Future Internet **2009**, *1*, 59-87; doi:10.3390/fi1010059

OPEN ACCESS

future internet

ISSN 1999-5903

www.mdpi.com/journal/futureinternet

Review

Mashups: A Literature Review and Classification Framework

Brandon Beemer*, Dawn Gregg

Business School, University of Colorado, Denver, CO, USA; E-Mail: dawn.gregg@ucdenver.edu (D.G.)

* Author to whom correspondence should be addressed;
E-Mail: brandon.beemer@email.cudenver.edu; Tel.: +1-720-427-1335

Received: 19 October 2009; in revised form: 15 December 2009 / Accepted: 16 December 2009 / Published: 22 December 2009

Abstract: The evolution of the Web over the past few years has fostered the growth of a handful of new technologies (e.g. Blogs, Wiki's, Web Services). Recently web mashups have emerged as the newest Web technology and have gained lots of momentum and attention from both academic and industry communities. Current mashup literature focuses on a wide array of issues, which can be partially explained by how new the topic is. However, to date, mashup literature lacks an articulation of the different subtopics of web mashup research. This study presents a broad review of mashup literature to help frame the subtopics in mashup research.

Keywords: web mashups; end-user programming; data integration; mashup agents; enterprise mashups

---

1. Introduction

In the past five years the web has experienced a surge in growth, a phenomena described by O'Reilly [1] as the emergence of Web 2.0, a new trend for web applications that emphasizes services, participation, scalability, remixability, and collective intelligence. While some argue that the new applications emerging on the Web represents a gradual evolution of the Internet and not a new version of the Web [2], the term Web 2.0 is commonly used to refer to the current generation of social web applications being developed today. For example, the first metric used to evaluate eCommerce sites simply emphasized page views, but now eCommerce sites are evaluated by their cost per click [3]. In addition to eCommerce applications, other Web 2.0 applications include Blogs and Wikis, both of

which foster communication, collaboration, work processes, and knowledge sharing [1, 4-6]. Since Web 2.0 applications facilitate user involvement in contributing to information sources, there has been a vast increase in the amount of information and knowledge sources on the web. To foster the aggregation of differing information sources a content-syndication protocol (Really Simple Syndication: RSS feeds) was developed and enables web sites to share their content with other applications [7]. While RSS feeds and web services provide the medium for aggregating differing information sources, the latest trend in Web 2.0 research focuses on mashup applications that are designed to synthesize knowledge by semantically connecting disjointed information and knowledge sources [8].

Web mashup research is gaining a lot of momentum in both the academic and industry communities. However, to date, web mashup literature lacks an articulation of the different categories of research. To address this literary shortcoming, a methodology was developed and conducted to review web mashup literature. A review of 60 publications revealed the following six categories of mashup research: access control and cross communication, mashup integration, mashup agents, mashup frameworks, end user programming, and enterprise mashups. The remainder of this paper is structured as follows. First, the methodology used to review the literature will be discussed. The literature composing each of the 6 research categories is then discussed, including related research and foundational concepts that web mashups are built upon. Following the literature review, the characterizing attributes from each section are aggregated into an overall web mashup classification framework that can be used by researchers to frame future research, and by web developers to aid in anticipating the appropriate design attributes of mashup environments.

2. Review Methodology

Over the past few years web mashups have become a popular topic amongst both research and industry communities. However, the current posture of mashup literature is rather disjointed, which can partially be explained by the newness of this topic. A possible explanation for this is that web mashup literature lacks a thorough literature review to identify its boundaries, common issues, and subtopics. Therefore, in an effort to address this need and help frame future research, a four step methodology was developed and conducted to review mashup literature and identify its boundaries and subtopics.

The first step in the review process was to gather as many mashup related publications as possible. To do this Google Scholar was used to search for publications with the following phrases in their title: mashup, mash-up, web mashups, web mash-ups, Web 2.0 mashups. The IEEE Explorer and ACM Portal libraries were also searched with the same keywords used for Google Scholar. Additionally, the references of each web mashup publication retrieved were also reviewed to identify additional mashup publications that fell outside of our search criteria. In result, 60 web mashup publications were gathered, which are summarized in Appendix A. The second step in the review process was to identify the number of subtopics within web mashup literature. To accomplish this, all of the publications were reviewed and were grouped based on the research questions addressed, key terms describing the research, and on research methods used. This resulted in six groupings of mashup papers. The third step of the review process consisted of developing names and definitions for each grouping of

literature. The publications within each group of literature were reviewed a second time to help develop names (access control, integration, agents, frameworks, end user programming, enterprise) and definitions of each group. The final step of the review process was to ensure that each publication appropriately fit into the group it was originally placed in, after the group names and definitions were given. In this final review, 3 and 4 articles were moved from the Frameworks group to the Access Control and End-User Programming groups, respectively. Both authors discussed this change in classification and it was decided that while the 7 articles moved were framework oriented, the focus of the frameworks and subsequent research was primarily on the groups they were moved to. One interesting observation made was that of the 60 web mashup publications, 19 of them came from the International World Wide Web Conference.

3. Literature Review

Currently there's a lot of buzz surrounding web mashups in industry and academic communities. The term mashup was actually derived from the music industry, where disc jockeys remix original content from various artists to create new material [9]. Therefore the idea behind a web mashup is to synthesize new information by reusing and combining existing content from disparate information sources. Mashups allow end-users to combine information and knowledge from a plethora of sources, and integrate them into customized, goal-oriented applications [10].

Web 2.0 mashups have emerged as a trend in website design that focuses on synthesizing new content by combining data and information from multiple sources in a unique way. As such, mashups have proliferated because of how quickly they can be created, given that they're composed of pre-existing data [11-13]. For example Huynh *et al.* [48] found that all users were able to learn their mashup interface and complete a mashup within 45 minutes. The expansion of mashups can be viewed from three different perspectives. The first area of expansion is simply in the number of new mashups that are continually being created. Since their building blocks are existing data sources, building new mashups is a relatively quick process. Secondly, the number of sources being used to build mashups has also greatly expanded. Initially, many of the resources used in mashups were publicly available data (e.g. web services and RSS feeds), but now mashups are also being composed of databases, data warehouses, and even legacy systems [14, 15]. The number of mashup research domains is another area of expansion found in mashup literature, for example, Hoyer and Fischer [16] distinguish between consumer and enterprise mashups.

The remainder of this section presents the results of the literature review and categorization process described in the previous section. Table 1 contains category names and descriptions for the 6 different literary groupings of mashup literature. The first 5 categories (access control and cross communications, integration, agents, frameworks, and end user programming) represent the interrelated technological challenges common to all web mashup applications, with the 6th category covering organizational topics that are present in the domain of enterprise mashups.

3.1 Access Control and Cross Communication

The first area of mashup research found in prior literature examines issues surrounding access control to mashup data and cross communication between backend resources. Mashups connect

disparate applications and data resources to provide their services. This requires mashup applications to gain access to disparate data sources that have a variety of different sensitivity levels from RSS feeds (open access) to legacy systems (restricted access). This creates security risks for both mashup users, who have to give their credentials to mashup sites, and legacy systems managers, which must open their systems to external access.

Table 1. Mashup Research Categories.

Research Topic	Description	Related References
Access Control & Cross Communication	Mashups connect disjointed applications to provide unified services, however access control to legacy systems is difficult and hinders a mashup's potential while increasing security risks to users who have to give their credentials to mashup sites.	[17-19]
Mashup Integration	Mashups aggregate various different types of data sources (e.g. databases, legacy systems, xml, dynamic web pages, and rss feeds), this area of research addresses the data extraction obstacles presented by these different data sources.	[11, 14, 20-23]
Mashup Agents	A promising potential of mashups is the ability to semantically determine information sources that are relevant to the user, and autonomously include them in the mashup.	[8, 24-29]
Mashup Frameworks	While mashups are gaining exponential popularity, their individual applications tend to be ad-hoc, partially because of the vast differences in data sources and purpose. Researchers have identified the need for mashup frameworks, to provide developers with a set of best practices.	[4, 15, 30-43]
End User Programming	Enabling the end user to create their own custom mashups is a major reason why mashups have gained such popularity. However this presents an obstacle in that most end users do not obtain the technical expertise to develop mashups, to address researchers are developing end user programming languages and tools, to enable non-technical users to easily create mashups.	[10, 28, 44-56]
Enterprise Mashups	Organizations have identified the potential strategic advantage that could be provided by mashups in terms of business intelligence. As such business researchers are focusing on the various enterprise mashup related issues such as accountability, design principals, and intranet deployment.	[27, 57-67]

From a logistical and security perspective, the technical challenges involved in mashing legacy systems is much different than mashing multiple RSS feeds. As is the case of enterprise mashups, where the back-end resources being utilized by the mashup are legacy systems or databases, access control becomes a legitimate concern for several reasons [68]. First, it can be difficult to include back-end systems in the mashup because of the access control systems that they have in place. Often times the workaround is for the user to give the mashup their credentials and for the mashup to go on and

impersonate the user. In situations where the legacy system or database is not designed to permit provisional mashup access, the mashup receives all access. If a malicious mashup was developed and gained access to a back-end system there would be great risk to the owners and other users of the back-end system. Similarly, if a legitimate mashup was developed by a user and included the user's credentials to one or many back-end systems and the mashup was compromised by a malicious attack, the user's credentials could be leaked to many different sources. This access control "problem" has been addressed by several publications and continues to receive attention [17-19, 68].

