SIGCHI pages 1435 -1444, NY, USA, ACM.

Wong, J., Hong, J., (2008). Patterns in Mashups, WEUSE IV ACM 1-58113-000-0/00/0004.

Wuensch, Karl L. (2005). "What is a Likert Scale? and How Do You Pronounce 'Likert?'". East Carolina University. Retrieved March 30, 2011

Yahoo Pipes, [n.d.] <http://pipes.yahoo.com> [accessed 25.10.2008].

Yue, KB., (2010), Experience on Mashup Development with End User Programming Environment , Published at Journal of Information Systems Education, Vol. 21(1).

Zang, N. and Rosson M. B., (2008). What's in a Mashup? And why? Studying the perceptions of web-active end users, In VL/HCC.

Zang, N. Rosson,M. B., and Nasser,V. (2008). "Mashups: who? what? why?," in proceeding of international conference of Computer-Human Interaction CHI2008.

Zang, N. and Rosson, M. B., (2009). Playing with Information: How End Users Think About and Integrate Dynamic Data, IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC).

Zhang, D., Adipat, B.,. (2005). Challenges, methodologies, and issues in the usability testing of mobile applications. International Journal of HCI 18 (3), 293–308.

Zhang, Z., 2003. Overview of usability evaluation methods.
<http://www.usabilityhome.com> (retrieved 30.12.2011).

Zillner, T., (2007). Mashing It Up, A talk at Annual WAAL Conference, available at: http://www.wils.wisc.edu/events/waal07/Mashing_It_Up.webbed.ppt

Zimmerman, J., Forlizzi, J., and Evenson, S., 2007. Research through design as a method for interaction design research in HCI. CHI07 Proceedings. New York, NY: ACM Press, 493-502.

List of Publications

- Al Sarraj. W., De Troyer, O., (2009). Usability Framework for Web Mashup Makers for Casual User, Proceedings of The 9th International Conference on Web Engineering - Doctoral Consortium (ICWE2009-DC), Eds. Gustavo Rossi and Jon Iturrioz, Publ. CEUR-WS, ISBN 1613-0073, San Sebastian, Spain.

- Al Sarraj. W., De Troyer, O., (2010). Web Mashup Makers for Casual Users: A User Experiment, ACM proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services (iiWAS2010), ISBN 978-1-4503-0421-4, Paris, France.

- Al Sarraj. W., De Troyer, O., (2012). From a Conceptual Model for Identifying Usability Factors to a Usability Evaluation Framework for Mashup Makers for End-users, International Journal of Human-Computer Interaction, Print ISSN: 1044-7318, Online ISSN: 1532-7590, (in preparation for submitting).

Appendices

Appendix 1: Pilot study details

1.1 General-purpose mashup makers considered in the pilot study

Mashup Maker	Abbreviation	URL
Yahoo Pipes	YP	http://pipes.yahoo.com
Microsoft Popfly	MP	http://www.popfly.com
Intel mashmaker	IM	http://mashmaker.intel.com/web
Openkapow robomaker	OK	http://www.openkapow.com
IBM Mashup Center	IC	http://www.ibm.com/software/
Jackbe	JB	http://www.jackbe.com
Apatar	AP	http://www.apatar.com
Dapper	DA	http://www.dapper.com

1.2 Ranking activity by dimension level

Dimension level	Very high	High	Moderate	Low	Missed
Rank	5	4	3	2	1

1.3 preparation sheets

MashupMaker	Activity 1	Activity 2	Activity 3	Activity 4	Activity5

Ranking 5 = Excellent 4 = Good 3 = Satisfied 2 = Weak 1 = Missed

			Activity 1						Activity 2						Activity 3						Activity 4						Activity 5						Tot	Avr	Notes
			5	4	3	2	1		5	4	3	2	1		5	4	3	2	1		5	4	3	2	1		5	4	3	2	1				
1	Abstraction Gradient																																	
2	Closeness of mapping																																	
3	Consistency																																	
4	Diffuseness																																	
5	Error-proneness																																	
6	Hard mental operations																																	
7	Hidden dependencies																																	
8	Premature commitment																																	
9	Progressive evaluation																																	
10	Role-expressiveness																																	
11	Secondary notation																																	
12	Viscosity																																	
13	Visibility																																	

