Journal of Theoretical and Applied Information Technology
10th February 2014. Vol. 60 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A SYSTEMATIC LITERATURE REVIEW OF END-USER PROGRAMMING FOR THE WEB MASHUP

RODZIAH LATIH[1], AHMED PATEL[2], ABDULLAH MOHD. ZIN[3]
Faculty of Information Science and Technology,
Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia
E-mail: [1]rodziah@ftsm.ukm.my, [2]amp@ftsm.ukm.my, [3]amz@ftsm.ukm.my

ABSTRACT

End-user Programming for the web is currently of interest because Web 2.0 technologies have resulted in a vast array of tools available for mashup making. This paper presents a Systematic Literature Review of EUP for web mashups. Its objective is to outline a comprehensive review and synthesis of the literature related to EUP for web mashups. A Systematic Literature Review was performed of peer reviewed published studies that focused on research in EUP for Web mashups. A review was conducted on 21 relevant articles, mostly recent (published between January 1st 2000 and December 31st 2012) and published in English. Five EUP approaches for web mashups were identified from the studies; browsing, programming by demonstration or example, spreadsheet, widget, data-flow and block-based approach. Other researches regarding EUP for web mashups were also identified, such as ubiquitous platform mashups, users' support functions, data extraction techniques, and process-oriented mashups.

Keywords: *SLR, End User Programming, Web 2.0, Web Meshup*

1. INTRODUCTION

The term 'End-user Programming' (EUP) was established by [1] referring to some of the programming tasks i.e. modifying or extending the software applications that were given to end-users. This group of users is called 'end-user programmers'. End-user programmers are experts in their domains and have taught themselves to program [2]. They can be mechanical engineers, doctors, physicists, teachers, accountants etc., but they write computer programmes in order to help themselves in their primary jobs. They may be experienced computer users, but they are not experienced in conventional programming languages such as C, C++, Java, etc. Other terms used referring to end-users are power users and casual users. Power users are a group of users who have no programming skills but have detailed functional knowledge about a specific tool or a set of tools. On the other hand, casual users are a group of users who only have the skills to use the functionality of the web browser and are able to navigate through the web [3].

Spreadsheet applications are the most popular end-user system environment [4] and as predicted by [5], about 60% of 90 million American end-users will utilize spreadsheet and database applications in the year 2012. This prediction shows the growing number of end-user programmers as well.

The emergence of Web 2.0 has changed the way software developers and end-users use the web [6]. The term Web 2.0 is commonly associated with web applications that facilitate interactive information sharing, interoperability, user-centred designs, and collaborations on the World Wide Web (WWW). People can share their thoughts, interests, photos, video clips, and others through social network applications like Facebook, MySpace, Flickr, Instagram, and Twitter. They publish their views in blogs and get instant responses and feedback from the e-communities. The deployment of Web 2.0 technologies resulted in the exponential growth of the number of end-user programmers compared to the number of software professionals because most web users are end-users. Web 2.0 technologies also support web customization and integration and this has led to the development of web mashup applications. The emergence of Web 3.0 technologies later on will definitely also support these necessities through semantic web, data mining, micro formats etc.

Journal of Theoretical and Applied Information Technology
10th February 2014. Vol. 60 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Web mashup applications are an interesting genre of interactive web applications that has become common recently. Web mashup applications are applications that integrate various data, presentations, and functionalities from two or more sources through Application Program Interfaces (APIs). Web mashup applications were developed by professional programmers because it involved multifaceted programming skills, thus requiring high programming skills. Therefore, to assist end-users in developing mashup applications, mashup tools can be used. Most mashup tools employ EUP techniques like scripting, data-flows, widgets, spreadsheets and Programming by Demonstration (PBE) to guide end-users in developing web mashup applications [3].

A Systematic Literature Review (SLR) was conducted to outline a comprehensive review and synthesis of the literature related to EUP for web mashups. The SLR is a systematic review of all available research results which aims to aggregate all existing evidence on research questions, leading to potential issues in identifying research gaps and contributions. It is believed that the results from this study can be used to improve both the mashup tools and the EUP techniques.

2. REVIEW METHOD

The review protocol for this SLR was developed by following the guidelines as proposed by [7]. The steps in this protocol are shown in Table 1.

Table 1: Phases and detailed SLR process steps.

Phase	Detailed steps
Planning	• Identify the need for SLR • Formulate review research questions.
Conducting	• Carry out a comprehensive search for primary studies. • Assess and record the criteria of included studies. • Classify data needed to answer the research questions. • Extract data from each included study. • Summarize and synthesise study results. • Interpret results to determine their applicability.
Documenting	• Write up study as a report.

This method includes three main phases; planning, conducting and documenting. In the planning phase, the need for conducting this review is identified and the Research Questions (RQ) that assist in the aim of this work are formulated based on the reasons that initiated this review. During this phase, the databases that will be used in the search for the articles, the main keywords, and the final search string for similar studies are identified.

The second phase is conducting the SLR process. The articles are searched for based on the search keywords and then scanned based on the inclusion and exclusion criteria. Then the articles are classified to answer the research questions (RQ). A data extraction form is used to record the data (Table 2). The third phase is documenting the process and findings.

Table 2: The data extracted from each study.

• The source and full references • Classification of the study (research report or empirical study) • Main topic area • Summary of the study

The review protocol of this SLR has been checked and evaluated by a few researchers with good experience in conducting literature reviews. Moreover, parts of this protocol have been previously used and established by several researchers in SLR [8] [9] [10].

2.1. Research questions

The SLR was conducted to obtain an overview of the research reported in the field of EUP for web mashups. In this review, the following research questions were addressed:

RQ 1: What studies are being addressed in the field of mashup tools development?
RQ 2: Which EUP technique has been commonly used?
RQ 3: Is there evidence that mashup tools are difficult to use by end-users?