There are three primary mechanisms being used to provide access control for mashups. The first approach is referred to as a "strawman" approach, in which a series of access checks are made to authenticate the user and then the mashup application is provided the user's credentials to allow it to access the system (e.g. acts as the user when interacting with the application). Two examples of this approach are Authsub [69] and OAuth [70]. Authsub is Google's protocol web services. Authsub requires users of web applications to complete an "access consent" page which provides the user with an authentication token. This token allows the web application to interface with Google, acting as an agent for the user. OAuth [70] is an open authentication specification that provides consumer applications requesting restricted data with an "unauthorized token" that is converted to an "authorized access token" when the user successfully logs. Once the service provider receives the authorized token it redirects the user to the consumer application where the authorized token is exchanged for an access token that contains a password. One complexity of the OAuth [70] approach is that it requires that the service maintain the state for all previously issued tokens. There are a number of downfalls to "strawman" schemes for mashup delegation [68]. First, the mashup receives all of the user's privileges to the back-end system. Since the user cannot restrict the mashup's access within the back-end system, the user must completely trust the mashup. Additionally, a comprised mashup can leak user credentials, making the user, who is completely trusting the mashup, extremely vulnerable.

One alternative to "strawman" approach is an approach that mimics real-life permit-based authorization schemes. Hasan *et al.* [68] propose a delegation permit model that allows a user to grant a mashup "delegation permits," which specify limited access rights to specific services. This approach to mashup delegation addresses two of the shortcomings of other methodologies. First, the user can restrict the scope of access available to the mashup in the back-end system. It also limits the length of time that the mashup has access to the system.

A second alternative for providing access control to mashup applications is Open ID 2.0 [19]. There are four entities in the Open ID 2.0 model: the user, mashup, claimed identification, and identity provider. First, the user visits the mashup and provides a claimed identification which specifies which identity provider they wish to use to access the mashup (e.g. users can log into many online mashups using Twitter, Facebook, Yahoo or Google credentials). The mashup redirects the user to the identity provider where the user can log in. The identity provider then redirects the user back to the mashup with cryptographic proof that the user has been authenticated and also provides the mashup with any profile information the user chooses to release. Similar to the delegation permit model [68], Open ID 2.0 allows users to control the amount of time the mashup is authorized to access the identity provider [19].

As Table 2 illustrates, access control methods can fall into one of three different categories: anonymous, full delegation, and limited delegation. Anonymous access control would be a situation

where publicly published data sources are being accessed (e.g. web services, RSS feeds). Situations where the mashup is given the user's credentials, and the backend system cannot differentiate between the user and the mashup, would be categorized as *full delegation* (e.g. [69, 70]). Finally, approaches that provide the user some control over what access the mashup has to the data sources (e.g. [19, 68] would be considered *limited delegation* because the amount of access and the length of time that it is granted to the mashup is restricted.

Table 2. Classifications for Cross Communication and Access Control.

Back End (Access Control)	Front End (Cross Communication)
Anonymous (e.g. public web services)	**Unrestricted** (e.g. using <script> tags)
Full Delegation [69, 70]	**Proxy** [71]
Limited Delegation [19, 68]	**Abstracted** [17, 18]

A fundamental feature of mashup applications is that they allow users to combine data from a variety of sources. However, when the underlying data resources are available from different providers, security issues can arise. Cross communication between multiple back end systems poses a problem in mashup development because the current generation of web browsers cannot adequately support it. Currently, browsers are designed around a "same origin" policy, where data (or code) from one source can only interact with content from its same origin. Consequently, mashup developers must choose between security and functionality, which often results in giving uncontrolled cross domain execution through the use of <script> tags and extending the browser with plug-ins for cross domain interactions [17]. This sort of cross-site scripting introduces a computer security vulnerability which might enable malicious attackers to inject client-side script into web pages viewed by other users. It can also allow one back end system complete control over another [18]. Figure 1 illustrates a warning message from a Google personal page, where a gadget requires "inlining" which provides the gadget more control over the full Google page because it is not wrapped in an "inline" frame. This allow the gadget to change aspects of the Google page and may also allow the gadget to access the user's Google account [17].

Keukelaere *et al.* [18] propose a model designed to address the cross communication issue by abstracting content from differing sources through a component based encapsulation. Instead of being proxied or using a <script> element, disparate components are loaded from their own server and then are isolated from the mashup code. "This has security advantages in (1) not requiring the component to completely trust the mashup application, and (2) making the component abstraction compatible with password anti-phishing mechanisms that use the component (DNS) domain or the certificate of the

SSL connection" [18, pg. 536]. Jackson and Wang [17] propose a similar approach that uses nested (mediating) frames to hinder direct communication from disparate web services, but because certain web development platforms are shying away from the use of frames (e.g. MS Visual Studio), this approach may pose more obstacles in the future.

Figure 1. Google Cross-communication Warning [17].

Table 2 also illustrates the three different ways that cross communication between disparate data sources can be handled in the browser: unrestricted cross communication, proxy cross communication, and abstracted cross communication. Full cross communication is when developers bypass the browser's "same origin" security policy, and use <script> tags to extending the browser with plug-ins for uncontrolled cross domain interactions [17]. Proxied cross communication is more secure than full cross communication and occurs when a web application proxy server is used to fetch the disparate content from different servers and then serves it to the mashup [71]. From a security perspective, the most desired approach to cross communication would be abstraction, where content from differing sources is abstracted through component based encapsulation [17, 18].

3.2 Mashup Integration

Web mashups offer a lot of potential to both consumers and enterprises [16]. One of the most noted advantages of mashups is that people with little technical expertise can easily build new web applications and create new forms of visualizations [38]. However, there remains a handful of technical challenges that need to be overcome before these benefits can be realized. This section looks at cross communication from a compatibility and syntactical perspective. When mashups aggregate heterogeneous data sources it becomes difficult to convert, condense, and intelligibly communicate the summarization on a common web interface [21, 23]. An example given would be combining XML with output from a legacy COBOL system. In this situation semantically predicting relevant mashups would be a challenge given that output from the latter of the two systems contains little or no metadata. While the following section covers semantic understanding of heterogeneous data types (Mashup Agents), this section covers the conversion aspect of this problem and has been labeled ‘Mashup Integration’ and includes topics such as integration, transformation, and cleansing.

Mashup integration is very similar to data integration, which has been well addressed in IS literature (e.g. [22, 72-75]). While much of the data integration work was published well before the term ‘web mashup’ was coined, this literature does provide a foundation for the problem currently being

experienced in mashup integration. Two competing data integration models are Local as View (LAV) and Global as View (GAV) [72, 75]. The LAV approach is based on the enterprise model, and states that content from each source should be characterized in terms of a view over the global schema [75]. An example of the LAV approach in practice is a data integration system that is based on the enterprise model [76]. "This approach is effective whenever the data integration system is based on a global schema that is stable and well-established in the organization" [74, pg. 235]. Therefore, the LAV approach would be suitable for enterprise mashups domains. Conversely to LAV, GAV is based on the concept that the content from each source is treated as a view which is aggregated to a global schema [74]. Each source (or view) has it's own query interface, therefore GAV would be a good integration approach for consumer mashups, that aggregate sources from multiple organizations.

The aforementioned integration approaches are good solutions for mashups that are aggregating database or data warehouse's, but may not be warranted for aggregating XML sources. Thor *et al.* [23] propose an integration framework for XML that takes a transformation approach. Since mashups rely heavily on user interaction, the user should be included in the process when multiple sources are being transformed to a summary form [23]. They provide a "fuser" that takes multiple XML documents or portions of XML documents and merges them into one, from there the user can make incremental refinements to the transformation process, until a sufficient solution is developed. Murthy *et al.* [21] present a similar transformation integration process, but add an additional emphasis on first cleaning the data before it is transformed.

