

# Semantic-aware Mashup for Personal Resources towards Open Semantic Enterprise

## DISSERTATION

zur Erlangung des akademischen Grades

**Doktor/in der technischen Wissenschaften**

eingereicht von

**VO Sao Khue**

Matrikelnummer 0727941

an der

Fakultät für Informatik der Technischen Universität Wien

Betreuung: Univ.Prof. Dipl.-Ing. Dr.techn. A Min Tjoa

Diese Dissertation haben begutachtet:

---

(Univ.Prof. Dipl.-Ing. Dr.techn.  
A Min Tjoa)

---

(Univ.-Prof. Dr. Josef Küng)

Wien, 30. April 2014

---

(VO Sao Khue)



# **Semantic-aware Mashup for Personal Resources towards Open Semantic Enterprise**

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

**Doktor/in der technischen Wissenschaften**

by

**VO Sao Khue**

Registration Number 0727941

to the Faculty of Informatics

at the Vienna University of Technology

Advisor: Univ.Prof. Dipl.-Ing. Dr.techn. A Min Tjoa

The dissertation has been reviewed by:

---

(Univ.Prof. Dipl.-Ing. Dr.techn.  
A Min Tjoa)

---

(Univ.-Prof. Dr. Josef Küng)

Wien, 30. April 2014

---

(VO Sao Khue)

# Erklärung zur Verfassung der Arbeit

Sao-Khue, VO

Donaufelder straße 54/3301, 1210 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

---

(Wien, 30. April 2014)

---

(VO Sao Khue)



# Acknowledgements

The work of this dissertation has been conducted at the Institute of Software Technology and Interactive System, Vienna University of Technology, Austria. However, I would never have been able to finish my dissertation without the support and encouragement of my Professors, my colleagues, my friends, and my family.

First of all, I would like to express my deep gratitude to my supervisor, Professor A Min Tjoa for his patient guidance, encouragement and useful critiques of this research work. More important, I am greatly indebted for his spiritual support in many aspects of life.

I would also like to express my sincere thank to Professor Josef Küng, my second advisor for his revise and guidance to my work.

I am very grateful to Dr. Amin Anjomshoaa for his inspiring advices and valuable comments during the planning and development of my work. He also supported me in the new literature and pointed me to the good ideas.

In addition, I would also like to thank to my friends, my colleagues at the Institute of Software Technology and Interactive System for supporting me in my research. Especially, I very much appreciate the members of Secure 2.0 project and SEMERGY project providing me the perfect working environment.

I would finally like to thank to my wife, my older brother, and my father who were always encouraging me with their best wishes. Although I will not have the chance to share this work with my mother, she is always in my heart and I will try to make her proud always in everything I do.

# Abstract

Business objectives are accomplished successfully when human resource management systems are developed and implemented according to organizational goals, particularly if personal information management (PIM) is adopted to utilize all employee information productively. Some PIM systems use Semantic Desktops, the semantic personal desktop layer for integrating applications and personal life items, as the means to support users in information management. More and more organizations/enterprises are taking advantage of mashups, which support users in fast integration of heterogeneous data from multiple sources. However, although most people and organizations/enterprises benefit from the collaborative principles of Web 2.0 technologies by using social networking sites (SNSs) to build their social activities/relations and support their knowledge management, the unstructured data overload is increasing and distributing in both human lifetime and workplace. In addition, those PIM systems are limited to local storage or isolated data repositories, and do not fulfill most of the requirements for a collaborative environment, above all at the organizational level. Therefore, a key issue arises in the necessity to provide a flexible and semantic-based way for bringing in internal and external data sources (especially personal information sources from Semantic Desktops and SNSs) into enterprises.

This thesis aims to utilize semantic web and mashup technologies for semantic-based information integration and to leverage existing personal resources in Semantic Desktops and SNSs. A lightweight mashup language and a semantic-based mashup framework are proposed to enable a semantic-aware mashup dataflow that primarily supports non-experts to create mashup data for personal/organizational use. In addition, reusable components for Web 2.0 information retrieval are developed to prepare mashable resources and trustworthy mashup data. Subsequently, the mashup results can be combined with other policies for self-monitoring purposes in preventing the disclosure of personal/organizational information in Web 2.0 via SNSs. The introduced mashup system and its components could be adapted to the layered approach of Open Semantic Enterprise for the semantic data integration in enterprises. Such adaption would support knowledge workers and enable activity-oriented collaboration, as well as to team up with coworkers in the collaborative environment of enterprises.

# Kurzfassung

Die Erreichung von Unternehmenszielen ist wesentlich von der Effizienz ihrer Personalmanagementsysteme abhängig. Deshalb ist es ein Forschungsziel, Personal Information Management (PIM) - Systeme so zu gestalten, dass möglichst viel Mitarbeiterinformationen berücksichtigt werden.

Einige PIM-Systeme verwenden zur Unterstützung des Informationsmanagements Semantic-Desktop Ansätze mit einer eigenen semantischen Schichte zur Integration von Applikationen und rechtlich konformen persönlichen Informationen. Sehr oft jedoch erfüllen die existierenden Systeme die spezifischen Anforderungen einer Kollaborationsumgebung nicht.

Obwohl Organisationen und deren MitarbeiterInnen von den Kooperationsmöglichkeiten der Web 2.0-Technologien und deren Social-Networking-Sites (SNS) sehr wesentlich profitieren können, haben wir es doch mit einer zunehmenden Überlastung durch unstrukturierte Daten am Arbeitsplatz zu tun.

Deshalb zielen immer mehr Organisationen/Unternehmen darauf ab Mashups, welche eine schnelle Integration heterogener Daten aus unterschiedlichen Quellen ermöglichen, zu nutzen. Hierzu ist es notwendig, ein flexibles und semantisch gesteuertes System für interne und externe Datenquellen - insbesondere persönliche Datenquellen- im Unternehmen zur Verfügung zu stellen.

Diese Arbeit zielt weiters darauf ab, Semantic Web und Mashup Technologien so zu nutzen, dass eine semantisch-basierte Verwendung vorhandener Ressourcen in Semantic Desktop sowie Social Network Systemen ermöglicht wird. Zu diesem Zweck wird eine Mashup Sprache und eine semantisch-basierte Mashup- Grundstruktur vorgeschlagen, die dem Benutzer in leichter Weise das Arbeiten mittels semantischen Mashup- Workflows gestatten.

Darüber hinaus wurden wiederverwendbare Komponenten für Web 2.0-Information-Retrieval entwickelt, um „mashable“ Ressourcen und vertrauenswürdige Mashup Daten vorzubereiten. Es ist ein weiteres Ziel dieser Arbeit, dass Mashup-Ergebnisse mit dem Monitoring von Social Network Systemen (zur Verhinderung eines möglichen Missbrauchs von schutzwürdigen Informationen) integriert werden, sodass die Weitergabe und Nutzung solcher Informationen im Web 2.0 präventiv verhindert werden können.



Die vorgestellten Mashup-Ansätze können in einem mehrschichtigen Open Semantic Enterprise für die jeweilige semantische Informationsintegration in Unternehmen angepasst werden. Solche Anpassungen würden in einem kollaborativen Umfeld eines Unternehmens eine verbesserte handlungsorientierte Zusammenarbeit der MitarbeiterInnen ermöglichen.

# Contents

CHAPTER 1	INTRODUCTION .....	1
1.1	Introduction and Motivation.....	1
1.2	Research Questions .....	3
1.3	Thesis Contributions .....	3
1.4	Thesis Organization.....	4
CHAPTER 2	BACKGROUND AND RELATED WORK.....	5
2.1	Background .....	5
2.1.1	Web 2.0 and Social Web .....	6
2.1.2	Linked Data .....	7
2.1.3	Mashup.....	9
2.1.4	Open Semantic Enterprise.....	13
2.1.5	Knowledge Worker .....	17
2.2	Related Work.....	19
2.2.1	Semantic Desktop .....	19
2.2.2	Open Semantic Enterprise.....	20
2.2.3	Mashup Approaches.....	20
2.2.4	Mashup Security.....	28
2.3	Summary .....	29
CHAPTER 3	MASHABLE PERSONAL RESOURCES AND SERVICES.....	30
3.1	Personal Resources in SematicLIFE .....	30
3.2	Personal Resources in SocialLIFE.....	32
3.3	Mashable Personal Resources .....	33
3.3.1	Linking Personal Resources with LOD Cloud.....	33
3.3.2	Semantic-based Personal Resources Retrieval .....	38
3.3.3	Semantic-enabled Personal Services .....	39
3.4	Summary .....	42

CHAPTER 4	TRUSTWORTHINESS OF MASHUP DATA.....	43
4.1	Self-Monitoring in Social Networking Sites.....	43
4.2	Knowledge Sharing Policies .....	49
4.3	Exploiting Disambiguated Information Retrieval .....	50
4.3.1	Word Sense Disambiguation for Mashable Resources .....	51
4.3.2	SOM-based Personal Resources Clustering.....	54
4.4	Summary .....	58
CHAPTER 5	SEMANTIC-BASED MASHUP .....	59
5.1	Semantic-based Mashup .....	59
5.2	Personal Resources Mashup Language .....	61
5.3	Semantic Mashup Formulation .....	64
5.3.1	Definition and Rule .....	64
5.3.2	Realization of Definitions and Rules .....	65
5.3.3	Widget-based Query Generation .....	66
5.3.4	Mashup Algorithm.....	68
5.4	Semantic Mashup Patterns.....	70
5.5	Summary .....	72
CHAPTER 6	IMPLEMENTATION RESULTS AND EVALUATION .....	73
6.1	Implementation Results .....	73
6.1.1	Personal Resources Retrieval from SocialLIFE .....	73
6.1.2	Mashup Workspace.....	77
6.1.3	Widget Tree.....	77
6.1.4	Widget UI.....	79
6.1.5	Mashup Editor .....	82
6.1.6	Mashup Portal .....	83
6.1.7	Mashup Sequence Diagram .....	84
6.1.8	Mashup Use Cases .....	85
6.2	Mashup Framework Evaluation.....	91
6.2.1	Mashup Framework Components.....	91

6.2.2	Data Retrieval Strategy.....	91
6.2.3	Mashup Development Cycle .....	92
6.2.4	Simple User Interaction Mechanism .....	92
6.2.5	Security and Privacy Policy.....	93
6.2.6	Integrating SemanticLIFE and SocialLIFE data.....	94
CHAPTER 7 CONCLUSION AND FUTURE WORK .....		95
7.1	Conclusion.....	95
7.2	Future Work.....	98
Bibliography .....		99

# List of Figures

Figure 2.1: Recent mashup trends and API growth [22] .....	10
Figure 2.2: An example of data mashup for Austrian museums .....	11
Figure 2.3: Enterprise Mashup Architecture [24] .....	12
Figure 2.4: Principal pillars of Open Semantic Enterprise [29].....	15
Figure 2.5: Layered approach in Open Semantic Enterprise [29].....	16
Figure 2.6: The implementation of the vision in Open Semantic Enterprise [30] .....	17
Figure 2.7: An example of DERI Pipes [84].....	25
Figure 2.8: An example of MashQL Queries [88] .....	27
Figure 2.9: Mashup creation with JackBe Presto [51] .....	28
Figure 3.1: SemanticLIFE framework [12] .....	31
Figure 3.2: The most popular SNSs in 2013 via the conversation prism [104].....	32
Figure 3.3: Linking SemanticLIFE and SocialLIFE with LOD Cloud.....	35
Figure 3.4: Architecture for publishing data as Linked Data .....	36
Figure 3.5: Conversion of financial data from OFX format into RDF/N3 .....	37
Figure 3.6: RDF representation of personal resources in SemanticLIFE and SocialLIFE .....	38
Figure 3.7: SPARQL query towards data mashups for SemanticLIFE and SocialLIFE .....	39
Figure 3.8: LIDS description for APIs and RESTful services.....	40
Figure 3.9: Lowering and lifting in LIDS for APIs and RESTful service .....	41
Figure 3.10: An example of semantic-enabled personal services .....	42
Figure 4.1: Overall solution for self-monitoring in social network [124].....	46
Figure 4.2: SOM visualization of high risk group on the friends' interest map [124].....	47
Figure 4.3: Mashup solution to create a SOM visualization of high risk group of the friends' interest in Facebook [12] .....	48
Figure 4.4: The main classes and properties of the Privacy Preferences Ontology [128] .....	50

Figure 4.5: Word Sense Disambiguation for mashable personal resources.....	53
Figure 4.6: Input and Template Vector file of SOM training for clustering friends' interest in Facebook.....	57
Figure 4.7 SOM Visualization and clustering for friends' interest in Facebook .....	57
Figure 5.1: Semantic-based mashup architecture for SemanticLIFE and SocialLIFE .....	60
Figure 5.2: The schema of personal resources mashup language (PRML).....	61
Figure 5.3: The schema of widget parameters of PRML.....	63
Figure 5.4: The process for applying ontologies and mashup rules in the mashup.....	65
Figure 5.5: Query generation based on widget parameters .....	67
Figure 5.6: An example of query generation based on widget parameters.....	68
Figure 5.7: An example of mashup pattern to retrieve personal resources. ....	71
Figure 5.8: Mashup patterns store of personal resources .....	71
Figure 6.1: An example of personal resources retrieval from social data in Facebook & Freebase.....	75
Figure 6.2: Query Freebase schema for data annotation .....	76
Figure 6.3: Mashup Workspace .....	77
Figure 6.4: Widget tree for personal resources mashups in SemanticLIFE and SocialLIFE...	78
Figure 6.5: Widget UI generation mechanism.....	79
Figure 6.6: Parsing FacebookEvents widget' parameters into Widget UI and relevant SPARQL query. ....	81
Figure 6.7: Mashup editor with highlighted feasible connection .....	82
Figure 6.8: Mashup Portal with some sample widget UIs .....	83
Figure 6.9: Sequence diagram of designing and running mashup .....	84
Figure 6.10: Personal finance mashup for showing bank statements in calendar view .....	85
Figure 6.11: SOM visualization and clustering of friends' interest from Facebook. ....	87
Figure 6.12: SOM visualization and clustering of friends' tweets from Twitter for self-monitoring. ....	88

Figure 6.13: A personalized mashup use case on demand.....	89
Figure 6.14: Design personalized mashup in Mashup Editor .....	90
Figure 6.15: Semantic-aware dataflow.....	92
Figure 6.16: Running mashup platform in multiple environments perspective. ....	93
Figure 7.1: The vision of adapting mashup application in the layered approach of Open Semantic Enterprise. ....	96

## List of Tables

Table 2.1: Most popular Web 2.0 technologies and their usage possibilities in organizations/enterprises. ....	7
Table 2.2: Status and features comparison of popular mashup tools .....	26
Table 6.1: Advanced features in semantic-based mashup system compared with other mashup products. ....	94

## Algorithm

Algorithm 5.1: Mashup algorithm .....	69
---------------------------------------	----

## Abbreviations

AJAX	Asynchronous JavaScript and XML
BPEL	Business Process Execution Language
EKM	Enterprise Knowledge Management
EMML	Enterprise Mashup Markup Language
JSON	JavaScript Object Notation
LED	Linked Enterprise Data
LIDS	Linked Data Services
LOD	Linked Open Data
LOS	Linked Open Services
OFX	Open Financial Exchange
OMA	Open Mashup Alliance
OSE	Open Semantic Enterprise
PKM	Personal Knowledge Management
RDF	Resource Description Framework
RSS	Rich Site Summary or Syndication
SOM	Self-Organization Map
SPARQL	Simple Protocol and RDF Query Language
SNSs	Social Networking Services
XML	Extensible Markup Language
XPDL	XML Process Definition Language
WSDL	Web Service Description Language



## INTRODUCTION

### 1.1 Introduction and Motivation

Personal information management is considered as a key to human resource management, which plays an important role in accomplishment of business objectives. In 1988, Baird and Meshoulam stated, *“Business objectives are accomplished when human resource practices, procedures and systems are developed and implemented based on organizational needs, that is, when a strategic perspective to human resource management is adopted”* [1]. From the organization and individual perspectives, employees should be aware that the value of personal knowledge management (PKM) is *“helping individuals to be more effective in personal, organizational and social environments”* [2].