2.2. Data source and search strategy

The strategy for collecting the relevant literature in this review is using a keyword search in a list of electronic databases of specific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

conference proceedings and journals papers. The data sources for this review are:

- ACM Digital Library (*dl.acm.org*)
- ScienceDirect – Elsevier (*www.sciencedirect.com*)
- IEEE Xplore (*ieexplore.org*)
- Directory of Open Access Journals (www.doaj.org)
- ISI Web of Science (*www.isiknowledge.com*)
- Springer LNCS (*www.springer.com/lncs*)
- Google Scholar (*scholar.google.com.my/*)

2.3. Keywords for Searching

Publications on web mashup tools development from the specific journals and conference proceedings were identified by searching through a variety of digital libraries. Only papers published in English between Jan 1st 2000 and December 31st 2012 were considered. To avoid overlooking relevant publications, all searches were carried out using two main search terms; "mashup" and "end-user programming." There are different spellings for the word "mashup" like "mash up" (without a hyphen) and "mash-up" (with a hyphen). These variants were also included in the search process. The term "end-user programming" is a common term referring to the programming techniques used by end-users. Therefore, the three words were used as one search keyword. The searches were conducted from April 2013 until May 2013. The bibliography for all the publications is stored in EndNote.

2.4. Inclusion and Exclusion Criteria

The selected papers collected should commit to a set of inclusion criteria as follows:

- The papers should be written in English.
- The papers should be published between January 1st 2000 and December 31st 2012.
- The papers should address web mashup development as an area of research. Therefore, the keywords "mashup" and "end-user programming" should exist either in the title, abstract or in a list of keywords.
- Only proceedings and journals were considered.

Accordingly, web pages, transactions, news, interviews, blogs, workshops, forums, reviews, discussions, posters, letters, tutorial or overhead presentations, opinion pieces, viewpoints, comments or purely anecdotal, technical reports, thesis, books or book chapters were not considered.

3. COLLECTING PROCESS

The SLR process is carried out in four steps (Table 3). The summary of the four steps of the review process and the number of articles identified at each step are shown in Figure 1.

Table 3: Four steps in SLR process.

Step	Tasks
1	**Collecting all the articles.** All articles from the identified digital libraries were searched using the keywords. Only articles published in the English language were downloaded. However, some articles could not be retrieved and these were considered as rejected.
2	**Applying inclusion and exclusion criteria.** The downloaded articles were then screened and articles that were not in the form of proceedings or journals were rejected.
3	**Verifying included articles.** The articles selected from the previous steps were skimmed and those that were not related to the topic of web mashup were rejected. A check was also done for repeated studies to ensure there are no duplications; for example if the same study is published in two different journals with different first authors, only one study would be included, usually the most comprehensive study or the most recent study.
4	**Extracting the data.** Lastly, the data from the selected articles were extracted and the name of the mashup tool and the EUP technique used were identified.

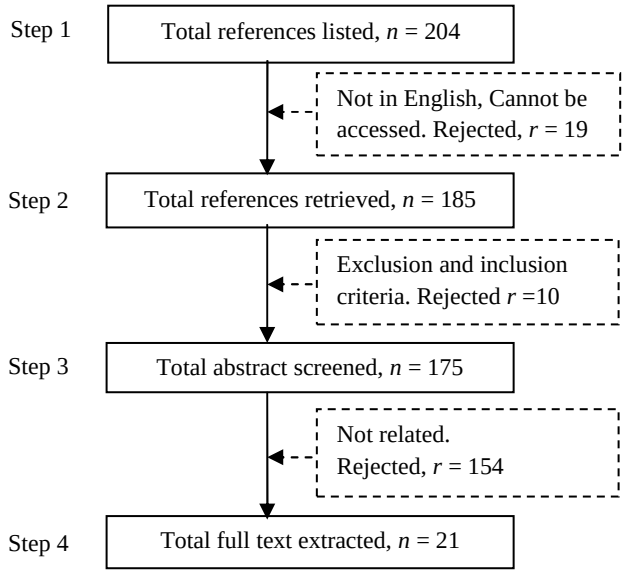

Figure 1: Literature Review Processes

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In step 1, 204 articles were found based on the search keywords. Three articles that were published in a non-English language and another 16 that could not be retrieved were rejected. In step 2, articles that were in the form of proceedings or journals were verified and articles that were in the form of book chapters, thesis, web pages, news, interviews, blogs, workshops, forums, reviews, discussions, posters, letters, tutorials or overhead presentations, opinion pieces, viewpoints, comments and purely anecdotal were excluded. 10 papers were rejected and only 175 papers were considered for the next stage.

In step 3, the output from the previous stage was skimmed and the abstracts of the articles were scanned. Research articles on EUP for web mashups were included and those that were not related to the topic or did not clearly explain the EUP techniques used were rejected. A check for repeated studies was also done to ensure there were no duplications; for example if the same study is published in two different journals with different first authors, only one study would be included, usually the most comprehensive study or the most recent study. As a result, only 21 articles regarding the research topics were identified and 154 articles that were not relevant to the study were rejected. Lastly in step 4, the data to identify the category for the articles, the main research focus, and the EUP paradigm used were extracted.

4. ANALYSIS

The total of the full texts extracted in this work was 21 articles only (Appendix A). Even though a criteria was set that searched articles must be published between January 1st 2000 and December 31st 2012, the earliest published articles retrieved was from the year 2007, the articles by (5) [11] and (17) [12]. The distribution of the articles according to the published year is as in Table 4.

Table 4: Distribution of retrieved articles.

Year	Articles	Number of articles
2007	(5,17)	2
2008	(2,3,13)	2
2009	(1,6,8,16)	4
2010	(9,12,14,15,20,21)	6
2011	(4,7,11,18,19)	5
2012	(10)	1

This section is organized according to the research questions. The articles are classified as in Table 5.