1.4 Activities considered in Mashup Makers evaluation

Activity	1	2	3	4	5
Action	Discover Mashup maker	Collect 2 websites data	Preface Mashup creation	Perform Mashup creation	Run Mashup

1.5 CDs Evaluation for Mashup Makers considered

Cognitive Dimension/ Mashup Maker	YP	MP	IM	OK	IC	JB	AP	DA
Abstraction Gradient	4	3	4	5	3	4	4	3
Closeness of mapping	4	4	3	3	3	5	5	4
Consistency	4	3	4	4	3	4	4	3
Diffuseness	4	4	3	4	5	4	4	4
Error-proneness	2	1	3	2	3	2	2	1
Hard mental operations	2	2	2	2	2	3	3	2
Hidden dependencies	3	4	3	4	2	4	4	4
Premature commitment	4	4	2	5	4	3	4	4
Progressive evaluation	1	2	3	3	2	3	3	2
Role-expressiveness	4	3	4	5	4	3	4	3
Secondary notation	2	3	3	3	2	2	3	3
Viscosity	4	4	3	5	3	2	2	4
Visibility	4	5	4	4	4	4	4	4

Appendix 2: SPSS results of user experiment

T-TEST PAIRS=YPit WITH YPnon (PAIRED) /CRITERIA=CI(.9500) /MISSING=ANALYSIS.

T-Test

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	YPit	4,1470	10	,18233	,05766
	YPnon	2,0280	10	,50006	,15813

Paired Samples Correlations

		N	Correlation	Sig.
Pair 1	YPit & YPnon	10	,491	,149

Paired Samples Test

		Paired Differences				
					95% Confidence Interval of the Difference	
		Mean	Std. Deviation	Std. Error Mean	Lower	Upper
Pair 1	YPit - YPnon	2,11900	,44014	,13918	1,80414	2,43386

Paired Samples Test

		t	df	Sig. (2-tailed)
Pair 1	YPit - YPnon	15,224	9	,000

T-TEST PAIRS=OMSit WITH OMSnon (PAIRED) /CRITERIA=CI(.9500) /MISSING=ANALYSIS.

T-Test

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	OMSit	3,7450	10	,30823	,09747
	OMSnon	1,7920	10	,84724	,26792

Paired Samples Correlations

	N	Correlation	Sig.

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
	OMSit	3,7450	10	,30823	,09747
Pair 1	OMSit & OMSnon		10	,213	,555

Paired Samples Test

		Paired Differences		
		Mean	Std. Deviation	Std. Error Mean
Pair 1	OMSit - OMSnon	1,95300	,83772	,26491

Paired Samples Test

		Paired Differences		t	df	Sig. (2-tailed)
		95% Confidence Interval of the Difference				
		Lower	Upper			
Pair 1	OMSit - OMSnon	1,35373	2,55227	7,372	9	,000

T-TEST PAIRS=DAit WITH DAnon (PAIRED) /CRITERIA=CI(.9500) /MISSING=ANALYSIS.

T-Test

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	DAit	4,4620	10	,18790	,05942
	DAnon	2,8840	10	,78561	,24843

Paired Samples Correlations

		N	Correlation	Sig.
Pair 1	DAit & DAnon	10	,358	,309

Paired Samples Test

		Paired Differences				
					95% Confidence Interval of the Difference	
		Mean	Std. Deviation	Std. Error Mean	Lower	Upper
Pair 1	DAit - DAnon	1,57800	,73938	,23381	1,04908	2,10692

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
	DAit	4,4620	10	,18790	,05942

Paired Samples Test

		t	df	Sig. (2-tailed)
Pair 1	DAit - DAnon	6,749	9	,000

T-TEST PAIRS=YPit WITH OMSit (PAIRED) /CRITERIA=CI(.9500) /MISSING=ANALYSIS.