While data integration literature provides a good foundation for web mashups, much of this literature is limited to homogeneous integrations. For example, the LAV and GAV integration approaches discussed previously focus on coupling databases together. Even if the databases being integrated are different (e.g. Oracle and SQL Server), this is still considered a homogenous integration because the systems being coupled are the same type. The same is true with the XML integration methods presented by Murthy *et al.* [21]. One area that integration literature needs to be extended for mashup domains is in the integration of heterogeneous data sources. This need is extremely prevalent in enterprise mashup domains, where mashup's integrate organizational legacy systems and databases with publicly available information sources like web services and RSS feeds. Sneed [14] approaches heterogeneous data integration by developing a framework that uses a Service Oriented Architecture (SOA) to convert legacy systems into XML and make them available as web services. Similarly, Thor *et al.* [23] take a similar approach that utilizes XML wrappers to bring heterogeneous data sources to a common platform. In fact, both Sneed [14] and Thor *et al.* [23] share common ground with, and could benefit from derivation of, the Enterprise Application Integration (EAI) body of knowledge. EAI incorporates programming across multiple enterprise applications, provides workflow languages and messaging, and gateways to common enterprise applications (e.g. databases, application servers, and web servers), which is similar to SOA approaches discussed in literature [22].

As illustrated by Table 3, integration literature can be viewed from three different perspectives: homogenous integration, heterogeneous integration, and application (process) integration. Homogeneous integration research focuses on coupling similar data from different sources, two examples would be enterprise mashup's coupling of legacy systems and consumer mashup's coupling of web services. Heterogeneous integration research addresses domains that couple dissimilar data from different sources, an example would be an enterprise mashup that couples legacy systems and

public web services. Application integration focuses on coupling disparate business processes in enterprise mashups through methodologies such as SOA.

Table 3. Classifications for Mashup Integrations.

Mashup Integration
Homogeneous
[11, 74, 76]
Heterogeneous
[21, 23]
Application (Process)
[14, 22]

3.3 Mashup Agents

A large number of mashup environments are being designed for end users that lack significant technical expertise (e.g. [47-49, 51]), which increases the need for tools that support the mashup creation process. These tools generally include algorithms, web crawlers, and other technologies that are designed to go out and find potentially relevant sources of information. All of these tools can be classified as 'Mashup Agents', even though this term was not used in any of the publications that were reviewed.

Web services are a common ingredient in web mashups. Much work has been done on web service composition including industry efforts on standard composition languages such as BPEL [77]. However, in terms of mashups, web service composition techniques are not suitable as they focus on describing messaging and synchronization processes; whereas mashups involve data integration and require content descriptions rather than process descriptions [28]. To address this shortcoming, Tatemura *et al.* [28] applied a machine learning based approach, and designed an agent based framework that continually observes web service feeds over time to learn the patterns, in order to provide a semantic mapping of the web service's content. In this approach, Artificial Intelligence (AI) is used to determine the semantic meaning of the potential mashup ingredient itself. Another approach is to utilize AI from the user's perspective, to predict and recommend potential mashup ingredients that they may deem useful. Wang *et al.* [29] do this by developing a Bayesian network to build user profiles as the representatives of usage patterns. A Bayesian network is a decision tree where each alternative outcome is assigned a probability, so in this case, each potential mashup ingredient has a probability of relevance, based on the user's previous browsing patterns. Natural Language Processing (NLP) is another AI technique that has also been found to be an effective means for web service semantic discovery. Blake and Nowlan [8] implemented such an approach and developed an agent that performed pair wise comparisons of web services based on the commonalties between the service's

inputs and outputs. An empirical evaluation of their agent showed that among the top 6 messages for mashup predictions were State, City, and Zip, which combined had a 29% prediction rate for valid mashups. In fact location based mashups could be considered a mashup sub-category all on its own (e.g. [26, 29, 34, 45]).

As illustrated by Table 4, mashup agents can fall into one of two basic categories, server based agents and client based agents. Server based agents are tools that reside on a web server like Microsoft Popfly or Yahoo Pipes, whereas client based agents are plugins that are installed in the user's browser. Server based agents is the most popular of the two in industry and are widespread in academia too. For instance, Ikeda *et al.* [25] developed a mashup development framework that provides a server based data management engine that enables the developer to identify semantic relationships, and then to browse based on semantic relevance. Similarly, Lu *et al.* [78] present a semantic based agent that identifies similar API's and then enables users to build server based mashups.

That being said, client based agents have several advantages over server based agents which include but are not limited to data access, privacy, performance, and user experience [24]. An ongoing research project titled MashMaker is an example that demonstrates the benefit of client based mashup agents [24, 46, 47]. "MashMaker can see everything the browser can see, including local files, information on the intranet, information requiring a login, and active content generated by Javascript" [24, pg. 29], and therefore provides performance and privacy advantages that could not be realized by a server based agent, since it doesn't need to transmit data to a central server in order to create the user interface [47].

Table 4. Mashup Agent Classifiers.

Induction	Orientation
Machine Learning [28]	**Server Based** [25, 78]
Bayesian Networks [29]	**Client based** [24, 46, 47]
Natural Language Processing [8]	

There are 2 classifiers that can be consistently used to categorize mashup agent literature, induction method and orientation. These are illustrated in Table 4. Despite the newness of web mashup research, we've already seen the application of 3 separate AI technologies applied to mashup agents for semantic induction: machine learning, Bayesian networks, and natural language processing [8, 28, 29]. This begs the question, "What other AI technologies might be used to enhance the capacity and effectiveness of web mashup agents?", which is the first area of mashup agent research. The second

and third areas of mashup agent research are server and client based agents respectively. As noted earlier both have their advantages. Server based agents are more widespread in industry, and thus may be easier for initial adoption. However client based agents offer many advantages over server based agents because they don't require interactive communication with a hosting server, and have access to local based resources that are not available to server based agents.

3.4 Mashup Frameworks

This section discusses mashups from an aggregated scope, in review of frameworks that address the different attributes of mashups collectively. Frameworks are extremely important in the initial phases of a technology's lifecycle as they provide an initial foundation that can be applied by early adopters in industry and that can be evaluated (or extended) by researchers in academia. Since mashups are designed by end users for specific contexts with specific purposes, they tend to be very ad hoc in nature and span a wide variety of domains. As such, many frameworks have been developed to help designers develop mashups for the different domains.

Mostarda and Palmisano [40] present a mashup framework that features a hybrid scripting language that supports unification as it is based on the type morphing paradigm. Much like inheritance in object oriented environments, type morphing is the ability of the language to cast any primitive type to another one where needed. The transformation of data across heterogeneous types is done by following a set of predefined rules. Through an interactive interface, users can iteratively overload (or mash) model elements until a sufficient solution is developed. Vancea *et al.* [15] also use an object-oriented data model for the exchange of data between the disparate data models. Furthermore they argue that current web mashup frameworks lack sufficient data models to handle data interchange and propose "a database-driven approach to web mashups that supports integration at the database level and enables mashup developers to work with a uniform abstract model and have direct access to powerful features of database systems" [15, pg. 162]. Abiteboul *et al.* [30] present a mashup model that quantifies the different roles of mashups which are: query data sources, import other mashups, use external Web services, and specify complex interaction patterns between its components. Their model is derived from semantic web, and various object oriented concepts like inheritance.

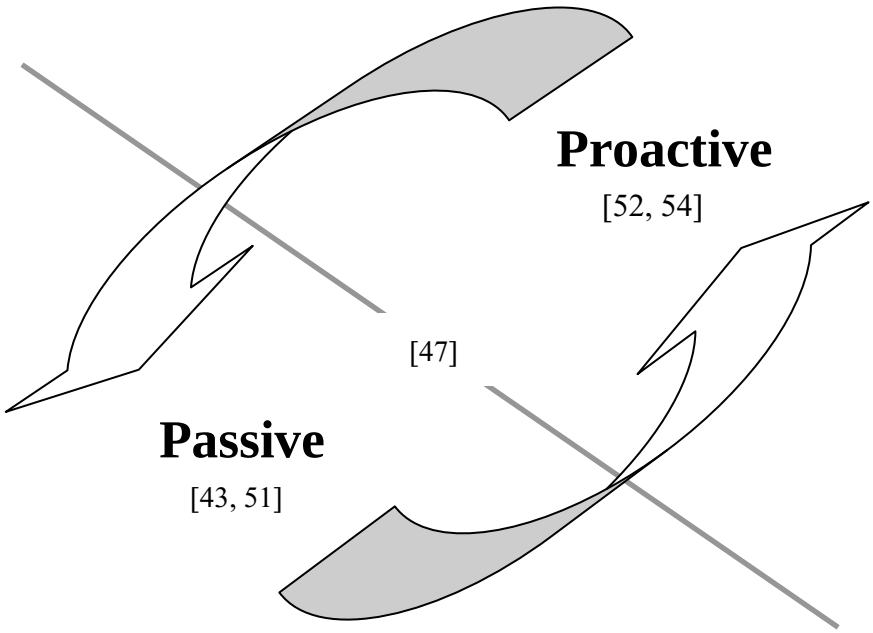
One concept that is commonly discussed in mashup frameworks is interactive (or iterative) processes. This is especially true in the 'End User Programming' literature and will be discussed in the following section, and can be explained by the fact that multiple mashup variations are frequently considered before an acceptable solution is discovered. Cetin *et al.* [32] present a framework for migrating legacy systems toward service-oriented mashups. In their framework, they place emphasis on modeling the business requirement up front and then analyzing the existing legacy systems to see if such functionality currently exists and can be implemented into the mashup. "The iterative mapping process is as follows: (a) if a business requirement can be satisfied by one of the existing legacy components, then simply wrap it by considering the QoS attributes; (b) if there is a gap with the existing legacy component and requirement, and the gap can be filled during service wrapping, then accustom the legacy component into a new service; (c) if the gap cannot be fulfilled, then develop a new service for the requirement" [32, pg. 171].