Table 5: The articles grouped according to the results.

Group	Subgroup	Articles
Research article	Browsing	(2, 17)
	PBD	(1)
	Spreadsheet	(8, 14)
	Widget	(3, 7)
	Data-flow	(5, 6)
	Ubiquitous platform	(4, 9)
	Support func.	(10)
	Extraction	(13)
	Process-oriented	(18)
	Block-based	(19)
Survey article		(11, 12, 15, 16, 20, 21)

4.1. Studies being addressed in the field of mashup tools development (RQ1)

The accessed articles were grouped into two groups; research and survey articles. However, only 10 out of 15 research articles discussed EUP approaches in making mashups. Those EUP approaches are *browsing*, *programming-by-demonstration (PBD)*, *spreadsheet*, *widget*, *data-flow*, and *block-based approach*. Articles that did not discuss any of the EUP approaches were grouped according to the article's main focus such as *ubiquitous platform mashups*, *users' support functions*, *data extraction techniques*, and *process-oriented mashups*.

The EUP approaches will be discussed in the following section on the next research question, RQ2. One of the retrieved articles discussed ubiquitous platform mashups. An *ubiquitous platform mashup* is a mashup that is developed on an ubiquitous platform like a mobile (4) [13] or a smart device (9) [14]. [13] described their work on mobile multimedia mashups as an ecosystem. The proposed ecosystem architecture consists of three domains; home domain, mobile domain and cloud domain. Likewise, another article [14] discussed an intermediate web based architecture in the context of a smart device. The proposed

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

architecture is called User Language Domain which uses a domain-specific embedded language approach.

In article (10) which is categorized in the *users' support functions* category, [15] proposed an approach called the "idea garden" which helps users to help themselves when composing data. The *data extraction technique* category also has an article (13) by [16] in which they proposed a method to integrate general web applications. This method considers both websites that provide public APIs (Application Programming Interfaces) and websites that do not. This is because current mashups are created by integrating only websites that provide public APIs like Google Maps, You Tube, and Amazon, while most existing websites do not provide this service. The last category is the *process-oriented enterprise mashup*. The example article (18) is by [17] which proposed the design of the *process-oriented enterprise mashup*. They argued that enterprise mashups can be formed in two ways; data-oriented and process-oriented.

4.2. EUP techniques in mashup tools (RQ2)

The EUP techniques used in mashup tools that were found in the review are *browsing, programming-by-demonstration (PBD), spreadsheet, widget, data-flow,* and *block-based approach* (Table 5).

The browsing paradigm was discussed in two articles; (2) [18] and (17) [12]. In the article [18], the researchers discussed the facet browsing approach used in their mashup tool called Potluck. The facet browsing approach allows the user to explore and identify subsets of data of interest or subsets of data that need alignment and clean up. The article by [12] on the other hand describes the browsing paradigm as an extension of normal browsing habits, where users work directly on the data found on the web and create the mashup immediately while browsing. Their mashup tool is called the MashMaker.

Programming by Demonstration (*PBD*) as discussed by (1) [19] is a technique to automatically populate spreadsheet-like tables with information collected from various websites. Using this technique, the user demonstrates a series of actions on how to fill the columns. These actions are recorded into scripts, which can be re-executed immediately on other rows in the table.

The spreadsheet paradigm is inspired by a spreadsheet-like programming pattern which works on the columns and the rows of a table. The article by (8) [20] introduced the Mashroom, a mashup development environment that uses the spreadsheet paradigm with an expressive data structure and a set of formally-defined mashup operators. They proposed nested tables as data models for the extracted data services. The advantage is that this nested table is simple and allows access to the underlying data sources intuitively. In their article (14) [21] also proposed the use of the spreadsheet paradigm in making mashups. However, they introduced a two-level programming model that combined spreadsheet-like visual programming and data flow chart programming to record the order of operations and data dependencies during the mashing process.

The widget paradigm was implemented in two articles; (3) [22] and (7) [23]. The article by [22] discussed the composition of widgets to make mashups in a situational mashup system. A widget is a small application with limited functionalities that is executed on the user's website. The widget paradigm works either through a pure program or by calling on other web services. In their article,[23] introduced fladget or a Flash widget which refers to RIA (Rich Internet Applications) oriented Wiki add-ons.

Another EUP paradigm that was found in this review is the *data-flow paradigm*. Two articles on data-flow paradigm were found which were articles by (5) [11] and (6) [24]. The article by [11] discussed the *data-flow paradigm*, the idea of which they adopted from the UNIX pipe. Web services are represented by blocks called operators. The operators need to be connected to each other and the data will flow from one block to another block. Therefore each block should have an input and an output. However, the article by [24] proposed a combination of data-flow paradigm and scripting paradigm in making mashups. The selected components or widgets are wired together and the user can write a simple script for customizing. The last approach found is the *block-based development approach* that was introduced by (19)[25]. The block-based development approach is a combination of end-

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

user programming techniques with component-based software development. The block-based development approach allows users to develop an application by integrating several blocks. These blocks are pre-developed blocks that support certain tasks or functions.

4.3. Empirical studies on mashup tools (RQ3)

Six survey articles were found that matched the search keywords; (11) [26], (12) [27], (15) [28], (16) [29], (20) [30], and (21) [31]. The article by (11) [26] presented a qualitative empirical result of the end-user mashup programming study from the perspective of the end-user programmers' problem-solving processes. Programming is difficult for end-users. Besides, problem solving, creativity, and design thinking are also barriers for end-users in the programming environment. To overcome these problems, they proposed that end-users initiate and refine their own ideas. The result from their empirical study shows that end-users need help and support to become confident along the process. The article by (12) [27] on the other hand presented results from a survey conducted to explore the factors that motivate end-users to learn about and explore remix tools. The finding suggests that end-users are much more socially motivated. Therefore, there is a need to look towards social solutions when building tools to support them.