T-Test

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	YPit	4,1470	10	,18233	,05766
	OMSit	3,7450	10	,30823	,09747

Paired Samples Correlations

		N	Correlation	Sig.
Pair 1	YPit & OMSit	10	,523	,121

Paired Samples Test

		Paired Differences				
					95% Confidence Interval of the Difference	
		Mean	Std. Deviation	Std. Error Mean	Lower	Upper
Pair 1	YPit - OMSit	,40200	,26360	,08336	,21343	,59057

Paired Samples Test

		t	df	Sig. (2-tailed)
Pair 1	YPit - OMSit	4,823	9	,001

SAVE OUTFILE='C:\Users\Miranda\Documents\waael.sav' /COMPRESSED.

T-TEST PAIRS=YPit WITH DAit (PAIRED) /CRITERIA=CI(.9500) /MISSING=ANALYSIS.

T-Test

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	YPit	4,1470	10	,18233	,05766
	DAit	4,4620	10	,18790	,05942

Paired Samples Correlations

		N	Correlation	Sig.
Pair 1	YPit & DAit	10	,651	,041

Paired Samples Test

		Paired Differences				
					95% Confidence Interval of the Difference	
		Mean	Std. Deviation	Std. Error Mean	Lower	Upper
Pair 1	YPit - DAit	-,31500	,15472	,04893	-,42568	-,20432

Paired Samples Test

		t	df	Sig. (2-tailed)
Pair 1	YPit - DAit	-6,438	9	,000

T-TEST PAIRS=OMSit WITH DAit (PAIRED /CRITERIA=CI(.9500) /MISSING=ANALYSIS.

T-Test

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	OMSit	3,7450	10	,30823	,09747
	DAit	4,4620	10	,18790	,05942

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
	OMSit	3,7450	10	,30823	,09747

Paired Samples Correlations

		N	Correlation	Sig.
Pair 1	OMSit & DAit	10	,796	,006

Paired Samples Test

		Paired Differences				
					95% Confidence Interval of the Difference	
		Mean	Std. Deviation	Std. Error Mean	Lower	Upper
Pair 1	OMSit - DAit	-,71700	,19522	,06174	-,85665	-,57735

Paired Samples Test

		t	df	Sig. (2-tailed)
Pair 1	OMSit - DAit	-11,614	9	,000

T-TEST PAIRS=YPnon WITH OMSnon (PAIRED) /CRITERIA=CI(.9500) /MISSING=ANALYSIS.

T-Test

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	YPnon	2,0280	10	,50006	,15813
	OMSnon	1,7920	10	,84724	,26792

Paired Samples Correlations

		N	Correlation	Sig.
Pair 1	YPnon & OMSnon	10	-,154	,672

Paired Samples Test

		Paired Differences		
		Mean	Std. Deviation	Std. Error Mean
Pair 1	YPnon – OMSnon	,23600	1,04793	,33138

Paired Samples Test

	Paired Differences	t	df	Sig. (2-tailed)

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
	YPnon	2,0280	10	,50006	,15813

		95% Confidence Interval of the Difference				
		Lower	Upper			
Pair 1	YPnon – OMSnon	-,51364	,98564	,712	9	,494

T-TEST PAIRS=YPnon WITH DAnon (PAIRED) /CRITERIA=CI(.9500) /MISSING=ANALYSIS.