3.5 End User Programming

Another challenge being addressed by mashup researchers is how to seamlessly package mashup technologies in such a way that non-technical users can easily and effectively create mashup application in a myriad of different domains. As represented by the attention given to this topic in literature, developing mashup environments that are conducive to non-technical end user development is one of the more difficult challenges. Of the 60 publications that were reviewed in Appendix A, 25% (15) can be classified as 'End User Programming'.

Two different approaches are commonly taken in addressing (enabling) end user mashup development. The first approach is passive by nature and focuses on designing plugins that work with the user's current browser, observe what the user is viewing, to suggest related sources for potential mashing (e.g. [46, 47, 51, 53]). There are several benefits to extending the user's current browser for the mashup process. First, the mashup process is very close to the user's web browsing experience, if the user encounters data that they would like to manipulate, the user would not have to launch a separate program to begin mashing [56]. Secondly, as mentioned previously in the client based mashup agent discussion, it makes it easier to access local data on the user's machine that the browser normally has access to. Lastly, installing a browser plug-in is easier than installing a separate application, and thus may foster use [56].

Figure 2. Classifications of Web Mashup User Interfaces.



A second approach to end user mashup development is proactive by nature and is necessary when the mashup process becomes more complicated (e.g. process modeling or advanced interface integration) [52]. Tuchinda *et al.* [54] present a tool that enables users to develop mashups in complicated integration domains by first providing examples of what the end mashup should look like; the tool then aims to mimic the format of the end result, allowing for mashups to be developed by end users who don't have programming experience. Tatemura *et al.* [13], take a similar 'by example'

approach in developing a tool that allows users to mashup disparate data sources by creating abstracted target schemas that are populated based on examples provided by the user. Mashroom is another end user mashing application that is based on the nested relational model, and allows users to iteratively construct mashups through continuous refinement [55].

As mentioned previously, end user programming literature seems to generally fall into one of two categories, passive or proactive. Passive approaches piggyback onto the user's current web browsing experience and function as browser plugins, and proactive approaches are executed from a separate application and are designed for more complicated mashup domains. Figure 2 illustrates a third approach that is beginning to appear in literature that is a middle ground to the passive and proactive approaches (e.g. [47]). In this scenario, the user's browsing experience begins as normal with the mashup process being passive. The browser has a plugin installed that observes the user's browsing patterns. Next, the plugin activates a server based inference engine (e.g. Google Suggest) that suggests potential mashup sources in a side bar. The user can continue browsing or at any point begin creating a mashup.

3.6 Enterprise Mashups

Business researchers have identified an emerging opportunity for organizations to use technology to exploit information from desktops, the web, and other non-traditional enterprise sources, in order to react to situational business needs [65]. This tract of research has been labeled as 'Enterprise Mashups', and has been distinguished from consumer mashups because there are a plethora of legal and accountability related issues (e.g. security, availability, quality) that are specific to the organizational domain [16]. Similar to end user programming, enterprise mashups have received a significant amount of attention in mashup literature, of the 60 publications reviewed for this study, 20% (12) were classified as enterprise mashups.

Two areas of existing enterprise related research that support enterprise mashup domains are Service-oriented Architecture and Enterprise Information Integration [27, 61]. Service-oriented Architectures (SOA) are based on semantic web services and have widely been considered a key technology for achieving business-to-business integration within corporate intranets [79]. However, as Web 2.0 concepts are being applied to enterprise domains (e.g. enterprises mashups) the need to offer a user-centered focus to improve business productivity and innovation has been revealed [80]. This user-centered approach is beyond the functionality of traditional B2B SOA's [27, 61]. Therefore, a big focus in enterprise mashup related research is on developing tools and frameworks to extend SOA's into mashup-able applications.

Lizcano *et al.* [27] highlight the shortcoming of SOA's in mashup domains and present the two proof of concept research projects (FAST and EzWeb) for enterprise 2.0 mashups. DAMIA is another research project that focuses on enabling the creation of enterprise mashups that combine data from desktop, web, and organizational IT sources into feeds that can be utilized by user created web applications [57, 65]. Siebeck *et al.* [64] highlighted the advantages of using cloud computing infrastructures as a platform B2B integration mashups. As indicated by the investments in research, organizations are beginning to see the potential impact that enterprise mashups could have on competitive advantage.

4. Mashup Classification Framework

Mashups tend to be ad hoc in nature, as they are created by end users for specific problems. It could be suggested that no two mashups are alike, each being designed to fulfill a very specific need. This presents a unique challenge for researchers seeking to understand mashups and for developers seeking to provide tools to support this diverse domain. Every mashup could potentially have its own framework tailored particularly to the attributes of the domain it's developed in. However, this would be of little use to researchers or developers. Instead, this study seeks to find similarities amongst the differences and to provide system designers with a framework that outlines the major design characteristics of mashup development projects. The Mashup Classification Framework presented in Table 5 synthesizes the primary attributes of mashup applications identified during the literature review. It provides an overview of mashup design options that can be used by practitioners seeking to define their mashup architecture.

Table 5. Mashup Classification Framework.

Access	Communication	Integration	Induction	Orientation	Application	Interface
Anonymous	Unrestricted	Homogeneous	Machine Learning	Server Based	Consumer	Passive
Full Delegation	Proxied	Heterogeneous	Bayesian	Client Based	Enterprise	Proactive
Limited Delegation	Abstracted	Application	Natural Language Processing			
			Other form of AI?			

In addition to being a tool that supports mashup design, the Mashup Classification Framework can also benefit the academic community as well. First, it provides researchers a comprehensive scope of the technologies that are involved in mashup development. Second, by breaking down the technologies involved in web mashups, the framework provides insight into other bodies of literature that mashups can be built upon and derived from (e.g. Mashup Integration → Data Integration; Mashup Agents → AI; Mashup Interfaces → Human-Computer Interaction). Finally, in the same way that practitioners can use this framework to identify system attributes, researchers can use it to identify and focus their research on various aspects of mashups, and the interactions between them. To provide an example of this, 8 mashup applications have been selected from the literature review and are classified in terms of the mashup classification framework in Table 6.

5. Future Research

Over the past few years web mashups have generated buzz in research communities. However, as with any new technology, there remains many unanswered questions. In terms of access control, researchers are beginning to tackle the problem of proper access control for mashups, and differentiating them from actual users. Much work remains in this area, especially in regards to

enabling legacy systems, to differentiate between actual users and mashup agents. Similar security related issues remain in managing cross communication between disparate data sources in web browsers. As web 2.0 technologies have advanced, the old "same origin" security policy still used by web browsers today is no longer sufficient. With promising success, researchers have begun developing delegation frameworks to work with existing browser (e.g. [18, 68]). However, another approach that could be investigated would be to develop a next generation of web browsers that are more suited for the security and integration issues prevalent in web 2.0 technologies.

Table 6. Mashup Classification Framework (applied to literature).

Author	Access	Comm.	Integration	Induction	Orientation	Application	Interface
[54]	Anonymous	Unrestricted	Homogeneous	N/A - mobile mashup uses proximity	Server	Consumer	Proactive
[9]	Full Delegation	Abstracted	Homogeneous	Semantic Decision Tree	Server	Enterprise	Proactive
[26]	Limited Delegation	Proxied	Heterogeneous	N/A – as presented, study focuses on access control	Server	Consumer	Proactive
[78]	Anonymous	Abstracted	Homogeneous	Pair-wise Comparisons	Client	Consumer	Passive
[39]	Limited Delegation	Abstracted	Heterogeneous	N/A – as presented, study focuses on access control	Client	Consumer	Passive
[68]	Limited Delegation	Unrestricted	Homogeneous	Machine Learning	Server	Enterprise	Proactive
[70]	Full Delegation	Abstracted	Heterogeneous	Metadata and User Iterations	Server	Consumer	Proactive
[36]	Limited Delegation	Abstracted	Heterogeneous	N/A – as presented, study focuses on access control	Client	Consumer	Proactive

To date, the majority of web mashup literature focuses on extending the capabilities and functionality of this new technology. Unlike other, more mature, IS research topics, web mashup literature lacks theoretical application (e.g. Trust, Perceived Usefulness, Computer Self-Efficacy). Computer Self-Efficacy likely has a big role in the successful application of web mashup infrastructures, because generally speaking, they are aimed towards end users who are developing applications, and lack advanced technical experience. Additionally, Trust and Personal Innovativeness would likely be applicable constructs to web mashups being applied to the domain of decision support (e.g. [81]). That being said, one observation that should be noted is the common inference that the user's mashup process is iterative (e.g. [28, 47, 54, 56]). For example, Huynh *et al.* [48, pg. 13] state that they believe, "the users actually work iteratively on data, switching from aligning and clean up the data to using the data, and back, as they get to know the data better over time". This is important to mention because none of the literature that was reviewed investigated this aspect of the mashup

process directly, but rather, only discussed it peripherally, which suggests a gap in the literature. Another opportunity for future research would be to apply a clustering algorithm, such as k-means, on the reviewed papers to see how each relates to the proposed categorization of Table 1.