The article by (15) [28] is the only article retrieved that reports a review on mashup tools. This article presented review results of ten mashup tools; Yahoo Pipes, iGoogle, Apatar, IBM Lotus Mashups, Intel MashMaker, Marmite, Vegemite and Dapper. In their review, they found that some mashup tools are not really simple enough to handle and require the end-users to have a computer programming background to learn and understand its platform infrastructures and mechanisms. The article by (16) [29] presented a review of mashup literature to classify the subtopics in mashup researches. They found five common themes across multiple research studies; mashup aggregate content from disparate sources, integration as a technical challenge when developing mashups, information overload, ability of end-users to create custom mashup applications and finally a set of issues like security, availability and quality. In their article (20) [30] presented their think-aloud study with ten end-users creating a web mashup. The objective was to explore their design of a theory-based approach in order to understand and investigate programming by the end-user. The results showed opportunities for the environment to support end-users programming as a design activity.

The last article retrieved in the survey article category was an article by (21) [31]. In this article, he reported the results of a pilot experiment on open-ended mashup assignments using Yahoo Pipes, an end-user web-based visual development environment. The respondents stated that the tool was useful, interesting, appropriate and of the right level of difficulty. They also indicated that they were able to learn the tool in a short period of time.

5. DISCUSSION

The purpose of this study is to provide an overview of EUP for web mashups by reviewing and analysing published research articles. This work has been done based on the SLR guidelines as proposed by [7]. In this section, three main topics are discussed based on the findings of the study; mashup, EUP and EUP for web mashups. The keyword relationship model is as in Figure 2.

5.1. Mashup

The term mashup was actually derived from the music industry, where the original contents from various artists are remixed to create new material [32]. However, in the field of Computer Science, the term mashup can be defined as websites that combine multiple websites to support unique tasks through APIs [11]. It not only combines the data but also the process or view from several websites to provide information that could not be easily obtained by manually browsing the websites separately [12, 19].

Making mashups can be difficult even for those with programming experience because it requires knowledge of more than one programming language, several markup languages, and an understanding of how to assemble those elements together on the web. Making mashups involve five processes; data retrieval, source modelling, data cleaning, data integration and data display [33]. However, a study by [34] has divided the processes into three

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

stages; data gathering, data manipulation and data visualization. Their findings show that data manipulation was the most difficult step to understand and implement in making mashups.

Consequently, several mashup tools were developed to help end-users who lack programming skills in making mashups. In this review, several mashup tools were noted, such as Vegemite [19], Potluck [18], Marmite [11], MashMaker [12], Mashroom [35], SituMash [22], Lively Fabrik [24] and Whip [25]. The other tools that are also mentioned in this review are Karma [33], MASH [36], MaxMash [37], Xtractorz [38] and SpiderCharlotte [39]. From this review only two articles were found; articles (15) [28] and (21) [31] that reported the results of mashup tools reviews and evaluations. The article by (21) [31] reported the results of a pilot experiment on open-ended mashup assignments using Yahoo Pipes and found that the mashup tool they used is easy to learn. However, article (15) by [28] stated that in a study of ten mashup tools, it was found that some mashup tools are difficult to handle and require end-users to have a computer programming knowledge. Their review criteria were programming skills requirement, prompt suggestion to use, operability, share-ability and reusability, service, type and target users. Other studies that also stated the same findings are those of [40], [41] and [42]. These articles are not on the list of reviewed articles for this study. In their article [40] reported their user experiment studied three mashup tools; Yahoo Pipes, Open Mashup Studios and Dapper. Their evaluation criteria are visibility, hard mental operation, diffuseness, abstraction gradient, consistency, error-pronounce, role-expressiveness, progressive evaluation, viscosity, and provisionality and premature commitment.

Mashup tools also have been developed for mobiles and smart devices [13, 14]. In contrast with desktop-based mashup tools, these mobile and smart device mashup tools execute within a lightweight framework which supports both desktop and mobile devices [43]. Most lightweight mashup tools use simple interfaces like a widget programming approach to create mashups.

5.2. End-user Programming

EUP refers to programming activities by end-users (which are any computer users). End-users who write a program are called end-user programmers. However they are not professional programmers. They write a program or explicitly modify the software primarily for personal use or for a small group of users rather than for public use [44]. EUP can be seen in many areas like engineering, accounting, and education. End-users modify the application to complement their work. Recently with the popularity of smart phones, smart devices and Web 2.0, EUP also has been adopted in ubiquitous and web computing. It is well-known that conventional programming languages are hard to learn and use, demanding skills that many people do not have. In an attempt to make the programming easier, several approaches were introduced such as programming by example (also called programming by demonstration), visual programming, and scripting languages [45].

Programming by Example is a way of programming where the user of the system writes a programme by giving an example of what the programme should do. The system records the sequence of actions and performs it again. Programming by example allows a user to create programmes without doing conventional programming. [46] described Programming by Example precisely as “Do What I Did.” Other terms for Programming by Example are Programming by Demonstration and Programming by Sample.

Another approach for EUP is visual programming. [47] defined visual programming as a programme in two or more dimensional fashions and [48] defined visual programming as visual representations to accomplish what would otherwise have to be written in a traditional one-dimensional programming language. The characteristics of visual programming languages are that they have fewer concepts to programme, a concrete programming process, an explicit depiction of relationships and immediate visual feedbacks [49]. Examples of visual programming are ClockWorks [50], RAPTOR [51] and LabVIEW [52].