T-Test

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	YPnon	2,0280	10	,50006	,15813
	DAnon	2,8840	10	,78561	,24843

Paired Samples Correlations

		N	Correlation	Sig.
Pair 1	YPnon & DAnon	10	,110	,762

Paired Samples Test

		Paired Differences		
		Mean	Std. Deviation	Std. Error Mean
Pair 1	YPnon - DAnon	-,85600	,88363	,27943

Paired Samples Test

		Paired Differences				
		95% Confidence Interval of the Difference				
		Lower	Upper	t	df	Sig. (2-tailed)
Pair 1	YPnon – Danon	-1,48811	-,22389	-3,063	9	,013

T-TEST PAIRS=OMSnon WITH DAnon (PAIRED) /CRITERIA=CI(.9500) /MISSING=ANALYSIS.

T-Test

Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	OMSnon	1,7920	10	,84724	,26792
	DAnon	2,8840	10	,78561	,24843

Paired Samples Correlations

		N	Correlation	Sig.
Pair 1	OMSnon & Danon	10	-,129	,723

Paired Samples Test

		Paired Differences		
		Mean	Std. Deviation	Std. Error Mean
Pair 1	OMSnon – Danon	-1,09200	1,22724	,38809

Paired Samples Test

		Paired Differences		t	df	Sig. (2-tailed)
		95% Confidence Interval of the Difference				
		Lower	Upper			
Pair 1	OMSnon – Danon	-1,96992	-,21408	-2,814	9	,020

Appendix 3: User experiment usability problems:

3.1 Usability problems in the user interface area

Usability problems	# of problems	Description
Unclear design areas	25	End-users were unaware or misunderstood the design areas of the Mashup Maker
Unclear design components	28	End-users were unaware or misunderstood the design components of the Mashup Maker. End-users are not able to match a Mashup Maker components with its real world equivalent
Confusing design metaphors	36	End-users were confused by the metaphors used and sometime complained that those metaphors mislead him/her. End-users wondered whether different words, situations, or actions mean the same thing.

3.2 Usability problems in the user interaction area

Usability problems	# of problems	Usability problems description
Slowing and/or stopping during design process of a Mashup.	29	End-users faced difficulties in understanding the working of the Mashup Maker while interacting with it. He/she expected other actions, and he/she wrongly reacted to the unexpected action. This caused slowing down the design process and in many cases the users had to stop and ask for help.
Inability to	35	End-users had to rely on their memory to

memorize components and steps during the creation of a Mashup		recall components rather than to recognize components.
Uncertainty and fears of making errors during steps cascading or components composing.	40	End-users were in doubt on how to interact with the Mashup Maker. Users often made mistakes and were looking for a clearly marked "emergency exit" as well as an undo and redo functionality.

3.4 Usability problems in the functional area

Usability problems	**# of problems**	**Usability problems description**
Unawareness and/or misunderstanding of component's meaning and functionality	38	End-users were unaware or misunderstood the meaning, goal and functionality of different Mashup creation components as well as operators and parameters.
Unawareness of consequences of components composition actions.	28	End-users had difficulties in composing the single components required for creation of a Mashup and didn't know how to order them.

Unawareness of consequences of cascading steps.	32	End-users had difficulties in defining the individual steps required for creation of a Mashup and how to specify the order in which these steps should be cascading or executed.
Ambiguity and/or uncertainty of using and building structures that represent models and reuse of designs.	25	End-users showed very low understanding with respect to using and reusing of designs that build models. End-users also complained about incompatibility of components and designs.