Some PKM applications are partially adapted to managing personal information over a human lifetime by using Semantic Web and ontology as a basis for content representation. Semantic Web is an effort to create a new technological framework that represents information more meaningful for both humans and computers [3]. As the backbone of Semantic Web, ontology is a key technique which provides common vocabulary, represents knowledge, and annotates resources with semantic for organizing information and publishing data. However, these PKM applications are limited to local storage and do not fulfill most of the requirements for a collaborative environment. Although enterprise knowledge management (EKM) systems entail formally managing knowledge resources to facilitate accessing and reusing of knowledge [4], these systems cannot work effectively unless the knowledge workers contribute their knowledge resources and assets to organizations/enterprises.

Recently, many organizations/enterprises started using the Web 2.0 techniques by applying SNSs in order to increase effectiveness of their business in collaboration. In 2006, McAfee used the term Enterprise 2.0 to focus on aspects of Web 2.0 platforms that are used

by enterprise to make practices and results of their knowledge workers visible [5]. Web 2.0 has changed the Internet paradigm from a traditional read-only web to a social web that facilitates information sharing, collaboration and business processes. People have the tendency to share their knowledge or resources which are not only stored locally on their personal computer or isolated data repositories, but also transferred to SNSs on the web (e.g. Google documents, MindMeister mind maps, YouTube videos, LinkedIn profiles, Flickr images, Twitter tweets, etc.). With the new generation of the World Wide Web, people interact with SNSs by expressing their profiles, schedules, plans, and activities in an interoperable and extensible way.

In the meantime, there is quite a lot of discussions about the success and challenges of Enterprise 2.0 projects [6], [7] in applying Web 2.0 techniques via SNSs to create an effective collaborative community, and understanding how social computing can help the enterprises achieve their performance goals. The potential of Enterprise 2.0 cannot be fully realized without the active support of human resources [8]. With SNSs in an enterprise environment, employees can share their data (e.g. skills, interests, or activities, etc.) with their groups or colleagues. From the enterprise perspective, employees can collect business information from customers and partners through SNSs by exploring the relationship of business establishments, professionals, or individuals. In an informal survey of organizations, 65% of workers in big companies rely on themselves and co-workers, and not on their management, to solve their problems [9]. For example, people often solve their problems by searching in Google, reading in Wikipedia, finding experts or advisors in LinkedIn, etc. However, it should be noticed that the volume of sharing data has increased rapidly and the disclosure of personal/organizational information in Web 2.0 has created new security and privacy challenges. On the other hand, the sharing data are not structured enough to enable advanced data usage in an enterprise environment.

In order to support users in exploiting the potential of sharing data on the web, the data should be managed in a machine-processable way by applying Semantic Web technologies. During designing and discussing issues around the Semantic Web, Tim Berners-Lee came up with the new term “Linked Data” that describes a method for publishing and interlinking structured data so that it can become more useful for person or machine exploring the web of data [10]. Besides, more and more social software and Web 2.0 applications have published their APIs that enable software developers to “mashup” data by their APIs services. Mashup is considered as a new proposition of social software and Web 2.0 for combining various data

sources and services to be applied to new types of resources. The amount of user-generated content sharing in SNSs is the potential resources for mashup in personal and business use cases, such as mashup of customers or friends' geo locations from your social networks to find who have similar interested topics or products, etc. For an example of mashup, HousingMap is a popular one that combined housing data from Craigslist [11] and displayed them on Google Map.

From the above preliminary remarks, it is necessary to provide a flexible and semantic-driven way to bring internal and external data sources, and especially personal resources into enterprises. In addition, this research is trying to proceed the idea of "Integration of Personal Services into Global Business" [12] that bridges the gap between the personal information world and the global business. The ultimate goal of this research is also to find the solution for the success of semantic-aware mashup for personal resources from Semantic Desktops and SNSs in the trend of Enterprise 2.0.

## **1.2 Research Questions**

This dissertation will deal with the following questions:

- How to apply a semantic-driven approach that integrates personal life items in Semantic Desktops and SNSs in order to benefit individual, collaborative work and better solve business processes in organizations/enterprises?
- How to secure mashable resources that can be combined with personal/organization policies in order to protect and filter sharing data in a collaborative environment of enterprise?
- How to create a semantic-based unified mashup model to support end-users in the fast creation of data mashups and to fulfill users' requirements on demand for enterprise?

## **1.3 Thesis Contributions**

The main contributions of this work would be:

- Bridge the gap between Semantic Desktops and SNSs in order to integrate and reuse existing personal resources in an application. This objective also expands the scope of our Semantic Desktop system – SemanticLIFE [13] in particular - and Semantic Desktops in general into the web of data instead of isolated data silos.

- Contribute some formulations for mashup-related concepts such as semantic mashup, widget, and mashup rules.
- Propose a lightweight mashup language and a semantic-based mashup system that support end-users in designing semantic-aware mashup dataflow, aggregating and presenting mashup data.
- Utilize the proposed semantic-based mashup system to the layered approach of Open Semantic Enterprise for semantic information integration in organizations/enterprises.

## **1.4 Thesis Organization**

The thesis is composed of three main parts: Background knowledge and related work (chapter 2), Mashable personal resources and trustworthiness of mashup data (chapter 3 and 4), and Semantic-based mashup framework and Implementation results (chapter 5 and 6).

The chapters of this dissertation are organized as follows:

- Chapter 2 presents a comprehensive literature review of various background knowledge and related work to conduct our approach.
- Chapter 3 explores the data sharing of knowledge workers and proposes a solution to bridge the gap between Semantic Desktops and SNSs; and to prepare mashable artifacts for the mashup process.
- Chapter 4 investigates the issue of trustworthiness and self-monitoring of mashup data.
- Chapter 5 proposes a semantic-based mashup system that allows to mash up personal resources in both Semantic Desktops and SNSs
- Chapter 6 presents the implementation of our mashup system and the evaluation of results.
- Chapter 7 summarizes the contributions and the main results of this research.

## BACKGROUND AND RELATED WORK

This chapter will present a comprehensive literature review of various domains, such as Web 2.0 and Social Web. Current trends of Linked Data and Mashup technologies, which can be applied for the data integration towards an Open Semantic Enterprise, are also discussed. The related work will be investigated and discussed in further details.

### 2.1 Background

Collaborative environments particularly allow organizations/enterprises to realize a number of competitive advantages to use their existing technology infrastructures as well as knowledge workers for personal and teamwork collaboration. Recent approaches allow organizations to improve the performance of teamwork by collaborative software, workflow systems, documentation management systems, knowledge management systems, or social network systems. According to Ballesteros [14], the top-ranked characteristic requirements of a collaborative environment are ease of use, interoperability and scalability, service oriented architecture, any place – anytime, high quality of service, support for data security and privacy, low cost of entry and locating required information. Beyond these characteristics, the ability to fully integrate with desktop management and apply social computing concepts to become people and knowledge-centric are also considered [14]. Beside of this, it is reasonable to predict that the challenges to current collaborative environments are the lack of worker participation in knowledge management systems in particular, and the lack of organization sharing in general. The possible reasons for the former lacking could be that people do not submit or contribute their knowledge into a common knowledge repository; they do not have time to share, or even their organizations do not have supporting tools; the organizations are not ready for sharing or collaboration, or because of security issues.

Leveraging Web 2.0 features and Linked Data benefits, a number of Enterprise 2.0 platforms take a step toward a new paradigm by jointly generating, sharing and refining their

business knowledge. In this paradigm, knowledge workers are considered as co-producers of both information and software services for a collaborative environment in enterprises.

### 2.1.1 Web 2.0 and Social Web

Tim O'Reilly introduced the term Web 2.0 to refer to the second generation of services on the World Wide Web [15]. Web 2.0 is about connecting people and ideas through communications that let people collaborate and share information online in real-time. The communication mechanisms vary from podcasts, wikis, and feeds to social networking. Compared to the first generation, Web 2.0 gives users experiences closer to desktop applications than traditional static Web pages. Most of Web 2.0 applications use a combination of new technologies, including public application programming interfaces (APIs) or Representational State Transfer (REST) web services, Asynchronous JavaScript and XML (AJAX), or Really Simple Syndication (RSS) feeds. With these new technologies, Web 2.0 allows users to interact and collaborate with others by web-based social software, such as tagging, blogging, or wikis. The major inputs to Web 2.0 are users' activities and contributions. These contributions include content that users have submitted in their online activities.

Applying Web 2.0 in organizations/enterprises, which is called Enterprise 2.0 [16], allows the interaction among workers as well as customers in more collaborative and efficient ways. Enterprise 2.0 aims to help knowledge workers and customers in collaborating, sharing, and organizing information via Web 2.0 technologies. The table below gives a short description of the most popular of Web 2.0 technologies and their usage possibilities in organizations/enterprises.

Web 2.0 Technology	Short description	Usage in Organizations/Enterprises
Blog	A simple content publishing system that is easily maintained and is composed of posts.	Blog can be used as an internal communication channel in organizations (project management) or as an external communication channel (to partner and customers).
Wiki	A type of website that can be created and edited collaboratively by multiple users.	Wiki can be used extensively in organizations (e.g. knowledge management system, corporate intranets).

Social networking sites	Online community in which people create personal profiles, share information with their friends, and make contacts.	SNSs can also be built in organizations to allow workers to share ideas, activities. In addition, organizations can use SNSs to advertise their products/services.
Podcast & Video	Audio and video files are made available for streaming or downloading.	Podcasts and videos can provide learning programs for internal communication to employees or sharing of social activities.
RSS & Micro-blogging	Both methods aim to allow users to follow any updates of works. Micro-blogging are limited in characters, but can be set up broad-based conversations.	These techniques can be used to provide updated team activities or promote their products/services.
Tagging & Social bookmarking	Both techniques add notations to resources to enable easier categorization and retrieval. Social bookmarking is mainly managed for web pages.	Inside organizations, these techniques can facilitate the advanced search, better information sharing within groups as well as help staff to find relevant information and reduce duplication of research.
Mashups	Mashups aim to integrate disparate data sources or applications into a single tool.	Mashups are being used extensively in various organizations, and hold a significant potential for enabling end-users to access and manipulate information relevant to their work.

**Table 2.1:** Most popular Web 2.0 technologies and their usage possibilities in organizations/enterprises.

### 2.1.2 Linked Data

Tim Berners-Lee coined the term Linked Data to describe a method for publishing and interlinking structured data so that it can support people or machines in exploring the web of data [10]. It extends the standard Web technologies such as HTTP, URIs, and RDF to share information in a machine-readable way. Besides, the term of Linked Open Data (LOD), which is released under open license to encourage people or organizations to publish their raw data, is also mentioned as Linked Data [10]. The Semantic Web community has a gained momentum with the widespread publishing of LOD with very promising various datasets to explore. DBpedia is a typical example and a large dataset of LOD, which makes the content

of Wikipedia linked and especially incorporated to other datasets on the Web such as Geonames, WordNet, etc.

In his design issues [10], Tim Berners-Lee outlined four basic principles of Linked Data:

- Using URIs to indicate the names for things and start with HTTP.
- Using HTTP URIs to ensure that these things can be referred to and looked up.
- Showing useful information about things when their URIs is dereferenced or looked up.
- Including hyperlinks to other related things via their URIs when publishing data on the Web.

Based on the idea of LOD that turns document-oriented Web into a global giant database, Linked Enterprise Data (LED) concept has been proposed for linking enterprise data to interrelate enterprises' silos of information [17]. However, as Hyland mentioned, creating an application that combines linked enterprise content and public data is very compelling to management. Hyland summarized some principle tasks to prepare for LED as follows [17]:

- Publishing content in both human and machine understandable formats, that major search engines are able to parse.
- Leveraging existing controlled vocabularies, terms, and relationships, that enterprise has already spent resources to develop.
- Ensuring the longevity of linked data identifiers, such as by using Persistent URLs (PURLs) to manage their long-term resolution.
- Using Web Standards and Open Source Software (FLOSS) to achieve more effective information sharing, repackaging and reuse, with a minimum of specialized Web development skills.
- Following best practices and document them for others to use.
- Empowering a specialist in information analysis, access and data curation to assist data owners with procedures and support exposing data
- Recognizing that there is no such thing as a typical project; initial prototypes may be large or small, targeted toward critical data or purely academic in nature.