Writing small programmes called scripts or macros have been widely used in many commercial systems like Microsoft Word and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Excel. It is helpful for automating repetitive tasks and documenting what the user did. A macro or a script is a type of application-specific language as it is a part of the software application facilities or a part of the programming language. A macro or a script is written in a special language called scripting language like tcl/tk, Python, Perl, and JavaScript. Scripting languages are interpreted languages and are supposed to be a simplified from general-purpose programming languages like C, C++ or Java, because they are intended for a specific domain or environment. Moreover, some of the languages are designed for end-users. However, because of its powerful library, it is difficult for end-users to understand and use it.

The spreadsheet paradigm is another EUP approach that is widely used by end-users. The spreadsheet paradigm was initially from spreadsheet calculations, a fast process to express data and make calculations for the data on two-dimensional sheets [53]. The well-known example of the spreadsheet system is Microsoft Excel where users can use either a calculation formula or a script to perform several tasks. The spreadsheet paradigm offers several advantages like a direct manipulation interface that is easy to view, navigate and interact with the data [54]. The spreadsheet paradigm also provides a feature called "What You See Is What You Test" (WYSIWYT) which offers an immediate automatic visual feedback, a declarative approach to programming and has a dependence-driven characteristic [55, 56].

Natural programming is an approach that lets the non-professional programmer write a programme using natural language [57]. The goal is to make it possible for people to express their ideas in the same way they think about it. Examples of natural programming are HANDS [58], Board Game Language - BGL [59] and Grammatical Framework – GF [60]. Several works attempt to develop spoken natural language programming by integrating it with speech recognition instead of typed textual input [61].

Block-based programming is a new software development approach proposed by [62]. Block-based programming is derived from the combination of Component-based Software Development Approach with the EUP approach. Application software can be developed by integrating the pre-developed blocks. The blocks are a simple unit and it is easy for end-users to customize the blocks and build the application for their needs. The block-based software development approach involves two types of developers; block developers and application developers [63]. However, research in the block-based software development approach is still in progress. The list of EUPs is as in Table 6.

Table 6: List of EUP.

EUP Technique
Programming by Example/ Demonstration
Visual Programming
Macro programming/ Scripting
Natural paradigm
Spreadsheet Programming
Block-based programming

5.3. End-user Programming for the Web Mashup

The emergence of the World Wide Web (WWW) and specifically Web 2.0 has provided the opportunity for end-users to automate and customize selected web pages. Unfortunately, the complexity of current web technologies prevents most users from realizing this opportunity. To overcome this problem mashup tools were developed. The aim is to help the end-user create mashups without conventional programming but instead by using metaphors, formulae, sequence of GUI actions, circuit diagrams or application-specific languages or scripts. These alternative approaches are called EUP approaches. EUP approaches that are used within mashup tools are PBD/PBE, data-flow, spreadsheet, scripting, visual, widget, block-based and browsing paradigms (Table 7).

Table 7: List of EUP for the Web Mashup.

EUP Technique for the Web Mashup
Programming by Example/ Demonstration
Visual Programming
Scripting Languages
Spreadsheet paradigm
Data-flow/ work-flow
Widget paradigm
Browsing paradigm
Block-based development approach

Several mashup tools use mixed paradigms like Vegemite that combines PBD with the spreadsheet paradigm [19]. Vegemite consists of two main parts; the spreadsheet-like table called

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Vegetable to store the data and the CoScripter engine that runs the script for recording and playback actions during data extraction. Marmite also uses the mixed approach where it combines a spreadsheet-like table with a data-flow or work-flow approach [64]. Most mashup tools decide on the spreadsheet paradigm for its "What You See is What You Get" feature that is easy for end-users to understand. The scripting language is instead provided for customization. The widget paradigm is mostly employed in mobile and smart device mashup tools.

In this review, no reviews or empirical study articles on EUP for web mashups were found. However, there was an article [65] that evaluated a set of data-flow selection strategies in terms of efficiency and effectiveness. These data-flow selection strategies are used in general applications. In the data-flow paradigm, operators have input and output and can be connected together. The problem with the data-flow paradigm is that users can get confused with the role of the data and the operators [11].

6. CONCLUSION

The objective of this work is to outline a comprehensive review and synthesis of the current state-of-the-art related to EUP for the web mashups. 204 articles were identified based on the search keywords, of which 21 were found to be relevant. The articles were divided into two main groups; research articles and survey articles. Six EUP approaches for web mashups were identified from the studies; browsing, programming-by-demonstration (PBD), spreadsheet, widget paradigm, data-flow and block-based approach. Other researches regarding EUP for web mashups were identified, such as ubiquitous platform mashups, users' support functions, data extraction techniques and process-oriented mashups. The findings show that there are still many areas that need to be explored such as ubiquitous platform mashups, lightweight frameworks, enterprise mashups etc. The developments of mashup tools that are supposed to assist end-users in making mashups still have limitations. While most mashup tools have been developed to have a simple user interface, to be easy to use and learn, to be reusable and extendable and to be able to be customized, and yet other outstanding criteria in mashup tools are flexibility and adaptability.

REFERENCES:

[1] B. A. Nardi, *A Small Matter of Programming: Perspectives on End User Computing*: MIT Press, 1993.

[2] W. Harrison. (2004) The Dangers of End-User Programming. *IEEE Software*. 5-7.

[3] T. Fischer, F. Bakalov, and A. Nauerz, "An Overview of Current Approaches to Mashup Generation," in *International Workshop on Knowledge Services and Mashups (KSM09)*, Solothurn, Switzerland, 2009.

[4] B. A. Nardi and J. R. Miller, "The spreadsheet interface: A basis for end-user programming," in *Human-Computer Interaction: INTERACT '90*, Amsterdam: North-Holland, 1990.

[5] C. Scaffidi, M. Shaw, and B. Myers, "Estimating the Numbers of End Users and End User Programmers " in *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)*, 2005.

[6] T. O'Reilly. (2005, Sep 30). What is Web 2.0? Design Patterns and Business Models for the Next Generation of Software. *2005(30)*. Available: http://oreilly.com/web2/archive/what-is-web-20.html

[7] B. Kitchenham and S. Charters, "Guidelines for performing Systematic Literature Reviews in Software Engineering," Keele University2007.