6. Conclusions

Web mashups are an interesting and exciting web 2.0 application that are receiving lots of attention from both practitioners and researchers a like. The concept of web mashups was derived from the music industry where DJ's integrate and mix multiple tracks into a new track. While conceptually, web mashups are similar, they are extremely more complicated than mixing music, as they are a way to create new web applications by combining existing resources, data and APIs [82, 83]. This research focused on synthesizing the information available on mashups so that researchers and practitioners can better understand the scope of and challenges associated with this emerging research domain. This study revealed five common themes found across multiple research studies. First, because mashups aggregate content from disparate information systems access control is legitimate concern, especially in domains involving legacy systems which are not designed for delegated access control (e.g. [19, 68]). Additionally, cross communication is another security challenge in today's web browsers that operate in a "same origin" security policy (e.g. [17, 18]). Second, integration was identified as a significant technical challenge encountered when developing mashups. While integration has been addressed in IS literature, web mashups present new challenges like heterogeneous (e.g. mashing a web service with a legacy system) and application (e.g. business process) integration [11, 14, 20-22]. A third issue is related to information overload because there are simply too many information sources for individuals to process when selecting the best possible mashup resources. To address this problem researchers are applying artificial intelligence methods to mashup agents that go out to the web and retrieve appropriate mashup ingredients. Prior research has identified machine learning, Bayesian networks, and natural language processing as induction methods that can be used by mashup agents (e.g. [8, 28, 29]), but other methods like neural networks could also prove useful. A fourth issue identified during the literature review was the ability of end users to create custom mashup applications. This research focused on the alternatives available for user interfaces, which can utilize a passive or proactive mashup approach (e.g. [46, 47, 51, 53, 56]). Finally, this research identified a set of issues, including security, availability, and quality issues, that are unique to enterprise mashups (e.g. [27, 57, 60, 64, 65, 84]). The primary contribution of this study is the development of a mashup classification framework that is rooted in prior research but extends that research to provide a tool to for designing and classifying mashups. This framework can be used by researchers and practitioners a like as mashups evolve to solve even more complex problems.

References

1. O'Reilly, T. What-is-web-2.0. Available at: http://oreilly.com/web2/archive/what-is-web-20.html, Accessed September 15, 2009.
2. 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://www-128.ibm.com/developerworks/podcast/dwi/cm-int082206.txt, Accessed December 16, 2009.

3. Fain, D.C.; Pedersen, J.O. Sponsored Search: A Brief History. Proceedings of the Second Workshop on Sponsored Search Auctions, Edinburgh, Scotland, May 2006.
4. Cayzer, S. Semantic Blogging and Decentralized Knowledge Management. *Comm. of the ACM* **2006**, *47*, 47-52.
5. Dearstyne, B. Blogs, Mashups, & Wikis: Oh, My!, *Inf. Management J.* **2007** *41*, 24-32.
6. Hester, A. Analysis of Factors Influencing Adoption and Usage of Knowledge Management Systems and Investigation of Wiki Technology as an Innovative Alternative to Traditional Systems. Doctoral Dissertation, University of Colorado at Denver 2009.
7. Cold, J. Using Really Simple Syndication (RSS) to Enhance Student Research. *SIGITE Newsletter* **2006**, *3*, 6-9.
8. Blake, M.B.; Nowlan, M.F. Predicting Service Mashup Candidates Using Enhanced Syntactical Message Management. IEEE International Conference on Services Computing, Chicago, Illinois, June 2008; 229-236.
9. Murugesan, S. Understanding Web 2.0. *IT Prof.* **2007**, *9*, 34-41.
10. Albinola, M.; Baresi, L.; Carcano, M.; Guinea S. Mashlight: A Lightweight Mashup Framework for Everyone. International World Wide Web Conference, Madrid, Spain, April 2009.
11. Hornung, T.; Simon, K.; Lausen, G. Mashing Up the Deep Web. International Conference on Web Information Systems and Technologies, Funchal, Portugal, May 2008; 58–66.
12. Merrill, D. Mashups: The New Breed of Web App. IBM Web Architecture Technical Library 2006.
13. Tatemura, J.; Chen, S.; Liao, F.; Po, O.; Candan, K.; Agrawal, D. UQBE: Uncertain Query by Example for Web Service Mashup. International Conference on Management of Data, Vancouver, Canada, May 2008; 1275-1280.
14. Sneed, H. Integrating Legacy Software into a Service Oriented Architecture. Proceedings of the Conference on Software Maintenance and Reengineering, Indianapolis, Indiana, June 2006.
15. Vancea, A.; Grossniklaus, M.; Norrie, M. Database-Driven Web Mashups. International Conference On Web Engineering, Yorktown Heights, New York, July 2008; 162-174.
16. Hoyer, V.; Fischer, M. Market Overview of Enterprise Mashup Tools. International Conference on Service Oriented Computing, Sydney, Australia, December 2008; 708-721.
17. Jackson, C.; Wang, H. Subspace: Secure Cross-Domain Communication for Web Mashups. International World Wide Web Conference, Alberta, Canada, May 2007.
18. Keukelaere, F.G.; Bhola, S.; Steiner, M.; Chari, S.; Yoshihama, S. SMash: Secure Component Model for Cross-Domain Mashups on Unmodified Browsers. International World Wide Web Conference, Beijing, China, April 2008; 535-544.
19. Recordon, D.; Reed, D. OpenID 2.0: A Platform for User-Centric identity Management. *J. of Comp. Sec.* **2007**, *15*, 11-15.
20. Blau, B.; Lamparter, S.; Haak, S. Remash! Blueprints for RESTful Situational Web Applications. International World Wide Web Conference, Madrid, Spain, May 2009.
21. Murthy, S.; Maier, D.; Delcambre, L. Mash-o-matic. ACM Symposium on Document Engineering, Amsterdam, Netherlands, October 2006; 205-214.

22. Stonebraker, M.; Hellerstein, J.M. Content Integration for e-Business. International Conference on Management of Data, Santa Barbara, California, May 2001; 552-560.
23. Thor, A.; Aumueller, D.; Rahm, E. Data Integration Support for Mashups. Proceeding of the International Workshop on Information Integration on the Web, Rio de Janeiro, Brazil, April 2007; 104–109.
24. Ennals, R.; Brewer, E.; Garofalakis, M.; Shadle, M.; Gandi, P. Intel Mash Maker: Join the Web. SIGMOD, Beijing, China, June 2007; 27-33.
25. Ikeda, S.; Nagamine T.; Kamada, T. Application Framework with Demand-Driven Mashup for Selective Browsing. International Conference on Information Integration and Web-based Applications and Services, Linz, Austria, 2008; 33-40.
26. Kukka, H.; Ojala, T.; Tiensyrja, J.; Mikkonen, T. panOULU Luotsi: A Location Based Information Mash-up with XML Aggregator and WiFi Positioning. International Conference on Mobile and Ubiquitous Multimedia, Umea, Sweden, December 2008; 80-83.
27. Lizcano, D.; Soriano, J.; Reyes, M.; Hierro, J. EzWeb/FAST: Reporting on a Successful Mashup-based Solution for Developing and Deploying Composite Applications in the Upcoming Web of Services. International Conference on Information Integration and Web-based Applications and Services, Linz, Austria, 2008; 15-24.
28. Tatemura, J.; Sawires, A.; Po, O.; Chen, S.; Candan, K.; Argrawal, D.; Goveas, M. Mashup Feeds: Continuous Queries over Web Services. SIGMOD, Beijing, China, June 2007; 1128-1120.
29. Wang, W.; Zeng, G.; Zhang, D.; Huang, Y.; Qiu, Y.; Wang, X. An Intelligent Ontology and Bayesian Network Based Semantic Mashup for Tourism. IEEE Congress on Services, Honolulu, Hawaii, June 2008; 128-135.
30. Abiteboul, S.; Greenshpan, O.; Milo, T. Modeling the Mashup Space. Workshop On Web Information and Data Management, Napa Valley, California, October 2008; 87-94.
31. Ankolekar, A.; Krotzsch, M.; Tran, T.; Vrandecic, D. Two Cultures: Mash Up Web 2.0 and the Semantic Web. International World Wide Web Conference, Alberta, Canada, May 2007; 825-834.
32. Cetin, S.; Altintas, N.; Oguztuzun, H.; Dogru, A.; Tufekci, O.; Suloglu, S. A Mashup-Based Strategy for Migration to Service-Oriented Computing. Proceedings of the IEEE International Conference on Pervasive Services 2007; 169–172.
33. Chen, H.; Ikeuchi, N. Implementation of Ubiquitous Personal Study Using Web 2.0 Mash-up and OSS Technologies. International Conference on Advanced Information Networking and Applications 2008; 1573-1578.
34. Diego, L.; Vazquez, J.; Abaitua, J. A Web 2.0 Platform to Enable Context-Aware Mobile Mash-Ups. Proceedings of Conference on Ambient Intelligence, Sophia Antipolis, France, September 2007; 266-286.
35. Gendarmi, D.; Lanubile, F. A Web Mashup for Social Libraries. International World Wide Web Conference, Madrid, Spain, April 2009.
36. Hartmann, B.; Doorley, S.; Klemmer, S. Hacking, Mashing, Gluing: A Study of Opportunistic Design and Development. Technical Report 2006-14, Stanford HCI Group 2006.
37. Iwazume, M.; Kaneiwa, K.; Zettsu, K.; Nakanishi, T.; Kidawara, Y.; Kiyoki, Y. KC3 Browser: Semantic Mash-up and Link-Free Browser. International World Wide Web Conference, Beijing, China, April 2008; 1209-1210.