LOD is becoming increasingly important in information and data management. Once organizations/enterprises have made their data accessible as linked data, this opens new opportunities to efficiently reuse and leverage existing data in developing new applications targeting specific business needs.



### 2.1.3 Mashup

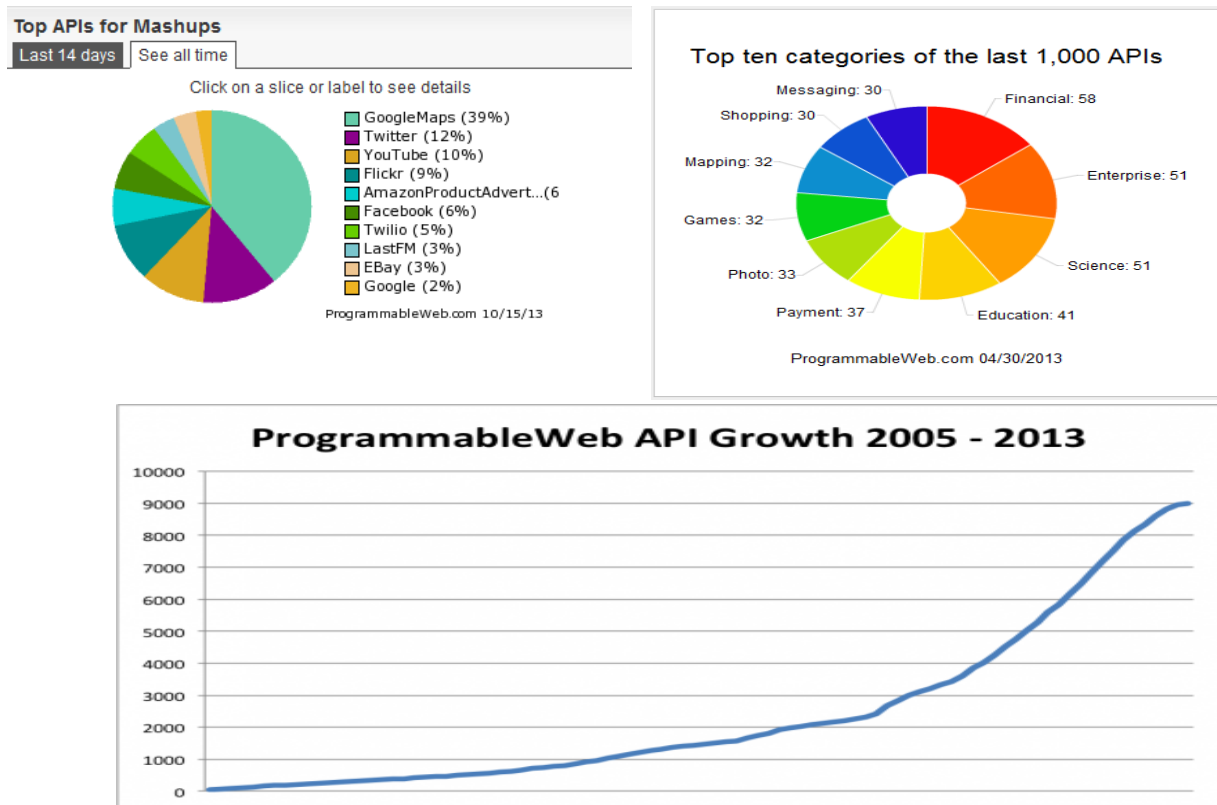
Mashup is a part of an ongoing shift towards social software and Web 2.0 for combining data and services to be used on new kinds of resources. According to a definition provided by Fichter [18], mashup is a web application that uses content from different data sources to generate a new web service presented in a single graphical interface. In enterprise-relevant definition proposed by JackBe [19], mashup is a user-driven micro-integration of Web-accessible data.

According to Breslin et al. [20], mashup can be applied to composite applications, gadgets, management dashboards, ad-hoc reporting mechanisms, spreadsheets, data migration services, social software applications and content aggregation systems. Mashup also has the potential for more fundamental and sophisticated tasks in conjunction with business processes [21]. The important benefits of mashup architecture are the possibilities of reusing existing components and sharing community-generated components with other users. These possibilities will radically decrease required time of development and implementation phases. In the largest online mashup platform ProgrammableWeb [22], there are more than 9000 mashup APIs of different topics that can be used. With those mashup APIs, clients can consume multiple services or aggregate results to facilitate their composition functionality. For a certain security reason, each service's policy must be dependent and respected.

Peenikal [23] showed that mashups may be divided into three main types:

- *Data mashups*: combine similar types of information and media from various sources into a single representation. The combinations of these sources create completely new web services that were not originally provided by either source.
- *Consumer mashups*: combine different data types; generate visual elements and data from multiple sources.
- *Business (or enterprise) mashups*: specify applications that combine their own data and application with other external web services. These mashups focus data on a single presentation and allow collaborative actions among business users and developers.

The below figure shows recent mashup trends and API growth, which are classified in categories (e.g. enterprise, financial, science, etc.) and mashup types (e.g. Flickr, Facebook, etc.) in ProgrammableWeb.



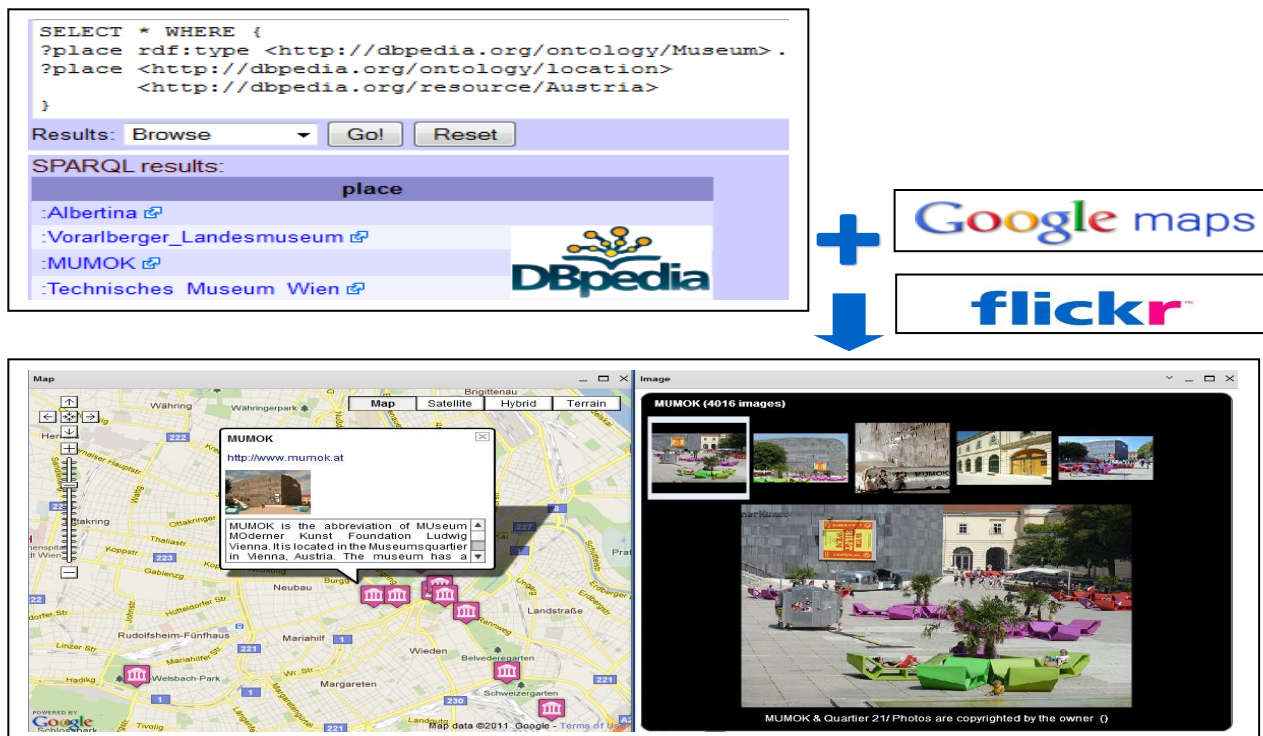
**Figure 2.1:** Recent mashup trends and API growth [22]

The following example illustrates a data mashup that combines museum data (name, description, real-world geographic location) and museum pictures. The issue is that the necessary data are stored in different sources and applications, for example museum data and pictures are stored in DBPedia and Flickr, respectively. In order to integrate these data and present them in a unique layout, a data mashup solution should be applied. The required components for this mashup include:

- Linked Open Data from DBPedia: is a structured dataset associated with Wikipedia resources.
- Open API Web Service from Google Maps: is a web mapping service application.
- Open API Web Service from Flickr: is a popular image hosting website.
- An HTML hosting the mashup content.

In this example, the Austrian museum data are queried via the SPARQL endpoint of DBPedia. Based on the unique real-world geographic location of each museum, a request

containing geolocation is sent to Google Maps API Web Service, and a response is parsed to display the museum on the Google map. Concurrently, the Flickr API Web Service is used for searching for the museum photos that are tagged by the community. These mashup data will be then displayed in an HTML host. The overview of this mashup is illustrated as follows:



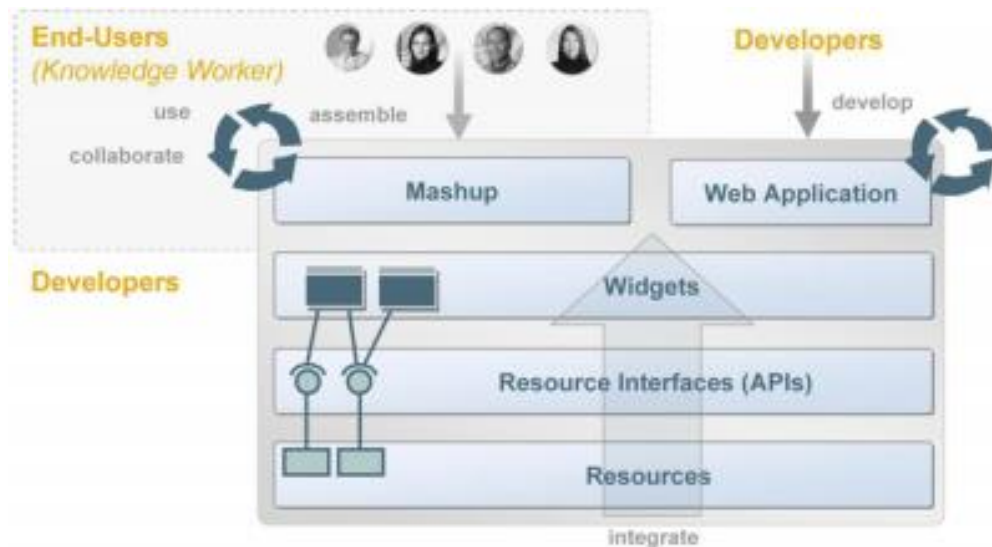
**Figure 2.2:** An example of data mashup for Austrian museums

### 2.1.3.1 Enterprise Mashup

Enterprise mashup is exploited as a possible enabling technology to get a step closer to Enterprise 2.0. Hoyer et al. has defined "An enterprise mashup is a *Web-based resource that combines existing resources, be it content, data or application functionality, from more than one resource by empowering end users to create individual information centric and situational applications*" [24]. The enterprise mashup concept can be compared with other common enterprise technologies like Business Process Management (BPM), Enterprise Service Bus (ESB), Enterprise Application Integration (EAI), or Enterprise Information Integration (EII) [25], [26]. Enterprise mashup is none of these things but conditionally complement all of them. Enterprise mashup supports and extends those technologies by aggregating and introducing the ability to create dynamic, user-centric solutions. ESBs are point-to-point solutions, such as application to application. EAI aims to connect corporate systems at

application level rather than at data level and streamline business processes, whereas enterprise mashup typically combines applications with goal of providing new functionality.

Architectural components of enterprise mashup are required resources, widgets, APIs, and mashups. While mashups focus on application-to-user solutions that provide a visual creation component oriented to end-users in real-time; enterprise mashup enables the automation of situational needs of end-users. The following figure depicts a typical architecture of enterprise mashup.



**Figure 2.3:** Enterprise Mashup Architecture [24]

Mashup tools contribute to a new vision of software development, where users are able to reuse data, contribute, and expose core services within an organization/enterprise. These tools should not require programming skills but rather support visual widgets, and integrated components together. It could be realized that drag-and-drop mashup tools are simple enough for users. In design principles for enterprise mashup architecture, Hoyer et al. [24] specified the particular user roles as follows:

- *End-Users*: execute mashup scenarios, or personalize individual environment.
- *Key-Users*: create mashup scenarios by adding pre-build widgets, or connecting widgets through their input/output ports.
- *Consultants*: create widgets by binding generic resources/services to UIs, or transforming and aggregating data.
- *Developers*: make resources/services available and create or deploy services.

### 2.1.3.2 Mashup Pattern

Mashup pattern represents the way to combine various services and the way to show visualization data. Mashup pattern relies on services that integrate data from multiple resources to create a new content, such as a new view or a new data source. There have been a number of studies in mashup patterns [27] [28]. These studies provided the concepts and addressed guidelines of different mashup patterns as follows:

- *Self-service Pattern*: Mashups are created by business users in order to support for their required solutions through this pattern. This kind of pattern can range from simple functions such as viewing data, personalizing information to complicated functions such as mass customization of mashups from multiple resources.
- *Source Integration Pattern*: Organizations/Enterprises can integrate both internal and external resources into potential mashups for decision-making or business needs. Through these mashup patterns, organizations/enterprises can present their backend information as services from disparate information sources.
- *Share & Reuse Pattern*: These mashup patterns can help business users in saving much time for creating new business tasks that reuse and recombine existing patterns.