[8] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic Literature Reviews in Software Engineering - A Systematic Literature Review," *Information and Software Technology*, vol. 51, pp. 7-15, 2009.

[9] S. Beecham, N. Baddoo, T. Hall, H. Robinson, and H. Sharp, "Motivation in Software Engineering: A Systematic Literature Review," *Information and Software Technology*, vol. 50, pp. 860-878, August 2008.

[10] T. Dyba and T. Dingseyr, "Empirical Studies of Agile Development: A Systematic Review," *Information and Software Technology*, vol. 50, pp. 833-859, August 2008.

[11] J. Wong, "Marmite: Towards End-User Programming for the Web," in *IEEE*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Symposium on Visual Languages and Human-Centric Computing, 2007. VL/HCC 2007, 2007, pp. 270-271.

[12] R. Ennals and D. Gay, "User-friendly functional programming for web mashups," *ACM Sigplan Notices*, vol. 42, pp. 223-233, Sep 2007.

[13] A. Salminen, J. Kallio, and T. Mikkonen, "Towards Mobile Multimedia Mashup Ecosystem," in *IEEE International Conference on Communications Workshops (ICC)*, 2011, pp. 1-5.

[14] N. Ahmadi, F. Lelli, and M. Jazayeri, "Supporting Domain-Specific Programming in Web 2.0: A Case Study of Smart Devices," in *21st Australian Software Engineering Conference (ASWEC)*, 2010, pp. 215-223.

[15] J. Cao, "The idea garden: From a qualitative evaluation toward an quantitative evaluation and generalization," in *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2012, pp. 219-220.

[16] H. Han and T. Tokuda, "A Method for Integration of Web Applications Based on Information Extraction," in *Eighth International Conference on Web Engineering (ICWE '08)*, 2008, pp. 189-195.

[17] P. d. Vrieze, L. Xu, A. Bouguettaya, J. Yang, and J. Chen, "Building enterprise mashups," *Future Generation Computer Systems-the International Journal of Grid Computing and Escience*, vol. 27, pp. 637-642, May 2011.

[18] D. F. Huynh, R. C. Miller, and D. R. Karger, "Potluck: Data mash-up tool for casual users," *Journal of Web Semantics*, vol. 6, pp. 274-282, Nov 2008.

[19] J. Lin, J. Wong, J. Nichols, A. Cypher, and T. A. Lau, "End-user programming of mashups with Vegemite," in *14th International Conference on Intelligent User Interfaces (IUI'09)*, 2009.

[20] G. Wang, S. Yang, and Y. Han, "Mashroom: end-user mashup programming using nested tables," presented at the Proceedings of the 18th international conference on World wide web, Madrid, Spain, 2009.

[21] H. Lin, G. Wang, P. Zhang, J. Wang, and Y. Han, "A Two-Level Programming Model Based on Spreadsheet and Data Flow Chart," in *7th Web Information Systems and Applications Conference (WISA)*, 2010, pp. 39-42.

[22] A. F. M. Huang, S. B. Huang, L. E.Y.F., and S. J. H. Yang, "Improving End User Programming with Situational Mashups in Web 2.0 Environment," in *IEEE International Symposium on Service-Oriented System Engineering 2008 (SOSE'08)*, Jhongli, 2008, pp. 62-67.

[23] M. Tosic and M. Manic, "A RESTful technique for collaborative learning content transclusion by Wiki-style mashups," in *5th IEEE International Conference on e-Learning in Industrial Electronics (ICELIE)*, 2011, pp. 38-43.

[24] J. Lincke, R. Krahn, D. Ingalls, and R. Hirschfeld, "Lively Fabrik A Web-based End-user Programming Environment," in *Seventh International Conference on Creating, Connecting and Collaborating through Computing*, 2009, pp. 11-19.

[25] R. Latih, A. Patel, A. M. Zin, T. Yiqi, and S. H. Muhammad, "Whip: A framework for mashup development with block-based development approach," in *International Conference on Electrical Engineering and Informatics (ICEEI)*, 2011, pp. 1-6.

[26] J. Cao, S. D. Fleming, and M. Burnett, "An Exploration of Design Opportunities for "Gardening" End-user Programmers' Ideas," in *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2011, pp. 35-42.

[27] N. Zang, "Information Remix and the Motivations of Everyday End-Users," in *Visual Languages and Human-Centric Computing (VL/HCC), 2010 IEEE Symposium on*, 2010, pp. 212-215.

[28] A. Patel, L. Na, R. Latih, C. Wills, Z. Shukur, and R. Mulla, "A Study of Mashup as a Software Application Development Technique with Examples from an End User Programming Perspective," *Journal of Computer Science*, vol. 6, pp. 1406-1415, 2010.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[29] B. Beemer and D. Gregg, "Mashups: A Literature Review and Classification Framework," *Future Internet*, vol. 1, pp. 59-87, 2009.

[30] J. Cao, Y. Riche, S. Wiedenbeck, M. Burnett, and V. Grigoreanu, "End-user mashup programming: through the design lens," presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Atlanta, Georgia, USA, 2010.

[31] K.-B. Yue, "Experience on Mashup Development with End User Programming Environment " *Journal of Information Systems Education*, vol. 21, pp. 111-119, 2010.

[32] S. Murugesan, "Understanding Web 2.0," *IT Professional*, vol. 9, pp. 34-41, 2007.

[33] R. Tuchinda, C. A. Knoblock, and P. Szekely, "Building Mashups by Demonstration," *Acm Transactions on the Web*, vol. 5, Jul 2011.

[34] N. Zang, "Mashups for the Web-active User," in *Visual Languages and Human-Centric Computing, 2008. VL/HCC 2008. IEEE Symposium on*, 2008, pp. 276-277.