38. Koschmider, A.; Torres, V.; Pelechano, V. Elucidating the Mashup Hype: Definitions, Challenges, Methodical Guide and Tools for Mashups. International World Wide Web Conference, Madrid, Spain, April 2009.
39. Le-Phuoc, D.; Polleres, A.; Hauswirth, M.; Tummarello, G.; Morbidoni, C. Rapid Prototyping of Semantic Mash-ups through Semantic Web Pipes. International World Wide Web Conference, Madrid, Spain, April 2009.
40. Mostarda, M.; Palmisano, D. MU: An Hybrid Language for Web Mashups. International World Wide Web Conference, Madrid, Spain, April 2009.
41. Raza, M.; Hussain, F.; Chang, E. A Methodology for Quality-based Mashup of Data Source. International Conference on Information Integration and Web-based Applications and Services, Linz, Austria, November 2008; 528-533.
42. Tatsubori, M. Towards an Advertising Business Model for Web Service Mashups. International World Wide Web Conference, Madrid, Spain, April 2009.
43. Wong, J.; Hong, J. What Do We "Mashup" When We Make Mashups? International Conference on Software Engineering, Leipzig, Germany, May 2008; 35-39.
44. Back, G.; Bailey, A. Increasing the Visibility of Web-Based Information Systems via Client-Side Mash-Ups. International Conference on Digital Libraries, Prague, Czech Republic, August 2008; 418-418.
45. Brandt, J.; Klemmer, S. Lash-Ups: A Toolkit for Location-Aware Mash-Ups. Proceedings of the 19th annual ACM Symposium on User Interface Software and Technology, Montreux, Switzerland, October 2006; 79-80.
46. Ennals, R.; Garofalakis, M. Mashmaker: Mashups for the Masses. International Conference on Management of Data, Beijing, China, June 2007; 1116-1118.
47. Ennals, R.; Gay, D. User-Friendly Functional Programming for Web Mashups. International Conference on Functional Programming, Freiburg, Germany, October 2007; 223-234.
48. Huynh, D.; Miller, R.; Karger, D. Potluck: Data Mash-up Tool for Casual Users. International Conference on World Wide Web, Alberta, Canada, May 2007; 737–746.
49. Jones, M.; Churchill, E.; Twidale, M. Mashing up Visual Languages and Web Mash-ups. Hawaii International Conference on System Sciences, Waikoloa, Hawaii, January 2008; 143-146.
50. Kongdenfha, W.; Benatallah, B.; Vayssiere, J.; Saint-Paul, R.; Casati, F. Rapid Development of Spreadsheet-Based Web Mashups. International World Wide Web Conference, Madrid, Spain, April 2009.
51. Lin, J.; Wong, J.; Nichols, J.; Cypher, A.; Lau, T.A. End-User Programming of Mashups With Vegemite. International Conference on Intelligent User Interfaces, Sanibel Island, Florida, February 2009; 97-106.
52. Nestler, T. Towards a mashup-driven end-user programming of SOA-based applications. International Conference on Information Integration and Web-based Applications and Services, Linz, Austria, November 2008; 551-554.
53. Sabbouh, M.; Higginson, J.; Semy, S.; Gagne, D. Web Mashup Scripting Language. International World Wide Web Conference, Beijing, China, June 2007; 1305-1306.
54. Tuchinda, R.; Szekely, P.; Knoblock, C.A. Building Mashups By Example. International Conference on Intelligent User Interfaces, Canary Islands, Spain, January 2008; 139-148.

55. Wang, G.; Yang, S.; Han, Y. Mashroom: End-User Mashup Programming Using Nested Tables. International World Wide Web Conference, Madrid, Spain, April 2009.
56. Wong, J.; Hong, J. Making Mashups with Marmite: Towards End-user programming for the Web. Conference on Human Factors in Computing Systems, San Jose, California, April 2007; 1435-1444.
57. Altinel, M.; Brown, P.; Cline, S.; Kartha, R.; Louie, E.; Markl, V.; Mau, L.; Ng, Y.; Simmen, D.; Singh, A. Damia: A Data Mashup Fabric For Intranet Applications. Conference on Very Large Data Bases, Vienna, Austria, September 2007; 1370–1373.
58. Guinard, D.; Trifa, V.; Pham, T.; Liechti, O. Towards Physical Mashups in the Web of Things. Proceedings of the 6th International Conference on Networked Sensing Systems, Pittsburgh, Pennsylvania, June 2009.
59. Hoyer, V.; Gilles, F.; Fleischmann, K.; Dreiling, A.; Stanoesvka-Slabeva, K. SAP Research Roof Top Marketplace. International World Wide Web Conference, Madrid, Spain, April 2009.
60. Hoyer, V.; Stanoesvka-Slabeva, K.; Janner, T.; Schroth, C. Enterprise Mashups: Design Principles towards the Long Tail of User Needs. International Conference on Services Computing, Honolulu, Hawaii, July 2008; 601-602.
61. Jhingran, A. Enterprise Information Mashups: Integrating Information, Simply. Conference on Very Large Data Bases, Auckland, New Zealand, July 2008; 3-4.
62. Majer, F.; Nussbaumer, M.; Freudenstein, P. Operational Challenges and Solutions for Mashups - An Experience Report. International World Wide Web Conference, Madrid, Spain, April 2009.
63. Makki, S.; Sangtani, J. Data Mashups and Their Applications in Enterprises. International Conference on Internet and Web Applications and Services, Athens, Greece, June 2008; 445-450.
64. Siebeck, R.G.; Janner, T.; Schroth, C. Cloud-based Enterprise Mashup Integration Services for B2B Scenarios. International World Wide Web Conference, Madrid, Spain, April 2009.
65. Simmen, D.; Altinel, M.; Markl, V.; Padmanabhan, S.; Singh, A. Damia: Data Mashups for Intranet Applications. International Conference on Management of Data, Vancouver, Canada, June 2008; 1171-1182.
66. Yang, F. Enterprise Mashup Composite Service in SOA – User Profile Use Case. IEEE Congress on Services, Honolulu, Hawaii, June 2008; 97-98.
67. Zou, J. Towards Accountable Enterprise Mashup Services. IEEE International Conference on e-Business Engineering, Xi'an, China, October 2008; 205-212.
68. Hasan, R.; Winslett, M.; Conlan, R.; Slesinsky, B.; Ramani, N. Please Permit Me: Stateless Delegated Authorization in Mashups. Proceedings of the Computer Security Applications Conference, Anaheim, California, December 2008; 173-182.
69. AuthSub, Google account authentication. Available at: http://code.google.com/apis/accounts/AuthForWebApps.html, Accessed September 15, 2009.
70. OAuth, Specification 2.0. Available at: http://oauth.net/, Accessed September 15, 2009.
71. Johnson, A.D.; Handsaker, R.E.; Pulit, S.L; Nizzari, M.M, O'Donnell, C.J.; Bakker P. SNAP: A Web-based Tool For Identification and Annotation of Proxy SNPs Using HapMap. *Bioinf. Adv. Access, **2008,** 24*, 2938-2939.
72. Halevy, A.Y. Answering Queries Using Views: A survey. Conference on Very Large Databases, Rome, Italy, September 2001; 270–294.