### 2.1.4 Open Semantic Enterprise

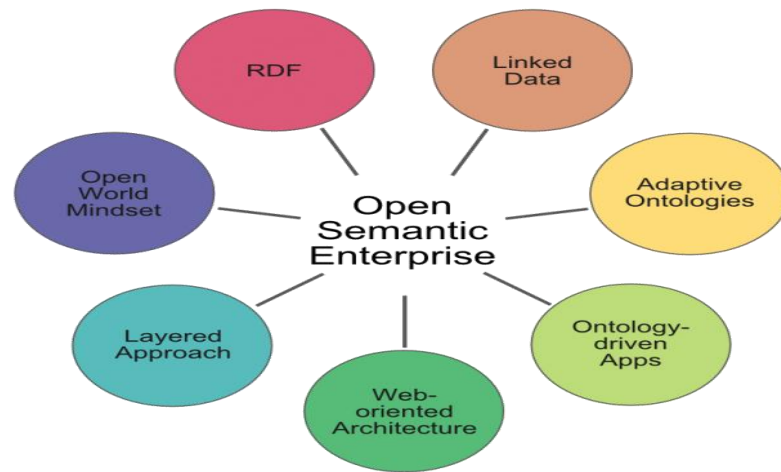
Open Semantic Enterprise (OSE) concept was proposed by K. Bergman [29] to apply for organizations that use languages and standards of Semantic Web, including RDF, RDFS, SPARQL, OWL and others. OSE aims to use Semantic Web, Linked Data and the open world assumption for integrating existing information assets and targeting knowledge management applications. The scope of OSE is in knowledge management and representation applications, which include data warehousing, data federation, business intelligence, enterprise information integration, and so forth.

Based on current understandings and still-emerging use cases being developed, K. Bergman [29] suggested seven guiding principles for OSE as follows:

- *RDF Data Model*: by defining controlled vocabularies with exact semantics, RDF can be applied to all structured, semi-structured, and unstructured data. With this expressive feature, RDF can be a useful language and data model for data federation as well as interoperability across disparate datasets.

- *Linked Data Techniques*: are applied as a method of publishing structured data in order to enable different data sources to be connected and queried for public or enterprise data, open or proprietary.
- *Adaptive Ontologies*: ontologies can be bridged with others for creating new structures and reusing useful relationships among concepts, integrating instance data, or mapping to other schema or other knowledge and domains.
- *Ontology-driven Applications*: these applications are designed modular to operate accordance to specifications, which contained in one or more ontologies, including adaptive ontologies.
- *Web-oriented Architecture*: this architecture extends SOA, wherein its functions are packaged into shareable and modular elements.
- *Layered Approach*: the conceptual architecture is layered in combination with existing assets of both internal and external data, web service layer for distributed and loosely coupled access, and Semantic technologies. This layer view is presented as Figure 2.5.
- *Open World Mindset*: in principle, this open word mindset implies that there always exist additional sources of data somewhere in the world, to be supplemented for uncompleted data at hand. The open world mindset enables incremental development, testing, and refinement.

The seven principal pillars of OSE are summarized by this following figure.



**Figure 2.4:** Principal pillars of Open Semantic Enterprise [29]

According to the idea of M. K. Bergman [29], embracing these principles of OSE can bring the following benefits in knowledge management:

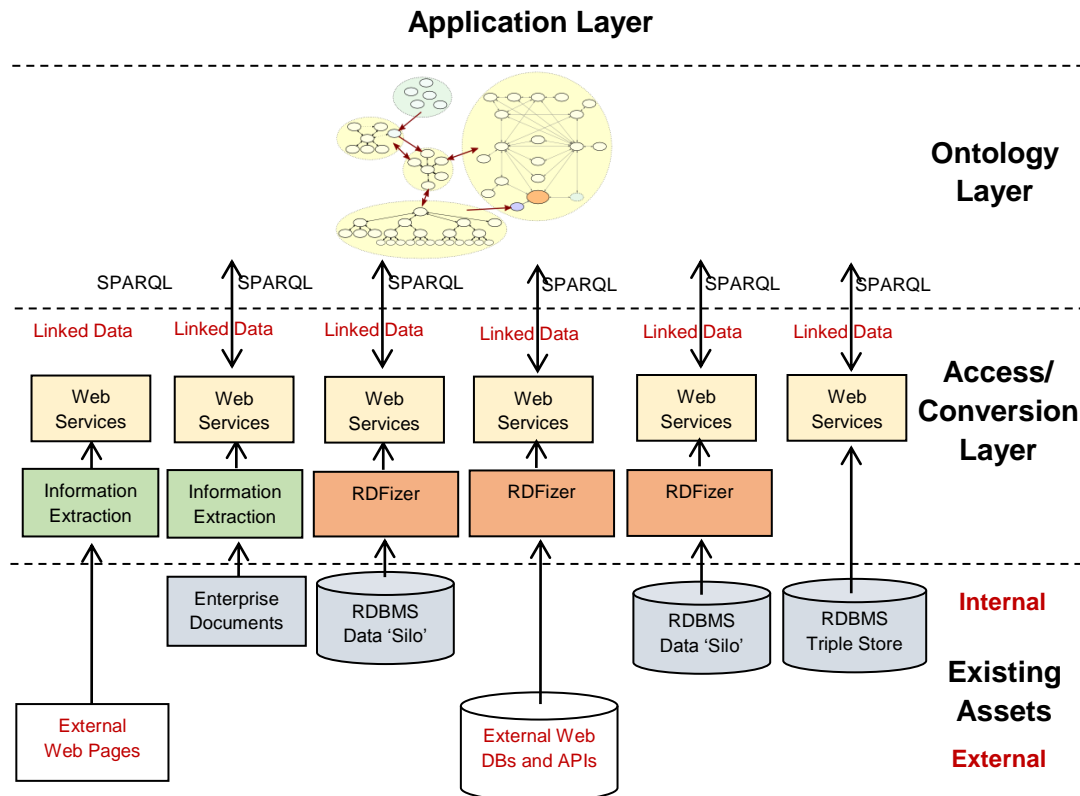
- Domains can be inspected and analyzed incrementally.
- Though schemas can be incomplete, they are developed and refined incrementally.
- Data and structures within these OSE frameworks can be expressed and used incompletely.
- Data with incomplete characterization can be combined with other complete characterization data.
- Systems, which are built with these OSE frameworks, are robust and flexible. As new structures or information are obtained, they can be incorporated without negating the existing information.
- Both closed and open world subsystems can be bridged.

The layered architecture approach in OSE includes four following major layers:

- *Application layer*: this top layer provides specific functionalities for each suitable application.
- *Ontologies layer*: this layer integrates adaptive ontologies and newly constructed ontologies to supplement the standard machine-readable purpose. This is also the effort and development of ontologies-driven applications.
- *Assess/Conversion layer*: in this layer, RDFizers or information extractors working upon semi-structured or unstructured documents exposing their information assets in RDF-ready form, which can be queried using SPARQL.

- *Existing assets layer*: the real key to OSE is to build upon appropriate architecture of existing information assets (e.g. internal database, external web data, or third party APIs, etc).

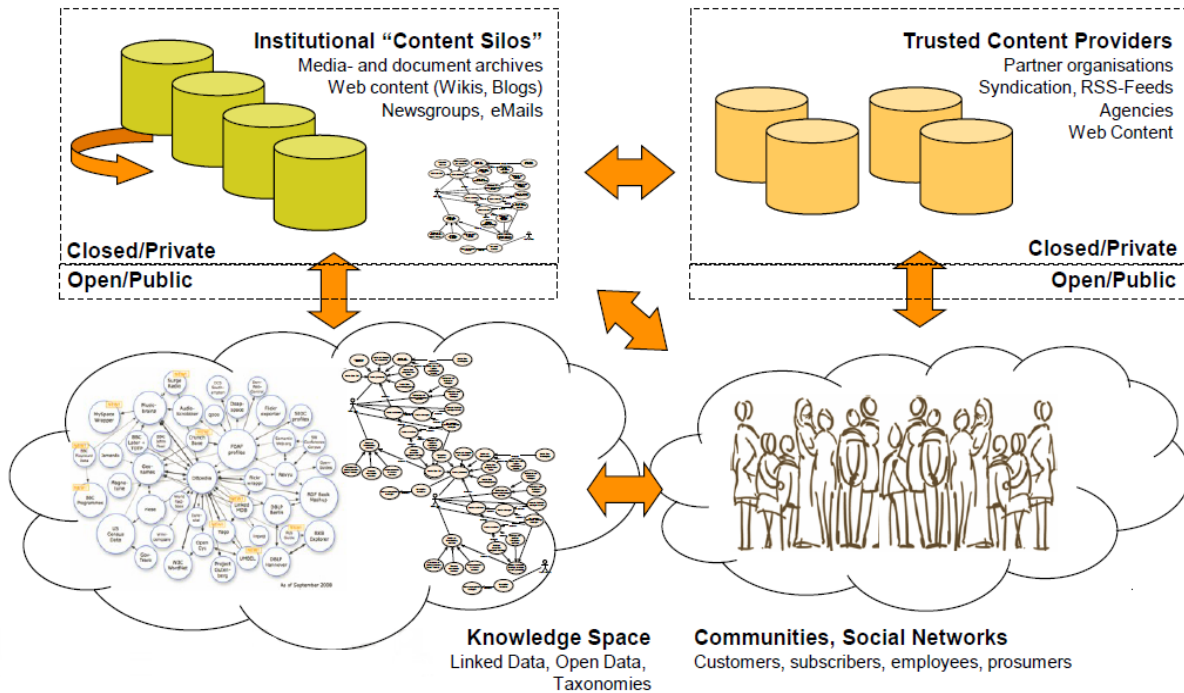
These layers are illustrated in the following architecture:



**Figure 2.5:** Layered approach in Open Semantic Enterprise [29]

Recently, Georg Güntner presented “The Open Semantic Enterprise – Enterprise Data Meets Web Data” in the recent 2<sup>nd</sup> International B2B software days in 2013 to outline the architectural and conceptual approaches to open enterprise data sources, and combine them with the Web of Data for realizing an OSE [30]. In addition, some open source tools and frameworks are suggested to be easily integrated into existing applications without replacing them. These toolsets focus on knowledge extraction (for example, Natural language processing, Entity linking and disambiguation, content classification, and metadata extraction) and networked knowledge (such as linked data platform, data federation, etc.). The implementation of the vision in OSE is presented in the next figure.





**Figure 2.6:** The implementation of the vision in Open Semantic Enterprise [30]

Although the foundational approaches to OSE do not mean open data or open source, they are suitable for these purposes with open source tools or are used for bringing external information into the business value of enterprise. In addition, the implementation of the vision in OSE can bring organizations into a 'Linked Enterprise Data' framework, a parallel idea to Linking Open Data initiative that applies the use of the Linked Data paradigm to integrating enterprise data.

### 2.1.5 Knowledge Worker

The concept of knowledge worker was introduced by Peter Drucker in the '60s [31]: *"One who works primarily with information or one who develops and uses knowledge in the workplace"*. The success of future enterprise environments requires the direct involvement of knowledge workers. In fact, Peter Drucker addressed *"Increase knowledge worker productivity is one of the biggest management challenges in the 21st century"* in his famous book "Management Challenges of 21st century" [32]. It is noticeable that productivity of knowledge workers changes widely. These changes have impacted the types of interactions in the workplace. Following research in social computing and enterprise collaboration [33], the major changes taking place in the knowledge worker environment have been considered are:

- *Distribution and globalization*: the workforce in organizations/enterprises is increasingly more distributed and more virtualized.
- *Cultural shift toward connectivity and transparency*: innovations in computing technology and SNSs have enabled people to stay more connected in their personal lives.

To accomplish a business task, knowledge workers need to retrieve not only common data sources in enterprises, but also data sources stored in personal desktops or even the Internet. It is true that knowledge workers are no longer tethered to desktops. The increasing participation of knowledge workers in SNSs provides more and more useful data sources in large-scale organizations or society activities. SNSs allow users to build and maintain online networks of friendships and relationships with friends or business partners for social and professional purposes. In SNSs, people are not just connected but share places, events, interests, and so on. These changes influence a transformational shift in how business work will be completed in the workplace.

In Web 2.0 context, a number of different types of events in a collaborative environment may occur that knowledge worker could bring benefits to an enterprise workplace. These collaborative events, which were reported by Platt [34], can be summarized as follows:

- *Content-based*: people collect and collaborate around a subject matter such as news or content, normally on a blog or a space-type environment.
- *Group-based*: people collect around an interest or idea such as a hobby and discuss it in a forum.
- *Project-based*: people work together on a project or common task such as a development project, an encyclopedia, or a book, etc.

The knowledge sharing of knowledge workers is considered as a key driver for enterprise mashup. The reason for this is that knowledge workers can provide and share their personal resources and their knowledge to other groups and other colleagues within organizations/enterprises. In addition, knowledge workers are recommended to define and build their own application in order to increase their productivity and innovation in collaboration with others.

## **2.2 Related Work**

In this work, three related subjects of this research are investigated. The first mentioned subject is to facilitate the management of personal information over Semantic Desktops. The next important one is an approach to create a semantic-based mashup prototype for proof of concept. The third subject focuses on the security issues of mashup data when building mashup applications. In this section, a comprehensive literature review of those mentioned subjects with existing solutions and approaches is presented.

### **2.2.1 Semantic Desktop**

Semantic Desktop is a new approach to PIM by creating a Semantic Web layer on a personal computer and building application on top of this layer. Most of PIM research approaches can be realized now via Semantic Desktops. In this part, some principal approaches of this area are discussed.

Haystack [35] is considered as the first major research of PIM system. It provides a platform based on RDF that manages all user information such as messages, documents, and events. DBIN [36] is another PIM semantic approach, which supports users to create personal semantic space by exchange of RDF data in peer-to-peer (P2P) network. In contrast to P2P applications, which grow the availability data storage in local, DBIN grows RDF knowledge.

seMouse [37], which is also based on semantic technologies, uses the mouse as an interactive semantic device. With seMouse, users can annotate desktop resources by pressing and sending the message of middle-button mouse click for building up OWL triple.

Gnowsis system at DFKI [38] takes existing Semantic Web technologies for integrating desktop resources into a unified RDF graph to let users manage their information in a semantic way. A refinement of Gnowsis is Nepomuk which aims to interconnect personal information with social networks and create social Semantic Desktops [39].