Appendix 4: Questionnaire to filled-in by the evaluator

	Question	disagree				agree
		1	2	3	4	5
1	Is the visual interface of the Mashup Maker clear?	❑	❑	❑	❑	❑
	Does the Mashup Maker have easy to understand and perceive user interface components?	❑	❑	❑	❑	❑
	Does the Mashup Maker allow to find user interface components easily and to see related components at the same time?	❑	❑	❑	❑	❑
	Visibility (weight)	❑	❑	❑	❑	❑
2	Is it easy to keep track of tasks performed and components used during the creation process of the Mashup?	❑	❑	❑	❑	❑
	Is it easier to make frequent tasks (e.g., in fewer steps) than non-frequent tasks? Does the difficulty of the tasks match their frequency?	❑	❑	❑	❑	❑
	Is the design of the Mashup easy and needs few intuitive thoughts?	❑	❑	❑	❑	❑
	Hard mental operations (weight)	❑	❑	❑	❑	❑
3	Does the Mashup Maker provide sufficient design space?	❑	❑	❑	❑	❑
	Do the visual interface components have proper size and shape?	❑	❑	❑	❑	❑
	Does the Mashup Maker provide space friendly component linking mechanisms?	❑	❑	❑	❑	❑
	Diffuseness (weight)	❑	❑	❑	❑	❑
4	Does the Mashup maker give you any way of defining new facilities or terms within the notation, so that you can extend it to describe new things or to express your ideas more	❑	❑	❑	❑	❑

	clearly or succinctly?					
	Is it easy to group components and can such a group be used as one element?	❑	❑	❑	❑	❑
	Is it easy to group tasks and can such a group be used as one element?	❑	❑	❑	❑	❑
	Abstraction gradient (weight)	❑	❑	❑	❑	❑
5	Are similar graphical entities (i.e. icons and menus) provided by the Mashup Maker laid out the same way?	❑	❑	❑	❑	❑
	Are semantically related components expressed by similar visual forms?	❑	❑	❑	❑	❑
	The Mashup Maker provides a consistent user interface (consistent naming, similar elements have similar meaning, etc)	❑	❑	❑	❑	❑
	Consistency (weight)	❑	❑	❑	❑	❑
6	Are only few actions needed to accomplish tasks during the Mashup design process?	❑	❑	❑	❑	❑
	Does the Mashup Maker provide suitable/understandable components that make it easy to change things after a design has been completed?	❑	❑	❑	❑	❑
	Is it easy to distinguish process cascading during the Mashup design process?	❑	❑	❑	❑	❑
	Viscosity (weight)					
7	In case of steps cascading, a step usually doesn't require premature commitment from the previous step.	❑	❑	❑	❑	❑
	Does the Mashup Maker provide the ability to change components or to redefine them during the Mashup design?	❑	❑	❑	❑	❑
	Are you able to select the order you liked when doing tasks	❑	❑	❑	❑	❑

	during Mashup design process?					
	Premature commitment (weight)	❑	❑	❑	❑	❑
8	Are dependencies, if any, clearly visible?	❑	❑	❑	❑	❑
	Do changes in one part of the visual interface affect other parts?	❑	❑	❑	❑	❑
	In case there is steps cascading, are dependency between steps clear?	❑	❑	❑	❑	❑
	Hidden dependencies (weight)	❑	❑	❑	❑	❑
9	Do components have semantically meaningful names?	❑	❑	❑	❑	❑
	Are most parts of the visual interface of the Mashup Makers easy to interpret?	❑	❑	❑	❑	❑
	In case of steps cascading, the achievement of a step doesn't require understanding of steps still to be achieved?	❑	❑	❑	❑	❑
	Role-expressiveness (weight)	❑	❑	❑	❑	❑
10	It is difficult to make mistakes?	❑	❑	❑	❑	❑
	Does the Mashup Maker prevent making mistakes?	❑	❑	❑	❑	❑
	When making mistakes, is it easy to undo the mistakes?	❑	❑	❑	❑	❑
	Error-proneness (weight)	❑	❑	❑	❑	❑
11	Does the Mashup Maker provide facilities and spaces to leave comments and notes (marks) or mechanism for brainstorming?	❑	❑	❑	❑	❑
	Does the Mashup Maker provide alternatives notations?	❑	❑	❑	❑	❑
	Does the Mashup maker provides easy and user friendly help tutorials?	❑	❑	❑	❑	❑
	Secondary notation (weight)	❑	❑	❑	❑	❑