73. Hull, R. Managing Semantic Heterogeneity in Databases: A Theoretical Perspective. Symposium on Principles of Database Systems, Tucson, Arizona, May 1997.
74. Lenzerini, M. Data Integration: A Theoretical Perspective. Symposium on Principles of Database Systems, Madison, Wisconsin, June 2002; 233-246.
75. Ullman, J.D. Information Integration Using Logical Views. International Conference on Database Theory, Delphi, Greece, January 1997; 19–40.
76. Gruninger, M.; Lee, J. Ontology Applications and Design. *Com. of the ACM* **2002**, *45*, 39–41.
77. Milanovic, N.; Malek, M. Current Solutions for Web Service Composition. *IEEE Int. Comp.* **2004**, *8*, 51–59.
78. Lu, B.; Wu, Z.; Ni, Y.; Xie, G.; Zhou, C.; Chen, H. SMash: Semantic-Based Mashup Navigation For Data API Network. International World Wide Web Conference, Madrid, Spain, April 2009; 1133-1134.
79. MacKenzie, M. OASIS - Reference Model for Service Oriented Architecture 1.0. http://www.oasisopen.org/committees/tc_home.php?wg_abbrev=soa-rm, Accessed September 15, 2009.
80. O'Reilly, T.; Musser, J. Web 2.0 Principles and Best Practices. O'Reilly Radar, Accessed September 15, 2009.
81. Walczak, S.; Kellog, D.L.; Gregg, D.G. A Web 2.0 Application to Support Consumer Decision-Making in Multi-Criteria Environments. *Int. J. of I.S. in the Serv. Sect.* **2009**, submitted.
82. Benslimane, D.; Dustdar, S.; Sheth, A. Services Mashups, The New Generation of Web Applications. *IEEE Int. Comp.* **2008**, *12*, 13-15.
83. Yu, J.; Benatallah, B.; Casati, F.; Daniel, F. Understanding Mashup Development. *IEEE Int. Comp.* **2008**, *12*, 44-52.
84. Hoyer, V.; Stanoesvka-Slabeva, K. Towards a Reference Model for grassroots Enterprise Mashup Environments. 17th European Conference on Information Systems, Verona, Itay, June 2009.

Appendix A. Literature Classification Table.

Source	Title	Synopsis	Category
[58]	Please Permit Me: Stateless Delegated Authorization in Mashups.	Reviews current mashup access frameworks and discusses their limitations and vulnerabilities, then presents a new access control framework for mashups.	Access Control & Cross Com.
[19]	OpenID 2.0: A Platform for User-Centric Identity Management.	Present a proxy-like methodology to provide password protected connections between Identity Providers and Relaying Parties, such as mashups.	Access Control & Cross Com.
[18]	Smash: Secure Component Model for Cross-Domain Mashups on Unmodified Browsers.	The security policies of the current generation of web browser prohibits content from different sources to interact, an action required by mashups. A communication abstraction technique is presented, to provide a secure platform for mashups to blend content from differing sources.	Access Control & Cross Com.
[17]	Secure Cross-Domain Communications for Web Mashups.	The current web security models states that services can only manipulate data that is from it's same origin. This is a problem for mashup implementations where data from multiple source need to be blended (e.g. aggregated or calculated). Authors propose a mediating frame approach where data from multiple source can be placed and manipulated by the mashup.	Access Control & Cross Com.
[20]	Remash! Blueprints for RESTful Situational Web Applications	A system is presented that is designed to harness collective intelligence to support end-user development of service mashups. It enables developers to specify policies about their ingrediential incompatibilities, and then mash services based on their ranking.	Mashup Integration
[23]	Data Integration Support for Mashups.	A framework is presented to enhance data integration in web mashup applications. It consists of components for query generation and online matching as well as for additional data transformation. Additionally the framework supports interactive and sequential result refinement to improve the quality of the presented result step-by-step by executing more elaborate queries when necessary.	Mashup Integration
[22]	Content Integration for e-Business	While this article was written before the term ‘Mashup’ was coined, it addresses some one the fundamental concepts that enables the development of mashup, namely content integration.	Mashup Integration

Source	Title	Synopsis	Category
[21]	Mash-O-Matic	The mashup production process is articulated with an emphasis on the steps needed to clean data from disparate sources and of disparate granularities.	Mashup Integration
[11]	Mashup the Deep Web.	A framework is presented that is designed to enable non-expert users to add to deep web sources to mashups by converting them into machine-processable query interfaces.	Mashup Integration
[14]	Integrating Legacy Software into a Service Oriented Architecture.	Discusses how to identify legacy systems that are good candidates for being utilized by web services, assess the conversion's potential business value, and granularize existing logic to be conducive with the web service.	Mashup Integration
[28]	Mashup Feeds: Continuous Queries over Web Services.	A collection-based, stream processing, semantic induction method is presented to enable information extraction by monitoring source evolution over time.	Mashup Agents
[26]	panOULU Luotsi: A Location Based Information Mashup with XML Aggregator and WiFi Positioning.	A mashup is presented that merges XML content in various forms, such as RSS/ATOM feeds from several content providers, into a database using a flexible XML aggregator. Information relevant to the user's current location is imposed on a map for a location-based browsing view, which allows the user to learn about nearby services, sites and events of interest.	Mashup Agents
[24]	Intel Mash Maker: Join the Web	Mash Maker, a mashing utility included in the current version of the FireFox browser is presented. The utility extracts metadata from pages being viewed by the user, anticipates material that the user might useful, and creates mashups accordingly.	Mashup Agents
[29]	An Intelligent Ontology and Bayesian Network Based Semantic Mashup for Tourism.	The authors discuss the importance and present the potential of coupling Semantic Web technologies with Web 2.0 services. A tourism recommendation mashup is presented which uses a Bayesian network to suggest tourist attractions to user, by comparing the user to other users.	Mashup Agents
[8]	Predicting Service Mashup Candidates Using Enhanced Syntactical Message Management.	Natural Language Processing is applied as a predictive agent to determine which web services may be related, and thus appropriate for a particular mashup.	Mashup Agents
[78]	Smash: Semantic-Based Mashup Navigation For Data API Network.	A semantic based agent is developed to determine which API's are related that users may couple API's in effective mashup building.	Mashup Agents

Source	Title	Synopsis	Category
[25]	Application Framework With Demand-driven Mashup For Select Browsing.	A mashup development framework is presented. It is based on a data management engine to enable the developer to identify semantic relationships, and then to browse based on semantic relevance.	Mashup Agent
[32]	A Mashup-Based Strategy for Migration to Service-Oriented Computing.	Propose a 6 phased methodology to incorporate legacy systems into mashups from a Service Oriented Architecture perspective. Phases include: Model, Analyze, Map (target enterprise model), Design (mashup server), Define (service level agreement), Implement, and Deploy.	Mashup Framework
[40]	MU: A Hybrid Language for Web Mashups.	A scripting language is presented that provides support for unification, it is based on the type morphing paradigm, provides user interface induction, and defines both the java runtime environment and java script profiles.	Mashup Framework
[15]	Database–Driven Web Mashups.	A database-driven approach to web mashups is presented that allows data integration and mashup logic to be managed within a database to enables developers to work with a uniform abstract model and to have direct access to powerful features of database systems.	Mashup Framework
[30]	Modeling the Mashup Space	A mashup model is presented that quantifies the different roles of mashups which are: query data sources, import other mashups, use external Web services, and specify complex interaction patterns between its components.	Mashup Framework
[31]	Two Cultures: Mashup Web 2.0 and the Semantic Web.	The differences between Web 2.0 and Semantic Web are disputed by reinforcing their commonalities. The authors advocate a paradigm shift from an overly machine-centered AI view of the Semantic Web towards a more user and community centered approach that draws from the insights of Web 2.0.	Mashup Frameworks
[4]	Semantic Blogging and Decentralized Knowledge Management.	Presents a framework to suffice the organizational need of a decentralized, informal, knowledge management system. The framework is a middle ground between blogging and mashups, where multiple users can centrally contribute to decentralized information.	Mashup Frameworks
[41]	A Methodology for Quality-based Mashup of Data Source.	A review of mashup literature is conducted, and the need for a mashup framework is identified. The authors present a framework that promotes mashup quality by focusing on inputs.	Mashup Frameworks