Open IRIS [40] enables users to create their personal maps across their knowledge work. The idea of SEMEX System [41] is to enable browsing of personal information in a semantic way by associations generated between data items on the personal desktop. SemNotes [42] is proposed as a note-taking application for creating a semantic knowledge around the notes, with emphasis on the interlinking of personal information.

SemanticLIFE [43] was developed by IFS Group, Vienna Technology University. It offers some core plug-ins to organize and manage various personal lifetimes' information in runtime by selecting Google Desktop for local information retrieval and two other plug-ins for communication with internal and external services.

These PIM and Semantic Desktop systems aim to support better personal information management and collaboration. However, they are limited only to the local storage and do not fulfill most of the requirements for personal information, especially according to a collaborative environment at the Enterprise 2.0 level.

### **2.2.2 Open Semantic Enterprise**

Some approaches have embraced the foundation of the OSE including Open Semantic Government, OpenSEA, and SemsSLATES. Open Semantic Government [44] has proposed as a branch of governments that adopts the OSE's seven principles. Also for an Open Semantic Enterprise Architecture (OpenSEA), Schaun et al. [45] proposed an architecture that combines an open semantic of ontology (TOGAF9) with a common logic (ISO 24707:2007 CL). However, this approach is mainly for semantic interoperation between enterprises. For enhancing Enterprise 2.0 ecosystem, SemSLATES [46] provides an approach that is based on Semantic Web and Linked Data technologies to enable a social semantic middleware architecture on top of existing ecosystem. Although SemSLATES also supports semantic mashup by using Exhibit [47], it focuses only on the integration of data from Web-based services.

### **2.2.3 Mashup Approaches**

In mashup approaches, a number of works have been already carried out in major projects. In this section, some of their significant features will be highlighted.

#### **2.2.3.1 Mashup Development Environments**

There are various approaches developed as mashup environments that enabled users to design mashups with an easy-to-use interface. In this section, a number of significant approaches are featured.

Simile [48] is the earliest mashup system that retrieves data from HTML pages by analyzing the DOM tree to tag retrieval data with keywords for searching or publishing later. With similar approach as Simile, Dapper [49] is improved by supporting users with an interactive screen. Dapper helps to build mashups by generating data feeds such as RSS,

XML, gadgets, or JSON, etc. from Web sites. Expanding users' ability to make mashup Web content for their personal needs or at work is the ultimate aim of Dapper system.

While Simile and Dapper require users to do the work such as data extraction, data integration manually; Yahoo Pipes [50], Jackbe [51], Automator [52] provide visualization tools that allow users to create mashups by aggregating content from various sources. Yahoo Pipes [50] is a Web-based mashup platform to mash data feeds and other services. The major objective of Yahoo Pipes is to generate data-oriented mashups. Mashups pipes are created by connecting widgets that are stored and executed on Yahoo servers. The output of these pipes can be accessed by clients as RSS, JSON or can be visualized on Yahoo Map. JackBe Presto [51] offers a robust enterprise mashup environment for small, medium and large enterprises to build internal enterprise mashup applications. This product provides some main components such as Presto Dash, Presto Studio, Presto Wires, and Presto Server that include mashup presentation, development, and processing capabilities. With Yahoo Pipes and JackBe, however, users need to express their data mashups with a specific language, such as YQL, EMMML, for a relevant editor Yahoo Pipes, JackBe, respectively.

WSO2 Mashup [53] is an open source mashup platform that hosts JavaScript-based mashups and provides the ability to consume, compose web services, feeds, and scraped web pages. The generated mashups are exposed as services, and their interfaces are described through WSDL. WSO2 Mashup is not targeted towards business users, but towards developers with knowledge of XML, JavaScript, and AJAX.

Damia [25], [54] is an enterprise-oriented mashup platform developed by IBM that mainly focuses on aggregation of data from enterprise data sources as well as Internet into feeds. Based on these combined data feeds, this platform allows business users to create mashups that are consumed by AJAX. Another end-to-end enterprise mashup platform of IBM was the IBM Mashup Center that supported rapid dynamic web application. However, this product of IBM had been withdrawn from marketing and was no longer available [55].

Exhibit [47] is a lightweight framework that aims at creating web pages with dynamic and rich visualizations of structured data from aggregated data obtained in various formats, like RDF/XML and Bibtex. Exhibit uses HTML pages as output and supports exporting its output to different formats, such as RDF/XML or Exhibit JSON.

Serena Mashup Composer [56] is a part of Serena Products Suite that specifies the front end of mashup by determining the execution of web services. However, this tool does not support the use of database yet.

My Cocktail [57] has been developed within the context of a European project named Omelette [58]. It is a web application, which provides a simple graphical user interface allowing users to build mashups by dragging and dropping various components or services within the interface. Besides, its mashups can be exported as iGoogle and Netvibes gadgets.

Another result within Omelette project is ResEval Mash [59] that presents a mashup platform for research evaluation, such as for assessment of productivity or quality of researchers, teams, institutions, and journals. However, this platform is specifically tailored to the need of data sourcing about scientific publications and researchers from the Web, aggregating them, computing metrics, and visualizing them.

Another European project is ServFace Builder [60] that is developed as a web-based mashup authoring tool. This tool supports non-programmers in design and creation of service-based interactive applications in a WYSIWYG manner. It applies the approach of service composition at presentation layer, and supports build applications by composing web services based on their frontends rather than application logic or data. In this tool, applications are designed as a set of pages that can be connected to create a navigational flow. Services of this tool are represented as form-based UIs and can be connected across pages in order to define a dataflow.

With Google mashup editor [61] or Microsoft Popfly editor [62], users can also create mashups quickly. However, these editors are no longer available.

### **2.2.3.2 Browser Extension Mashup**

Other mashup approaches are implemented as browser extensions. These tools provided a web-based personal portal to enable users to personalize mashups by adding HTML, CSS, JavaScript or data feeds. In this section, some of major approaches are discussed.

Intel Mash Maker [63], [64] is an interactive Web-based tool that supports users editing, querying, manipulating and visualizing semi-structured data; allows users to create mashups by combining content while browsing different web sites. This tool is distributed as a plugin for web browsers such as Mozilla Firefox and Internet Explorer. It also allows users to create custom mashups from a collection of widgets that can be inserted into web pages when users navigate them in the web browser.

Operator [65] is a Firefox add-on that leverages microformat and semantic data to support users in combining information on web sites by injecting semantic data into HTML. PiggyBank [66], [67] is another Firefox add-on that allows users to extract data on websites by turning users' Firefox browser into a mashup platform. Tabulator [68] is developed as both a Firefox

add-on and a web-based application that supports users in browsing the web of RDF data. Piggy Bank and Tabulator illustrate personal mashup tools with generic functionality that let users satisfy their own mashup needs. While Tabulator supports only RDF data, Piggy Bank can extract data from HTML pages via web screen scrappers.

D.Mix [69] supports users in creating scripts to copy annotation data instead of traditional copying of the whole web page elements to be used later while browsing web sites. Abiteboul et al. [70] introduced a mashup model with their basic component called mashlet. Their mashlets can query data sources, import other mashlets, use external web services, and specify complex interaction patterns between its components. Their models can facilitate dynamic mashlets composition, interaction, and reuse.

Ikeda et al. [71] proposed a mashup framework that provides GUI components to support users in browsing mashup items. This framework consists of a data management engine and a widget library that allow users to adopt demand-driven data creation and interactive widgets to browse mashup data.

Vasko et al. [72] introduced a model-driven approach to integrate different domains into Service Mashups design that included orchestration information derived from WS-BPEL processes, coequal integration of RESTful/ WS-\* Web services and role-based collaboration of process participants. SAP Research [73] proposed a lightweight platform Enterprise Mashup Application Platform (EMAP) to realize mashup applications. EMAP is a browser-based application composition environment and runtime. Its model is based on the creation and composition of reusable components with associated metadata via an exchange mechanism called Event Hub.

The mentioned approaches seem to be suitable to the current mashup context. However, those systems on the web require users with advanced knowledge of XML, JavaScript, CSS, HTML, or Microformat.

### **2.2.3.3 Semantic Mashup**

There are a number of proposals that adopted Semantic Web to support query and creation of data mashups. In this section, their considerable features are going to be presented.

iSPARQL [74] is a utility for building and executing SPARQL queries to fetch data. With iSPARQL, users with prior knowledge of SPARQL can build the query with triple patterns, optional patterns, or depiction of UNION queries. Potluck [75] is a tool for mixing data by merging chosen fields of RDF. With Potluck, users need to manually map attributes between

sources and specify data integration. These kinds of tools require users having prior knowledge of SPARQL, RDF, or MIME types.

ONKI [76], SA-REST [77] are other approaches that use ontologies to add semantic annotations to their RESTful services or HTML Pages. ONKI provides developer widgets that can utilize ONKI ontologies and services with a few lines of JavaScript code adding to the HTML page. SA-REST allows the use of OWL or RDF to represent their ontology and describe the service by embedding RDFa annotations into the HTML pages.

JOPERA [78] presents a layered architecture for mashup design separating the integration logic from presentation logic, in which the former can deal with bottom-up and top-down service composition and the latter is based on AJAX technology. Mashlight [79], which is a lightweight framework for generating and executing mashups, provides users with a simple method to create “process-like” mashups using “widget-like” Web 2.0 applications. This tool allows the creation of mashups that are composed of several interacting widgets. MashArt [80] is another mashup development platform, including a web editor based on advanced user interaction mechanisms that allow the integration of data, services and UI components through a unique language and a mashArt component library.

sMash [81] presents a semantic-based mashup navigation system by constructing and visualizing data API network in three steps, including discovery, removing and visualization. Their data APIs with the corresponding metadata are stored in the role of nodes in a graph, sMash will find the corresponding mashup graph based on the APIs’ metadata properties and present to the UI. Due to matching against the user query with APIs metadata, this process requires users to know sufficiently IT knowledge in advance.

MatchUp [82] proposes an approach, which designs mashups based on a novel auto completion mechanism. This approach exploits the similar classes of mashup components that are designed by different users. When users select a mashlet in UI panel, the system suggests the corresponding glue pattern. For this reason, the recommendation system is limited to only providing glue patterns.

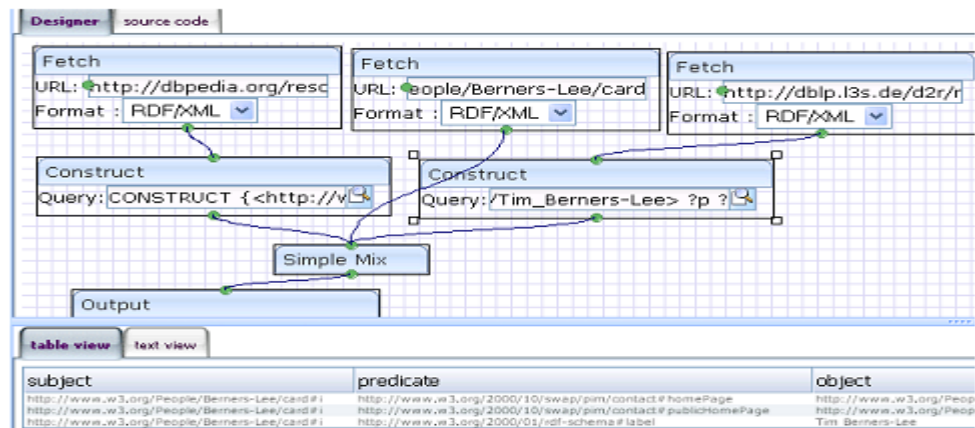
SPARQLMotion [83] is a graphical notation tool with an RDF-based scripting language that supports users in describing data processing pipelines. But this tool provides one vocabulary and one hard-coded execution engine for representing SPARQLMotion scripts.

DERI Pipes [84] is a better engine and graphical environment for general web data transformations, which is inspired by Yahoo Pipes. It is an open source project that is used to build RDF-based mashups. A pipe is written in an XML-based language that is defined by a specification and invoked by an HTTP GET request via the DERI Pipes execution engine.



DERI Pipes support SPARQL, XQuery, and several scripting languages, which are used to transform the fetched records, like an RDF resource. The output streams of data (XML, RDF, and JSON) are enabled to display directly the results in Web browsers, or are accessible via a URI and can be used in other applications.

The following figure depicts a mixing of three data sources in DERI Pipes with some SPARQL expressions and operators.



**Figure 2.7:** An example of DERI Pipes [84]

The following table shows a survey of current popular mashup tools with different functionality features.

	Yahoo Pipes	Dapper	Intel MashMaker	WSO2 Mashup	Jackabe	DERI Pipes
Owner	Yahoo	Dapper	Intel	WSO	Presto	DERI
Initial release	February 7, 2007	2005	April 22, 2008	January 28, 2008	March 2010	December 12, 2009
Current Status	Beta	Now part of Yahoo	Past project and has been retired	Migrated to WSO2 Application Server.	Version 3.6 with free 30 day trial.	0.7
Technology	Standard web, YUI	Standard web	Browser plug-in	Standard web	Standard web	Standard web

Drag & Drop feature	√	Missing	Possible but missing connected widget	Missing	√	√
Client Side	None	None	√	None	None	None
Server Side	√	√	None	√	√	√
Programming Skills	Average	Average	Non-programming	Expert	Expert	Average

**Table 2.2:** Status and features comparison of popular mashup tools

#### 2.2.3.4 Data Mashup Language

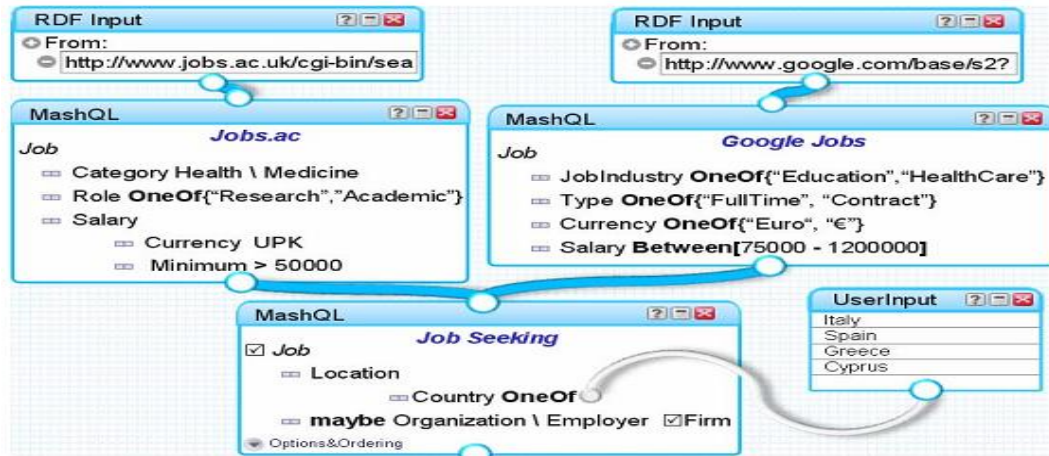
Besides researching into an easier way of mashups, mashup tools have also proposed some mashup languages for performing programming logic and presenting data. These mashup languages are characterized by several approaches that focus on different aspects of mashups development. Some languages are data-oriented, XML-based languages, such as MashQL, Enterprise Mashup Markup Language (EMML), and Open Mashup Description Language (OMDL), which can be used like programming languages.