12	Are all components provided by the Mashup Maker relevant?	❑	❑	❑	❑	❑
	Is the workflow for creating the Mashup self-explaining and clear?	❑	❑	❑	❑	❑
	The visual interface of the Mashup maker and its components are closely related to the process of creating Mashups.	❑	❑	❑	❑	❑
	Closeness of mapping (weight)	❑	❑	❑	❑	❑
13	Is it easy to stop in the middle of the design process and check your work so far?	❑	❑	❑	❑	❑
	Does the Mashup Maker provide methods to review completed or semi completed tasks?	❑	❑	❑	❑	❑
	Is it possible to try out partially completed mashups?	❑	❑	❑	❑	❑
	Progressive evaluation (weight)	❑	❑	❑	❑	❑
14	Is it possible to sketch things out when you were playing around with ideas during the design process of the Mashup?	❑	❑	❑	❑	❑
	Does the Mashup Maker provide methods and components to provisionally re-perform completed or semi completed tasks?	❑	❑	❑	❑	❑
	Does the Mashup maker provide ways to do “what if games” during the design process of the Mashup?	❑	❑	❑	❑	❑
	Provisionality (weight)	❑	❑	❑	❑	❑
15	**Visual support (weight)**	❑	❑	❑	❑	❑
	Interaction support (weight)	❑	❑	❑	❑	❑
	Functional support (weight)	❑	❑	❑	❑	❑
16	The most negative aspects of this Mashup Maker is ……. ……. …….					

17	The most positive aspects of this Mashup Maker is …….. ……. …….					
18	Overall suggestion to increase the usability: ……. ……. …….					
		Disagree				**agree**
		1	2	3	4	5
19	Overall opinion about usefulness of MUEF:	❑	❑	❑	❑	❑
20	Overall remarks and suggestions about MUEF and the evaluation: ……..……. …….. ……..…… ……..…… ……..…… …….					

Appendix 5: MUEF Evaluation process

5.1 Evaluation process Questionnaire (MUEF evaluation for experts)

	Question	disagree				agree
		1	2	3	4	5
1	MUEF method is user friendly.	❑	❑	❑	❑	❑
2	MUEF steps are a straightforward process.	❑	❑	❑	❑	❑
3	The materials provided by the MUEF fit my expectations/needs.	❑	❑	❑	❑	❑
4	Names and term provided by the MUEF clearly indicate their functionality.	❑	❑	❑	❑	❑
5	MUEF Steps and tasks provided and requested to be performed are relevant.	❑	❑	❑	❑	❑
6	MUEF provides methods to review completed or semi completed tasks.	❑	❑	❑	❑	❑
7	MUEF is not time consuming	❑	❑	❑	❑	❑
8	MUEF provides alternatives materials in case of misunderstanding	❑	❑	❑	❑	❑
9	Overall score of MUEF usefulness	❑	❑	❑	❑	❑
10	The most negative aspect of framework is …… …… ……					
11	The most positive aspect of this framework is …… …… ……					
12	Overall suggestion to enhance MUEF: ……					

	…… ……

Appendix 5.2 Evaluation results

Appendix 5.2.1 Table ¨Evaluators resume¨.

Evaluator	**Female**	**Male**	**HCI lecturer**	**Usability practitioner**	**User interface designer**	**Web development professional**
#	2	10	6	4	7	6

Appendix 5.2.2 Table ¨results of questionnaire 5.1 (average and standard deviation)¨.

	Question	Average
1	MUEF method is user friendly.	4.58
2	MUEF steps are a straightforward process.	4.32
3	The materials provided by the MUEF fit my expectations/needs.	3.93
4	Names and term provided by the MUEF clearly indicate their functionality.	4.19
5	MUEF Steps and tasks provided and requested to be performed are relevant.	3.87
6	MUEF provides methods to review completed or semi completed tasks.	3.92
7	MUEF is not time consuming	4.36
8	MUEF provides alternatives materials in case of misunderstanding	4.17
9	Overal score of MUEF usefulness	4.16