Source	Title	Synopsis	Category
[34]	A Web 2.0 Platform to Enable Context Aware Mobile Mashups.	Incorporate web 2.0 mashups to ambient intelligence technology environments, to allow users to access services and multimedia data according to their current context (location, identity, preferences). The authors postulate that applying Web 2.0 principles to the development of middleware support for context-aware systems could result into a wider adoption of ambient intelligence.	Mashup Framework
[33]	Implementation of Ubiquitous Personal Study Using Web 2.0 Mashup and OSS Technologies.	The authors propose a framework to organize individual information and support information access collaboration for all kinds of users. The framework, Ubiquitous Personal Study (UPS), is independent of service providers and stores personal resources (e.g. profiles and personal activity) and uses tagging to classify and organize information.	Mashup Frameworks
[39]	Rapid Prototyping of Semantic Mashups through Semantic Web Pipes.	Semantic Web Pipes is presented to support fast implementation of Semantic data mash-ups while preserving desirable properties such as abstraction, encapsulation, component-orientation, code re-usability and maintainability.	Mashup Frameworks
[43]	What Do We "Mashup" When We Make Mashups.	A qualitative review of various mashups is conducted, and mashup categories are discussed in terms of search, visualization, real-time, widget, personalization, folksonomy, and in-situ use.	Mashup Frameworks
[37]	KC3 Browser: Semantic Mashup an Link-free Browser.	A framework for a semantic browsing interface called knowledge communication is presented, it integrates multimedia and web services on grid networks, and makes a semantic mashups with various visual gadgets according to user's contexts. The framework achieves a link-free browsing for seamless knowledge access by generating semantic links based on an arbitrary knowledge models such as ontology and vector space models.	Mashup Frameworks
[36]	Hacking, Mashing, Gluing: A Study of Opportunistic Design and Development	Three themes in mash-up design are then evaluated: how components are combined, what the characteristics of the activity of opportunistic design are, and how mash-ups are unique artifacts.	Mashup Framework
[42]	Towards an Advertising Business Model for Web Service Mashups.	A business model is presented for online advertising in web service mashups.	Mashup Frameworks

Source	Title	Synopsis	Category
[35]	A Web Mashup for Social Libraries.	While social networks are a web 2.0 technology that are benefiting from enhanced user involvement (e.g. folksonomy), their collaborative contributions are restricted to each network, as the social network owners are not providing API to all such content to flow on the web. The authors present a framework to bridge the gap between the disparate social networks.	Mashup Frameworks
[38]	Elucidating the Mashup Hype: Definitions, Challenges, Methodical Guide and Tools for Mashups.	A literature review is conducted and a distinction between SOA's and mashups is postulated.	Mashup Frameworks
[10]	Mashlight: A Lightweight Mashup Framework for Everyone.	A mashup building framework is presented that enables non technical users to create process-like mashups using widget like web 2.0 applications. On benefit to the framework is that it does not require a particular web server but can be run on any webkit compliant browser.	End User Programming
[54]	Building Mashups By Example.	An end user mashup building approach is presented that combines most problem areas in Mashup building into a unified interactive framework that requires no widgets, and allows users with no programming background to easily create Mashups by example.	End User Programming
[56]	Making Mashups With Marmite: Towards End-User Programming for the Web.	An end user mashup tool is presented that is designed to enable end-users to easily extract information from web pages, process extracted data (e.g. exclude certain content, add metadata), integrate multiple data sources, direct the output to a specified location (e.g. DB, map service, text file).	End User Programming
[52]	Towards a Mashup-driven End-user programming of SOA-based Applications.	The authors discuss mashup development with a focus on reducing the development cost by empowering specified user groups to create applications that support their daily activities.	End User Programming
[28]	UQBE: Uncertain Query by Example for Web Service Mashup.	A query by example mashup tool for non-programmers is presented, it supports query by example over a schema made up by the user without knowing the schema of the original sources.	End User Programming
[46]	Mashmaker: Mashups for the Masses.	A tool is presented that allows users to collaboratively share and explore data and queries, the users can share data, widgets, and widget suggestions all using a simple social network that is dynamically maintained.	End User Programming

Source	Title	Synopsis	Category
[53]	Web Mashup Scripting Language.	The authors discuss an approach of using an interim web service that can be automatically generated from the pair-wise mappings of legacy web services' data models.	End User Programming
[49]	Mashing Up Visual Languages and Web Mashups.	A recent trend is discussed, detailing the shift away from the desktop computing model where software is installed locally on your machine, towards web applications where personal and public data and services coexist on remote servers distributed across the web. The following cognitive dimensions are then discussed from a mashup perspective: Stability, Robustness, and Share-ability.	End User Programming
[48]	Potluck: Data Mashup Tool for Casual Users.	The need and various scenarios of mashups is discussed. An end user mashup tool is presented and empirically evaluated. The results indicated that users were able to quickly learn the user interface and successfully created mashups.	End User Programming
[45]	Lash-Ups: A Toolkit for Location-Aware Mash-Ups.	A toolkit is presented that enables smart phone users to create mobile location specific mashups. The toolkit provides two main benefits. Firstly, it provides simple, standard API for mobile user. Secondly, it enables user to distribute their mashups to other users for reuse.	End User Programming
[47]	User-Friendly Functional Programming for Web Mashups.	While there are a plethora of different mashups sites currently available, the authors highlight areas where mashups are currently needed. In fact they mention that the number of needed mashups is so great that the only practical solution is to enable end users to create mashups. Thus, an end user mashup tool is presented.	End User Programming
[55]	Mashroom: End-User Mashup Programming Using Nested Tables.	An end-user programming model is presented that includes an expressive data structure as well as a set of formally-defined mashup operators. The model uses a table structure to enable users to express complex data objects. An empirical evaluation revealed that users effectively and efficiently used the application to build mashups.	End User Programming
[51]	End-User Programming of Mashups With Vegemite.	An end user mashup tool is empirically evaluated, the tool focuses on programming by demonstration, iterative and interactive transformation of data by the user, and mixed-initiative interaction.	End User Programming

Source	Title	Synopsis	Category
[44]	Increasing the Visibility of Web-Based Information Systems via Client-Side Mashups.	The common hindrances of mashups are discussed, such as client-side programming, proxy servers, and suitable web service interfaces. A mashup approach is then presented that avoids these three mashup pitfalls.	End User Programming
[50]	Rapid Development of Spreadsheet-Based Web Mashups.	The need for web based data mashups is discussed, then a spreadsheet like approach to data mashups is presented that addresses the following mashup obstacles: access, synchronization, re-use, and manipulation.	End User Programming
[27]	EzWeb/FAST: Reporting on a Successful Mashup-based Solutions for Developing and Deploying Composite Applications in the Upcoming Web of Services.	The authors elaborated on the synergies between the Web 2.0 and the enterprise application worlds that can be exploited. They then present a model for global user-centric SOA and a novel architecture for enterprise mashup composite applications.	Enterprise Mashups
[60]	Enterprise Mashups: Design Principles Towards the Long Tail of User Needs.	The components of Enterprise Mashups are presented as a stack composed of resources, API's, widgets, mashups, and web applications. The development lifecycle of enterprise mashups is discussed which is perpendicular to traditional lifecycles and resembles the rapid prototyping lifecycle.	Enterprise Mashups
[63]	Data Mashups and Their Applications in Enterprises.	The authors discuss web 2.0 mashups in the domain of enterprises. Specifically they compare mashups to traditional applications in terms of having a higher interpretation value and lesser navigational costs.	Enterprise Mashups
[65]	Damia: Data Mashups for Intranet Applications	The evolution of the enterprise intranet to a platform for web 2.0 applications is presented as an opportunity for business leaders to exploit information from desktops, the web, and other non-traditional enterprise sources, in order to react to situational business needs.	Enterprise Mashups
[61]	Enterprise Information Mashups: Integrating Information Simply.	The need for enterprise mashups is discussed, and a development approach is postulated that focuses on utilizing existing information systems to provide a foundation for enterprise mashup applications.	Enterprise Mashups
[67]	Towards Accountable Enterprise Mashup Services.	The legal implications of applying mashups to enterprises is discussed. A model is presented that can be used by information systems developers to understand the roles and responsibilities that need to be accommodated in a mashup service solution.	Enterprise Mashups

Source	Title	Synopsis	Category
[60]	SAP Research Rooftop Marketplace.	The authors discuss the lack of consistency in literature concerning the scope, role, and definition of enterprise mashups. The adoption of enterprise mashups is described as a phased approach.	Enterprise Mashups
[57]	Damia: A Data Mashup Fabric For Intranet Applications.	An enterprise mashup framework is presented. The framework consists of a browser-based UI to chart data flows, a server-based execution engine, and API's for mashing.	Enterprise Mashups
[62]	Operational Challenges and Solutions for Mashups – An Experience Report	The organizational and technical challenges of enterprise mashups are discussed. To address these issues the authors propose that focus be maintained on centralized configuration and monitoring, to enable mediating and managing of disparate components.	Enterprise Mashups
[64]	Cloud-based Enterprise Mashup Integration Services for B2B Scenarios	A literature review is presented that distinguishes between consumer and enterprise mashups. The relationship between reach and richness is discussed and a prototype mashup integration service is presented.	Enterprise Mashups
[66]	Enterprise Mashup Composite Services in SOA – User Profile Use Case.	Best practices for enterprise mashup development and implementation is discussed.	Enterprise Mashups
[58]	Towards Physical Mashups in the Web of Thing	Applies the concepts of Web 2.0 mashups to Enterprise Wireless Sensor Network's to develop physical mashups.	Enterprise Mashups