Mostarda and Palmisano [85] presented a novel approach based on hybrid functional and logic programming language in writing web mashups. They also presented a JSON-based scripting language called MU, which allows aggregating and manipulating data sources over multiple external sources.

Yahoo Query Language (YQL) [86] is an expressive SQL-like language that allows users to query data across web services or data from web pages. YQL also supports developing data mashups by retrieving and manipulating data from APIs through a single web interface. However, YQL facilitates data integration or data extraction, but does not intend to create a complete mashup application.

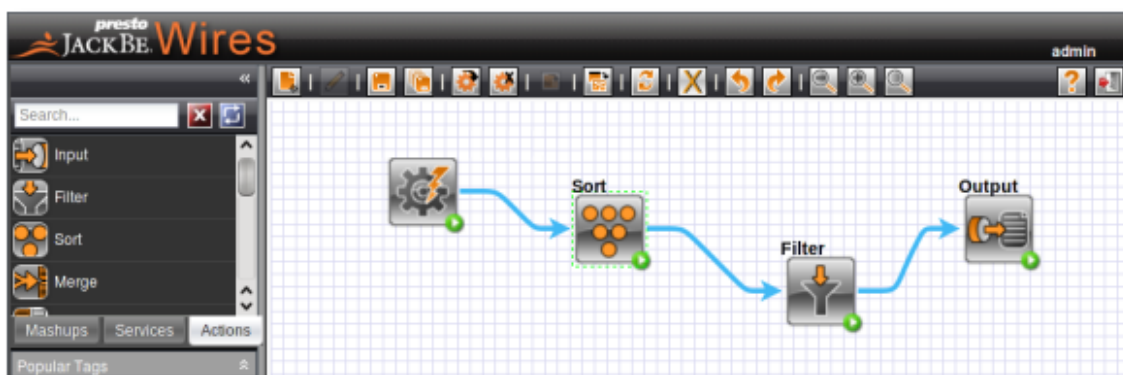
MashQL [87] is another query-by-diagram language that allows people to query, build data mashups, and pipeline RDF data on the Web. The idea of MashQL is allowing users to navigate and query RDF graphs without prior knowledge about their vocabulary, schema, or technical details. Queries in MashQL are parsed into and executed as SPARQL queries automatically. MashQL markup is developed based on XML to serialize MashQL pipes in interchangeable and textual format. MashQL pipe markup consists of some main elements to

represent metadata and input sources about the pipe itself. The output of MashQL queries can be rendered into certain formats, such as XML, HTML, or RDF input to other MashQL queries. The following figure illustrates an example of MashQL queries.



**Figure 2.8:** An example of MashQL Queries [88]

Enterprise Mashup Markup Language (EMML) [89] is another promising mashup language in enterprise environments. It is an XML-based language, which is developed by the Open Mashup Alliance (OMA). The EMML is an open language for development of enterprise mashups and describes the processing flow for a mashup. A runtime engine is required to interpret EMML statements of a mashup script. EMML provides a rich set of high-level mashup-domain vocabularies to mash flexibility a variety of Web data-sources (namely XML, JSON, JDBC, and Java Objects). EMML also provides a uniform syntax to invoke different service styles (such as REST, WSDL, RSS/ATOM, and RDBMS). Although EMML is an expressive language, it is quite complicated and difficult for users to apply. Besides, EMML has not supported ontology or customized ontology yet; it only supports constructing SPARQL queries and executing them. The next figure depicts a mashup sample of JackBe with advanced functions such as sorting, merging, filtering, etc.



**Figure 2.9:** Mashup creation with JackBe Presto [51]

Open Mashup Description Language (OMDL) is a part of OMELETE project that supports exporting mashup consisting of pages, layouts and widgets for importing into another platform such as Apache Rave – an engine that aggregates and assists web widgets [90].

Contributing to those mashup languages in enterprise mashup platform, Messias et al. [91] provided a way to invoke cross-domain SOAP web services by using client side languages. With a Domain Specific Language (DSL) to describe web mashups, Swashup (Situational Web Applications Mashups) [92] extends the Ruby on Rails architecture [93] to develop mashups. Swashup DSL provides statements that contain various concepts to model and describe web mashups.

In the current context, existing mashup tools and languages provide a mashup window containing SPARQL queries; RDF data sources, and web feeds; as well as some specific mashup languages that export mashup content consisting of pages or require end-users already having some IT-knowledge such as data feeds and RDF.

#### **2.2.4 Mashup Security**

The use of Web 2.0 applications inside organizations/enterprises has created additional security challenges. In order to ensure trustworthy resources and mashup data in enterprises, mashup security issues have to be taken into consideration and improved in the collaboration and personalization context. These issues have been investigated by several related works as follows.

Hotta et al. [94] proposed the use of web content personalization and collaboration simultaneously but without exchange of personal data. In order to support secure cross-domain communication for web mashup developers, OpenMashupOS (OMOS) [95] has been proposed as an extension of Mozilla Firefox that handles web pages as objects and allows

objects to communicate each other via their declared public interfaces. It can be configured to be backward compatible with same origin policy.

SMash [96] was proposed a secure component model where different trust domains can create components of content, and interact via a communication abstraction. Hasan et al. [97] presented a component controlling dataflow within a mashup by using a permit-based authorization delegation service named 'Permit Grant Service' that enables fine stateless access control and authorization in mashups.

Jonas et al. [98] presented a security lattice-based method to mashup security. The security lattice is built from the origins of mashup components and inferred directly from mashup itself. Besides, they proposed a mechanism that allows origins to specify escaped hatches for declassifying objects. Sqwelch [99] identified the gap between consumers and enterprise by providing semantically-enabled mashup makers and trusted collaborative environments which enable composition of mashups based on a concept of trust explicitly specified by users through a visual interface.

Matthias et al. [100] presented a proof-of-concept implementation that enables the secure usage of a mashup by protecting sensitive data against malicious widgets and operators. Heidelinde et al. [88] proposed an approach by modeling and defining security rules for mashup compositions in their own notations, and automatically evaluating submitted mashups for compliance with the respective policies.

Zibuschka et al. [101] proposed reversed identity-based encryption within a public key infrastructure to realize a secure mashup-providing platform (MPP). They used the new definition MPP as a Web-based server-side platform offering APIs and hosting functionalities that enable the creation of mashups.

## **2.3 Summary**

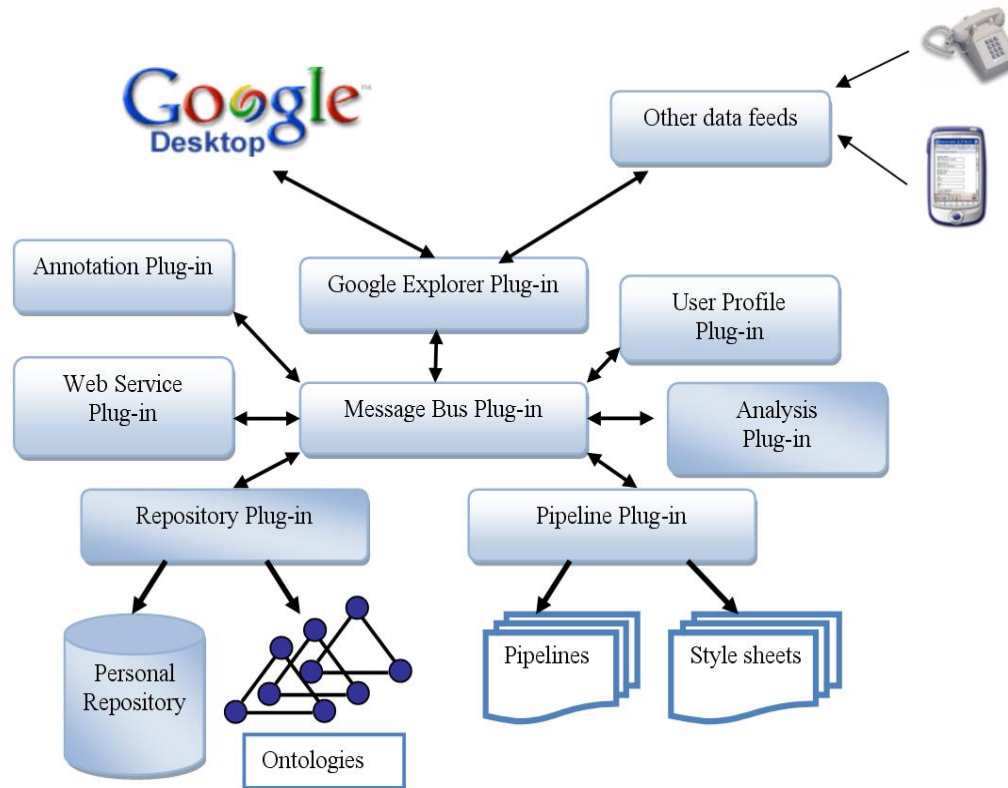
This chapter has summarized the current contexts of a numerous related work, which include Semantic Desktops, approaches to semantic mashups, and the security issues of mashups. Although these works have been proved to be useful, they are still limited and under development. Besides, the theoretical background of some domains such as Social Web, mashup, Linked Data, knowledge worker and Open Semantic Enterprise are also presented. This research is taking full advantage of these technologies to carry out our research. Next chapter will explore the data sharing of knowledge workers and proposes a solution to prepare mashable artifacts for our mashup system.

## MASHABLE PERSONAL RESOURCES AND SERVICES

People have the tendency to share their knowledge or resources, which are not only stored locally on their personal computers or isolated data repositories, but also transferred to SNSs. With this new generation of Web 2.0, people interact with SNSs by expressing their profiles, schedules, plans, and activities in an interoperable and extensible way. This chapter will explore the personal data sharing in their Semantic Desktops and SNSs platforms. From that point, the solutions that turn their heterogeneous sources into mashable artifacts are investigated. This chapter also aims to bridge the gap between Semantic Desktops and SNSs in order to integrate and reuse existing personal resources in an application. It also expands the scope of our Semantic Desktop system – SemanticLIFE [13] in particular - and Semantic Desktops in general into the web of data instead of isolated data silos.

### 3.1 Personal Resources in SemanticLIFE

SemanticLIFE is an attempt for PIM systems that has been developed by Information Software Engineering Group, Vienna Technology University to store, organize and manage various personal life items [13]. It provides a repository of lifetime personal data from varied resources such as email messages, browser web page, images, contacts, phone calls, life events and other resources. SemanticLIFE framework offers various core plug-ins such as Semantic Store, Message Bus and Web Service to manage the user profiles in static and dynamic way as well as long term activities. The whole SemanticLIFE framework is depicted as the following figure.



**Figure 3.1:** SemanticLIFE framework [12]

In SemanticLIFE, semantic store is responsible for lifetime personal data retrieval by selecting Google Desktop as a desktop search application for information retrieval and adding the appropriate semantic context in an ontological way to the index data of Google Desktop's repository. The two other plug-ins are responsible for providing a uniform access layer to internal and external services and their semantic, including personal and global services for service composition scenarios. Each personal resource retrieved in SemanticLIFE can be seen as an entity in the RDF graph, where a <subject, predicate, object> triple indicates a direct URI from a subject node to an object node, and the predicate indicates the relation between subject and resource.

The SemanticLIFE framework is another effort to come a step closer to Vanevar Bush's vision of the Memex: *'A memex is a device in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory'* [102]. However, the major limitation of Semantic Desktops approaches is restricted to users' desktops and precious semantic information is not yet effectively used in business processes and tasks that people deal with in their workplace and daily life [12].



### 3.2 Personal Resources in SocialLIFE

SNSs are platforms that allow users to create and maintain online networks of friends or business associates for professional and social reasons [103]. In addition, SNSs allow users to surf their platforms for various purposes such as finding people with common interests, finding jobs with relevant skills, or publishing messages to target users, etc.

In this work, the term ‘SocialLIFE’ is used to denote one’s lifetime information in SNSs, in which personal resources are his/her activities (messages, comments, tweets, etc.), interests (books, movies, etc.), and related connections (friends, colleagues, etc.). To be more specific, one’s SocialLIFE consists of interconnections of people relations such as friendship or business/professional relationships on Facebook or LinkedIn; their interests such as video, image or music on YouTube, Flickr or MySpace and so on. The following figure depicts recent various platforms that people are able to use for their SocialLIFE.



**Figure 3.2:** The most popular SNSs in 2013 via the conversation prism [104]

In some SNSs, their published contents are in structured formats or annotated with a number of Semantic Web vocabularies for expressing personal profile or social networking information such as FOAF, hCard (an open microformat for publishing people, companies,



and organizations on the web). Most of social software provides APIs for access and interaction. Those APIs define a generic set of service methods and functionalities to exchange data. For instance, Flickr, Twitter, or Facebook API allows users to access most of their data by using REST or HTTP-based web services. With these features, SNSs have served as useful platforms for linking or reusing heterogeneous data of one's SocialLIFE for performing operation or aggregation.