[35] G. Wang, S. Yang, and Y. Han, "Mashroom: End-User Mashup Programming Using Nested Tables," in *WWW 2009 MADRID*, Madrid, Spain, 2009.

[36] L. Mariani and F. Pastore, "MASH: A tool for end-user plug-in composition," in *34th International Conference on Software Engineering (ICSE)*, 2012, pp. 1387-1390.

[37] M. Shevertalov and S. Mancoridis, "A Case Study on the Automatic Composition of Network Application Mashups," in *23rd IEEE/ACM International Conference on Automated Software Engineering, 2008. ASE 2008*, 2008, pp. 359-362.

[38] R. A. G. Gultom, R. F. Sari, and B. Budiardjo, "Implementing web data extraction and making Mashup with Xtractorz," in *Advance Computing Conference (IACC), 2010 IEEE 2nd International*, 2010, pp. 385-393.

[39] G. Wang, S. Yang, and Y. Han, "A Spreadsheet-like Construct for Streamlining and Reusing Mashups," in *The 9th International Conference for Young Computer Scientists*, 2008, pp. 880-885.

[40] W. A. Sarraj and O. D. Troyer, "Web mashup makers for casual users: a user experiment," in *The 12th International Conference on Information Integration and Web-based Applications & Services (iiWAS '10:)*, 2010.

[41] L. Grammel and M.-A. Storey, "An End User Perspective on Mashup Maker," University of Victoria, Victoria, BC, CanadaSeptember 2008.

[42] G. D. Lorenzo, H. Hacid, H.-y. Paik, and B. Benatallah, "Data Integration in Mashups," *Sigmod Record*, vol. 38, pp. 59-66, Mar 2009.

[43] M. Albinola, L. Baresi, M. Carcano, and S. Guinea, "Mashlight: A Lightweight Mashup Framework for Everyone," in *WWW2009*, Madrid, Spain, 2009.

[44] A. J. Ko, R. Abraham, L. Beckwith, A. Blackwell, M. Burnett, M. Erwig, C. Scaffidi, J. Lawrance, H. Lieberman, B. Myers, M. B. Rosson, G. Rothermel, M. Shaw, and S. Wiedenbeck, "The State of the Art in End User Software Engineering," *ACM Computing Surveys*, vol. 43, pp. 21-44, 2011.

[45] H. Lieberman, F. Paterno, M. Klann, and V. Wulf, "End-User Development: An Emerging Paradigm," in *End-User Development*, H. Lieberman, F. Paterno, and V. Wulf, Eds., 1 ed: Kluwer Academic Publishers, 2006, pp. 1-8.

[46] D. C. Halbert, "Programming by Example," PhD, Computer Science Division, Dept of EE & CS,, University of California, Berkeley, 1984.

[47] B. A. Myers, "Visual Programming, Programming by Example and Program Visualization: A Taxonomy," in *The SIGCHI Conference on Human Factors in Computing Systems (CHI'86)*, 1986, pp. 59-66.

[48] N. C. Shu, *Visual Programming*. New York: Van Nostrand Reinhold Co., 1988.

[49] M. M. Burnett, M. J. Baker, C. Bohus, P. Carlson, S. Yang, and P. v. Zee, "Scaling up Visual Programming Languages," *Computer*, vol. 28, pp. 45-54, 1995.

[50] T. C. N. Graham, C. A. Morton, and T. Urnes, "ClockWorks: Visual

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Programming of Component-based Software Architectures," *Journal of Visual Languages and Computing*, vol. 7, pp. 175-196, 1996.

[51] M. C. Carlisle, T. A. Wilson, J. W. Humphries, and S. M. Hadfield, "RAPTOR: A Visual Programming Environment for Teaching Algorithmic Problem Solving," in *36th SIGCSE Technical Symposium on Computer Science Education (SIGCSE'05)*, 2005, pp. 176-180.

[52] R. Jamal and L. Wenzel, "The Applicability of the Visual Programing Language LabVIEW to Large Real-World Applications," in *11th IEEE International Symposium on Visual Language*, Darmstadt, 1995, pp. 99-106.

[53] M. Tukiainen, "Uncovering Effects of Programming Paradigms: Errors in Two Spreadsheet Systems," in *12th Workshop of the Psychology of Programming Interest Group*, Cozenza Italy, 2000, pp. 247-266.

[54] E. H.-h. Chi, J. Riedl, P. Barry, and J. Konstan, "Principles for Information Visualization Spreadsheets," *IEEE Computer Graphics and Applications*, vol. 18, pp. 30-38, 1998.

[55] A. Ambler, M. Burnett, and B. Zimmerman, "Operational Versus Definitional: A Perspective on Programming Paradigms.," *Computer*, vol. 25, pp. 28-43, 1992.

[56] K. J. Rothermel, C. R. Cook, M. M. Burnett, J. Schonfeld, T. R. G. Green, and G. Rothermel, "WYSIWYT Testing in the Spreadsheet Paradigm: An Empirical Evaluation," in *ICSE 2000*, Limerick, Ireland, 2000, pp. 230-239.

[57] B. A. Myers, "Natural Programming: Project Overview and Proposal," Human-Computer Interaction Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 152131998.

[58] J. Pane and B. Myers, "More natural programming languages and environments," in *End User Development*, e. Henry Lieberman et al., Ed., ed: Springer, 2006, pp. 31-50.

[59] A. Riker, "Natural Language in Programming An English Syntax-based Approach for Reducing the Difficulty of First Programming Language Acquisition," Master's Thesis, Department of Computer Science, The Faculty of the Graduate School of Arts and Sciences, Brandeis University, Waltham, Massachusetts, 2010.