### **3.3 Mashable Personal Resources**

#### **3.3.1 Linking Personal Resources with LOD Cloud**

As noted before in section [2.1.2](#), Linked Open Data (LOD) aims to interlink data on the Semantic Web and plays an important mechanism for information management and integration. In order to bridge the gap between Semantic Desktops and Linked Data, several research projects are conducted [105], [106]. These researches have the common goals to combine resources from Linked Data and Semantic Desktops by using semantic metadata as a common denominator, and to enrich the linked data within the personal information as well as enterprise space.

To fully benefit from LOD cloud, it is crucial to put personal resources into a context that interlinks resources and enables powerful personal services. In combination with SNSs, it is useful to provide better linking content with different services, in which the shared information is not just textual content but also multimedia content such as music, videos, or pictures.

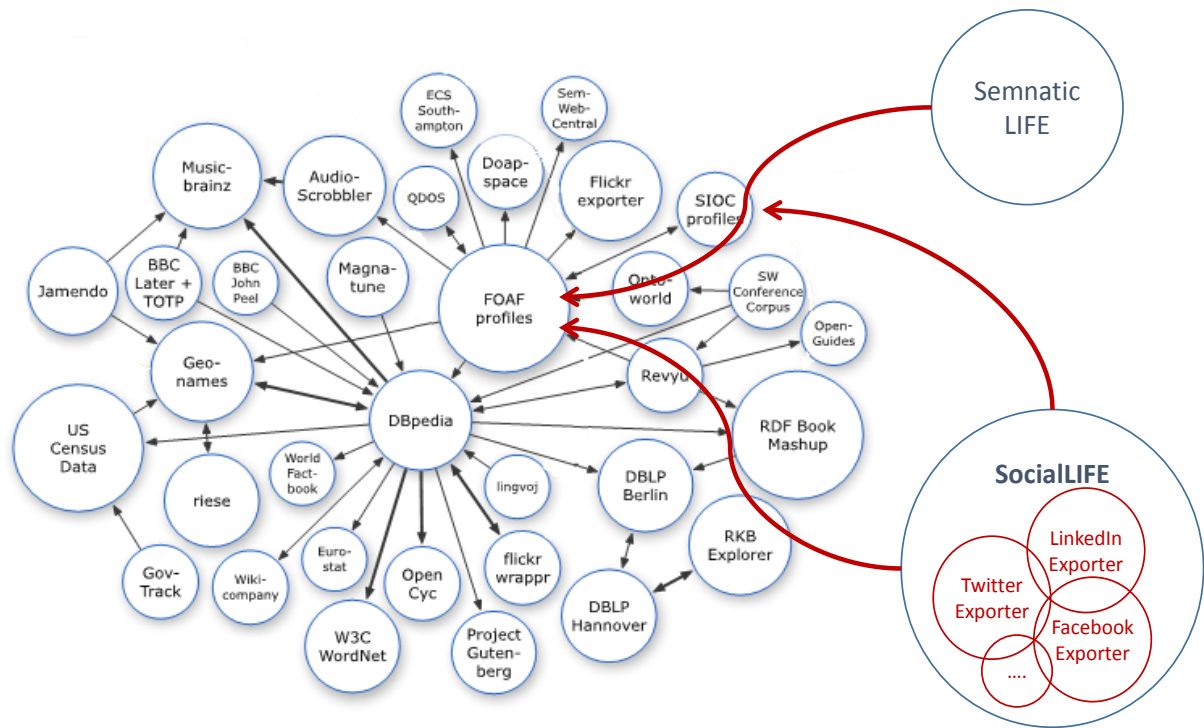
There are a number of vocabularies, which enable data sharing by using Semantic Web for linking personal resources with LOD cloud. The FOAF project [107] can be considered the first Social Semantic Web application that combines RDF technology with data in SNSs. FOAF profiles are used by many people, including researchers and professionals, as a means to be a machine-readable ontology for describing personal/professional information, activities and relations. With photo-sharing services from Flickr, there are some exporters such as FlickrRDF exporter [108], Flickr2RDF [109], and FlickrWrappR [110] that allow users to export their Flickr connections in construction with FOAF profile, extend DBPedia with RDF links to Flickr's photos, and parse metadata from Flickr's photo into the RDF description respectively.

With music-related content, MusicBrainz [111] – an open music encyclopedia - provides the links between DBPedia and artists, or Jamendo [112] – a digital service platform for free music – provides the links between MusicBrainz and GeoNames for geo location information. These types of information from the music metadata can be represented with Music

Ontology, FOAF and GeoNames vocabularies that enable both people and machines to have meaningful conversations about music. Semantically-Interlinked Online Communities (SIOC) aims to enable the information integration of online community, and is commonly used in combination with the FOAF vocabulary for expressing social networking and personal information. In the enterprise context, this combination plays as an entry for modeling individuals, teams, relations between individuals and teams; linking people to their interests and skills; representing activities of online communities with related content; and identifying unification across enterprise applications.

By following above remarks, the existing ontology models such as FOAF, SIOC, Geonames, etc., are reused to benefit from and build semantic mashups, as well as to extend SemanticLIFE ontology and support information integration with one's SocialLIFE. The built-in OWL properties such as owl:sameAs and rdfs:seeAlso are also re-used to link an individual to another individual. For instance, these properties can be used to state that a resource (an instance of a class) is related to another resource in an open dataset such as DBpedia, FreeBase, DBLP Bibliography Database and so on.

The following figure depicts the linking of personal resources in SemanticLIFE and SocialLIFE with LOD cloud in this work. It might be considered that consuming FOAF profiles provide a first step towards solving the issues of data portability between semantic applications in SNSs.

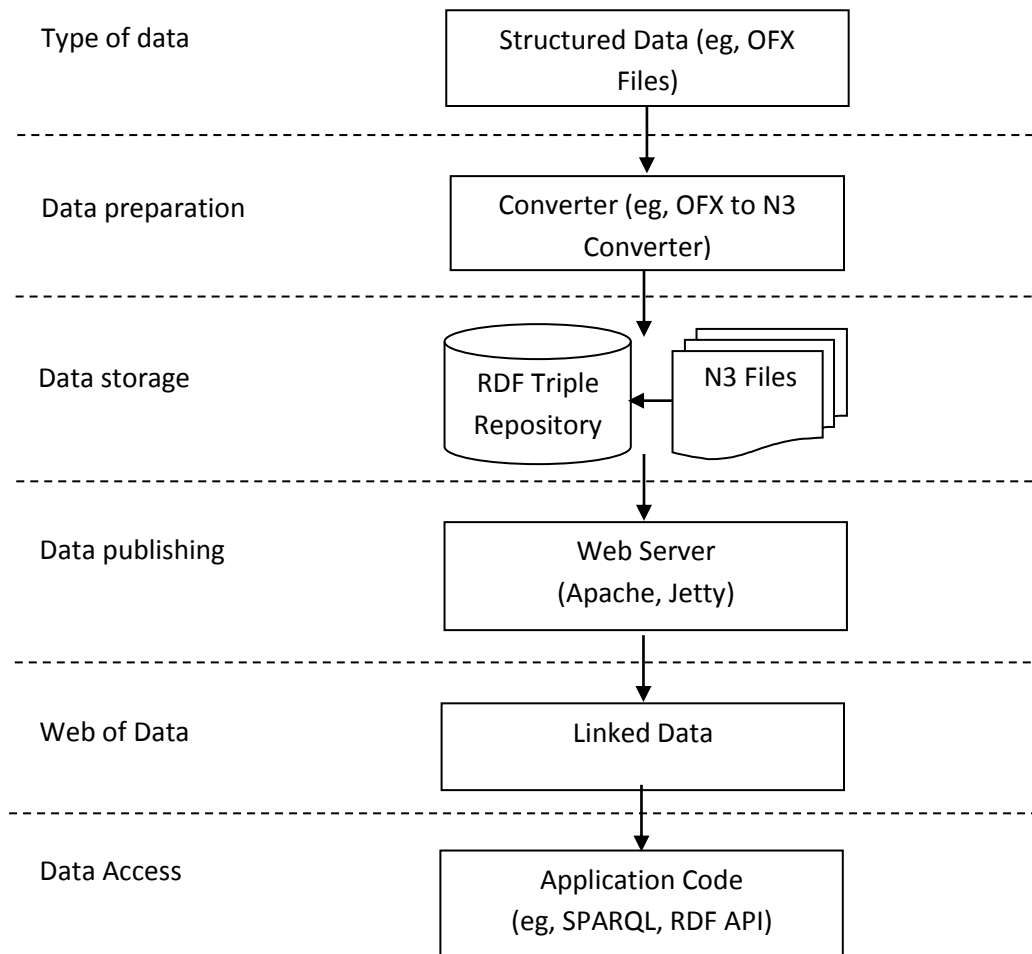


**Figure 3.3:** Linking SemanticLIFE and SocialLIFE with LOD Cloud

According to the guidelines in preparing data for LOD [17], publishing the personal resources into LOD is conducted in five simple steps as follows:

- Find some data with reuse potential, describe and give context as machine-readable structured data.
- Use URLs to identify information resources so that others may point to them.
- Plan for persistence, i.e. Persistent Uniform Resource Locator (PURL)
- Publish data on the Web in XML, RDF, or even comma-separated values.
- Create an online catalog of published data so that others can find and reuse it.

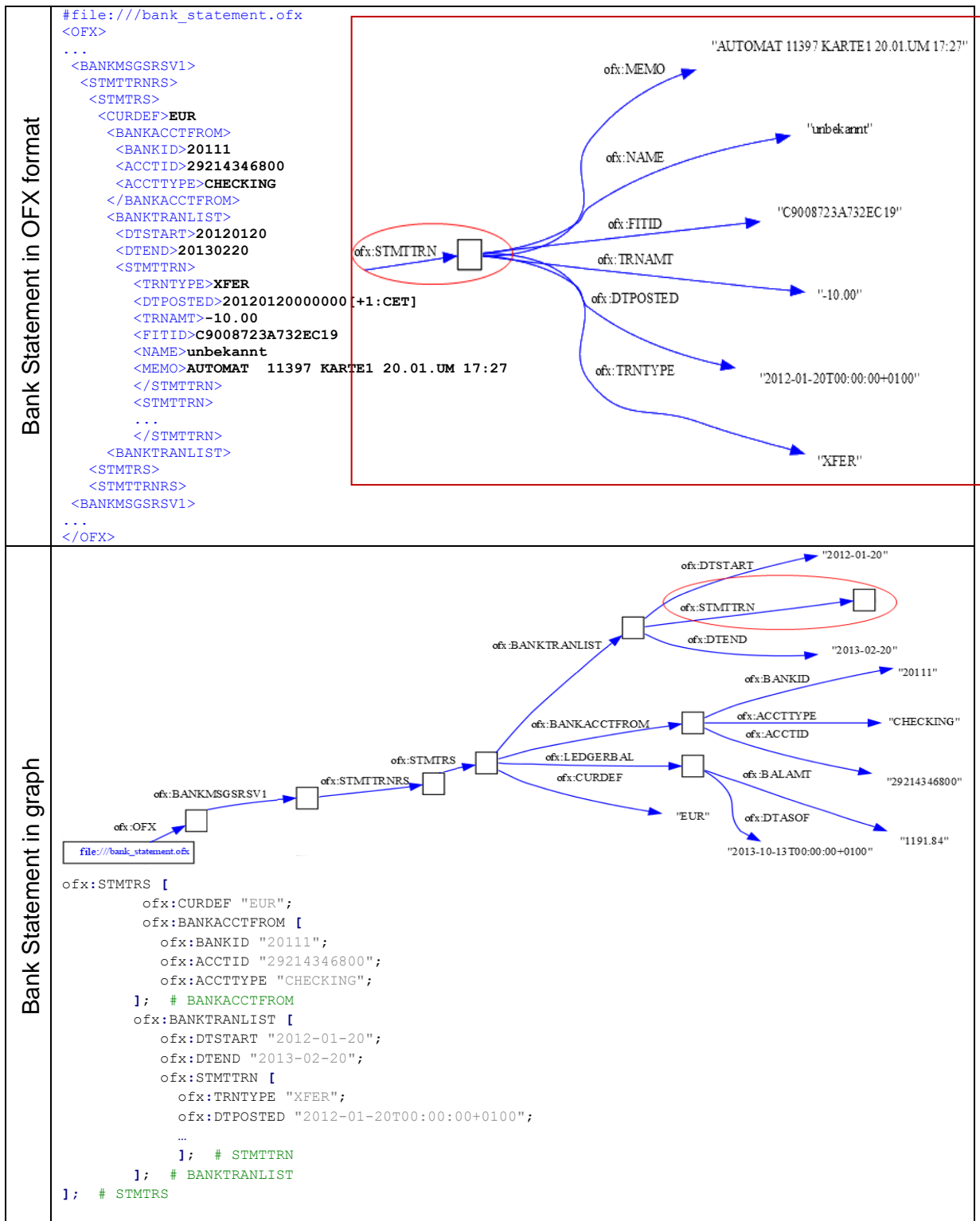
The architecture for publishing data as Linked Data can be illustrated in the following figure.



**Figure 3.4:** Architecture for publishing data as Linked Data

To illustrate this section, a motivating use case about personal financial data (such as bank statements) is realized. Many banks allow customers to use personal financial management software or APIs to download their bank statements in OFX format. Open Financial Exchange (OFX) is a unified specification to exchange electronic financial information between financial institutions, consumers and businesses via the Internet [113].

In order to turn personal financial data into mashable artifacts, users' OFX data should be converted to RDF/N3. The following figure shows a sample format of the bank statement from account <ACCTID> of bank <BANKID> that has some transactions <BANKTRANLIST> and their relevant bank statements <STMTRN>. Each bank statement has its own id <FITID>, statement name <Name>, amount <TRNAMT>, etc.