[60] B. Bringert, "Programming Language Techniques for Natural Language Applications," Degree of Doctor of Engineering, Department of Computer Science and Engineering, Chalmers University of Technology and University of Gothenburg, Göteborg, Sweden, 2008.

[61] H. Lieberman and H. Liu, "Feasibility Studies For Programming In Natural Language," in *End-User Development*, H. Lieberman, F. Paterno, and V. Wulf, Eds., ed: Kluwer Academic Publishers/Springer, 2005.

[62] A. M. Zin, "Block-Based Approach for End User Software Development," *Asian Journal of Information Technology*, vol. 10, pp. 249-258, 2011.

[63] S. N. H. Mohamad, A. Patel, Y. Tew, R. Latih, and Q. Qassim, "Principles and Dynamics of Block-based Programming Approach," in *2011 IEEE Symposium on Computers and Informatics (ISCI 2011)*, Kuala Lumpur, Malaysia, 2011, pp. 340-345.

[64] J. Wong and J. I. Hong, "Making mashups with Marmite: Towards End-User Programming for the web," in *SIGCHI Conference on Human Factors in Computing Systems (CHI'07)*, New York, USA, 2007, pp. 1435-1444.

[65] K. Winbladh and A. Ranganathan, "Evaluating test selection strategies for end-user specified flow-based applications," in *Automated Software Engineering (ASE), 2011 26th IEEE/ACM International Conference on*, 2011, pp. 400-403.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Appendix A: *List of Reviewed Articles.*

1. James Lin, Jeffrey Wong, Jeffrey Nichols, Allen Cypher and Tessa A. Lau, 2009. End-user programming of mashups with Vegemite, IUI '09: Proceedings of the 14th International Conference on Intelligent User Interfaces.
2. David F. Huynh, Robert C. Miller and David R. Karger, 2008. Potluck: Data mash-up tool for casual users, Web Semantics: Science, Services and Agents on the World Wide Web, Volume 6, Issue 4, November 2008, pp. 274-282.
3. Angus F.M. Huang, Shin Bo Huang, Evan Y.F. Lee and Stephen J.H. Yang, 2008. "Improving End-User Programming with Situational Mashups in Web 2.0 Environment,", IEEE International Symposium on Service-Oriented System Engineering 2008 (SOSE '08), pp.62-67.
4. A. Salminen, J. Kallio and T. Mikkonen, 2011. "Towards Mobile Multimedia Mashup Ecosystem", IEEE International Conference on Communications Workshops (ICC), pp.1-5.
5. J. Wong, 2007. "Marmite: Towards End-User Programming for the Web", IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2007), pp.270-271.
6. Jens Lincke, Robert Krahn, Dan Ingalls, and Robert Hirschfeld, 2009. "Lively Fabrik A Web-based End-user Programming Environment", Seventh International Conference on Creating, Connecting and Collaborating through Computing, (C5 '09), pp.11-19.
7. Milorad Tosic and Milos Manic, 2011. "A RESTful technique for collaborative learning content transclusion by Wiki-style mashups", 5th IEEE International Conference on e-Learning in Industrial Electronics (ICELIE), pp.38-43.
8. Guiling Wang, Shaohua Yang, and Yanbo Han, 2009. "Mashroom: end-user mashup programming using nested tables", Proceedings of the 18th international conference on World Wide Web, pp. 861-870.
9. Navid Ahmadi, F. Lelli, and M. Jazayeri, 2010. "Supporting Domain-Specific Programming in Web 2.0: A Case Study of Smart Devices", 21st Australian Software Engineering Conference (ASWEC), pp.215-223.
10. Jill Cao, 2012. "The idea garden: From a qualitative evaluation toward an quantitative evaluation and generalization", IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), pp.219-220.
11. Jill Cao, S.D Fleming, M. Burnett, 2011. "An exploration of design opportunities for “gardening” end-user programmers' ideas," IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), pp.35-42.
12. Nan Zang, 2010. "Information Remix and the Motivations of Everyday End-Users", IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), pp.212-215.
13. Hao Han and Takehiro Tokuda, 2008. "A Method for Integration of Web Applications Based on Information Extraction", Eighth International Conference on Web Engineering (ICWE '08), pp.189-195.
14. Hailun Lin, Guiling Wang, Peng Zhang, Jing Wang and Yanbo Han, 2010. "A Two-Level Programming Model Based on Spreadsheet and Data Flow Chart", 7th Web Information Systems and Applications Conference (WISA), pp.39-42.
15. Ahmed Patel, Liu Na, Rodziah Latih, Christopher Wills, Zarina Shukur and Rabia Mulla, 2010. “A Study of Mashup as a Software Application Development Technique with Examples from an End-User Programming Perspective”, Journal of Computer Science 6 (12): 1406-1415.
16. Brandon Beemer and Dawn Gregg, 2009. “Mashups: A Literature Review and Classification Framework”, Future Internet 2009, 1(1):59-87.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

17. Rob Ennals and David Gay, 2007. "User-friendly functional programming for web mashups", 12th ACM SIGPLAN International Conference on Functional Programming, 42(9):223-233.
18. Paul de Vrieze, Lai Xu, Athman Bouguettaya, Jian Yang, and Jinjun Chen, 2011. "Building Enterprise Mashups." Future Generation Computer Systems 27(5): 637-642.
19. Rodziah Latih, Ahmad Patel, Abdullah M. Zin, Tew Yiqi, and Siti H. Muhammad, 2011. "Whip: A framework for mashup development with block-based development approach", International Conference on Electrical Engineering and Informatics (ICEEI), pp.1 – 6.
20. Jill Cao, Yann Riche, Susan Wiedenbeck, Margaret Burnett, and Valentina Grigoreanu, 2010. "End-user mashup programming: through the design lens", Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp.1009-1018.
21. Kwok-Bun Yue, 2010. "Experience on mashup development with end user programming environment." Journal of Information Systems Education, 21(1): 111-119.