**Figure 3.5:** Conversion of financial data from OFX format into RDF/N3

### 3.3.2 Semantic-based Personal Resources Retrieval

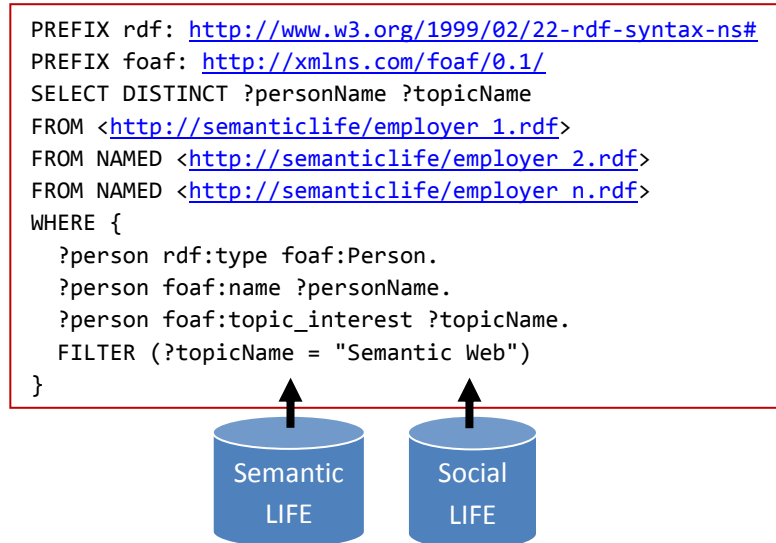
In this section, by using FOAF as a common representation format and in conjunction with RDF data, the personal resources can be retrieved from various semantic data sources. SemanticLIFE ontology can be further extended by adding the SocialLIFE entities. The following sample presents the representation of social data and mashable personal resources in RDF, in which a part of user information retrieval from SemanticLIFE and SocialLIFE would be achieved through their FOAF files.

Personal resources in SemanticLIFE	<pre> &lt;?xml version="1.0"?&gt; &lt;rdf:RDF xmlns:foaf='http://xmlns.com/foaf/0.1/' xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'&gt;   &lt;foaf:Person rdf:ID='http://voque.org/saokhue_foaf.rdf#Sao_khue'&gt;     &lt;foaf:firstname&gt;Khue&lt;/foaf:firstname&gt;     &lt;foaf:family_name&gt;Vo Sao&lt;/foaf:family_name&gt;     &lt;foaf:mbox rdf:resource='mailto:saokhue@ifs.tuwien.ac.at' /&gt;     &lt;foaf:birthday&gt;22/01/1981&lt;/foaf:birthday&gt;     &lt;foaf:gender&gt;male&lt;/foaf:gender&gt;     &lt;foaf:Document rdf:about='http://ifs.tuwien.ac.at/slfe- google#item=file://C:\Khue-CV.doc' /&gt; </pre>
Personal resources in SocialLIFE	<pre> &lt;foaf:holdsAccount   rdf:resource='https://www.facebook.com/profile.php ?id=1802357483' /&gt; &lt;foaf:holdsAccount   rdf:resource='http://twitter.com/#!/saokhue' /&gt; &lt;foaf:interest&gt; &lt;rdf:Description rdf:about='http://en.wikipedia.org/wiki/ Enterprise_social_software'   rdfs:label='Enterprise_social_software' /&gt; &lt;/foaf:interest&gt; &lt;foaf:currentProject&gt;   &lt;foaf:Project&gt;     &lt;foaf:homepage&gt;       http://www.sba-research.org/research/ data-security-and-privacy/secure-20/     &lt;/foaf:homepage&gt;   &lt;/foaf:Project&gt; &lt;/foaf:currentProject&gt; &lt;foaf:workplaceHomepage rdf:resource='http://ifs.tuwien.ac.at' /&gt; &lt;/foaf:Person&gt; &lt;/rdf:RDF&gt; </pre>

**Figure 3.6:** RDF representation of personal resources in SemanticLIFE and SocialLIFE

These personal resources can be used or accessed directly using SPARQL – a RDF query language [114]. The results of SPARQL query will be the input data for the other services in mashup process.

The following figure describes an example of SPARQL query language for extracting the social network matching by FROM NAMED clause:



**Figure 3.7:** SPARQL query towards data mashups for SemanticLIFE and SocialLIFE

### 3.3.3 Semantic-enabled Personal Services

Based on the idea of personal web services [12], each person has his/her own services that provide an intuitive way for sharing information with the outside world, helping to identify and transact with the appropriate remote business processes. In an effort to describe semantic for services, SA-REST [77] tries to annotate the HTML documentation of services. However, this solution needs the developer's decision in choosing which HTML pages to annotate. EXPRESS [115] exploits similarities between REST services and Semantic Web, such as realization of resources, self-describing representations, and uniform interfaces between client and server. With the approach of EXPRESS, an OWL ontology describing resources and relationships between them for Web Services must be provided.

Towards the combination of Linked Data and services technology, there are two existing approaches: Linked Data Services (LIDS) [116], [117] and Linked Open Services (LOS) [118]. LIDS is proposed for integrating data-providing services with Linked Data, which leads to data silos that are opened up to the Web of Data and enables the automatic integration of links to LIDS with existing datasets. LOS is defined to wrap existing services with descriptions based on SPARQL and RDF. While LIDS provides HTTP URIs with encoding parameters as key-value pairs in the query string, LOS consumes RDF as output. Unlike LIDS and LOS that are built based on SPARQL, RESTDesc [119] is built on N3. With LIDS, large amounts of data

can be transformed for using on the Semantic Web and enable (semi-)automatic service discovery and integration.

All gathered information in Semantic Desktops and SNSs platforms have to be parsed into semantic personal services or mashable semantic-based resources that provide a semantic way to express and exchange information from heterogeneous resources. Some SNSs can be accessed by using the API services; these technologies are used to provide a common and machine-readable model of metadata for content.

Realizing its potential, LIDS is applied as a suitable solution in this research, in which using the simple vocabulary of LIDS that defines and describes the relevant service. The principles for applying LIDS [116] are as follows:

- Describe input and output of services as SPARQL graph patterns.
- Communicate RDF by Restful content negotiation.
- The output should make explicit its relation with the input.

The following figure expresses a LIDS description for a common service:

```
@prefix lids: <http://openlids.org/vocab#>
LIDSDesc a lids:LIDS;
lids:lids_description [
  lids:endpoint ENDPOINT ;
  lids:service_entity ENTITY ;
  lids:input_bgp INPUT ;
  lids:output_bgp OUTPUT ;
  lids:required_vars VARS
] .
```

↑  
APIs and RESTful Services

**Figure 3.8:** LIDS description for APIs and RESTful services

Following [116], the LIDS syntax is explained in the following way:

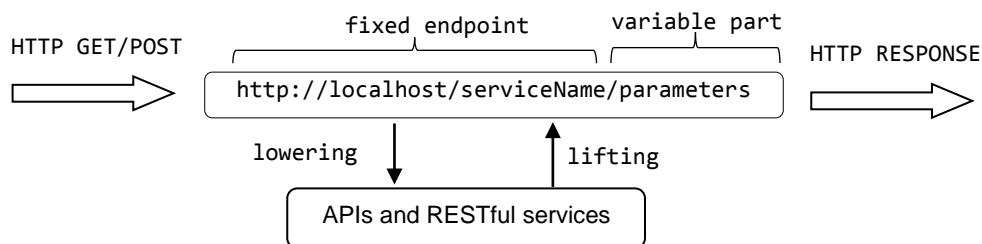
- LIDS is a resource representing the described Linked Data service.
- ENDPOINT is the corresponding URI.
- ENTITY is the name of the entity.
- INPUT and OUTPUT are basic graph patterns encoded as a string using SPARQL syntax.
- VARS is a string of required variables separated by blanks.

One of the advantages of LIDS approach is a method for semi-automatically build semantic models of Web APIs, including lowering and lifting concepts. When lowering



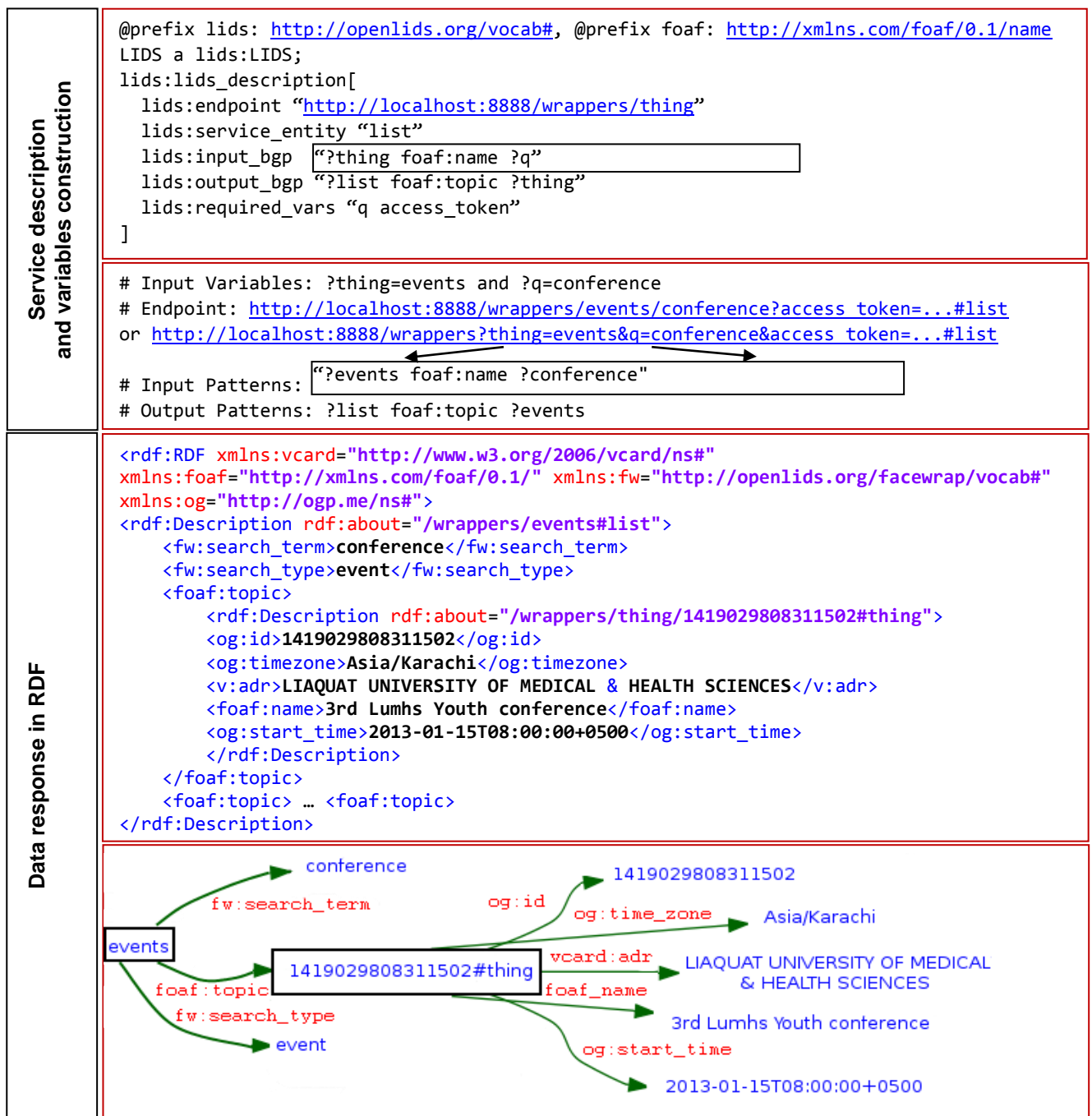
referred to the process of constructing non-semantic data for API calls (i.e., converting RDF to input data), lifting referred to the process of constructing semantic data out of a non-semantic API or service response (i.e., converting output to RDF before delivering results).

The lowering and lifting for all APIs and RESTful services are depicted as follows:



**Figure 3.9:** Lowering and lifting in LIDS for APIs and RESTful service

To illustrate this section, a service for searching personal events in SocialLIFE is created, which has the URL <http://localhost:8888/wrappers/thing>. In the lowering side of this service, the variables become part of URL service as a query string, in which the variables binding of SPARQL query “SELECT ?thing WHERE {?thing foaf:name ?q}” become “?thing” and “?q” variables of the service URL. In the case of searching “conference” events, the triple of input variables, which construct the SPARQL query “?thing foaf:name ?q”, are “?events foaf:name ?conference”. In the lifting side, this service takes the name and the query variables to form the service URL “<http://localhost:8888/wrappers?thing=events&q=conference>”. If the client accepts RDF format, the service result will return RDF data. This example is illustrated more details in the next figure.



**Figure 3.10:** An example of semantic-enabled personal services

### 3.4 Summary

In this chapter, some solutions are presented for turning personal resources from heterogeneous sources into mashable artifacts. These solutions also aim to bridge the gap between Semantic Desktops and SNSs in general, and between SemanticLIFE and SocialLIFE in particular. The next chapter will investigate the trustworthiness issue of mashup data; and self-monitoring mashup data when design mashups.

## TRUSTWORTHINESS OF MASHUP DATA

It has been observed that mashups bring new security threats on mashup-based business processes in organizations/enterprises. These issues have been the subjects of several works [96], [120]–[122]. The requirements of trustworthiness of mashup data in this mashup system are also taken into consideration in this study. These requirements could be:

- *Self-monitoring and disambiguation resources*: it is obvious that so far there is no optimistic solution to avoid completely an unwanted information disclosure with the data sharing on Web 2.0 by the owners. They include the membership in SNSs, blog entries, contributions on Wikipedia, shared media and virtual games. For these reasons, the issue is how to help users identify trustworthy mashup data.
- *Trusted resources (Data feeds, API, etc.)*: the resources are submitted, having lifetime and shared by users. These resources should be validated by users; otherwise untrusted resources will put other users at risk unintentionally.
- *Widget owner*: the sharing widget is decided by the widget owner. Users run a risk when creating mashups with others' shared widget. It is required that the shared widget could be guaranteed of containing trustworthy data as well.

This section provides the overall approach of our project Secure 2.0 – Secure the information sharing on Web 2.0 [123]. One of the main contributions of this project is a prototype based on the Self-Organization Map that extracts and classifies provided content on Web 2.0. This research aims to integrate recent research efforts from Semantic Web, Web Service area, and Word Sense Disambiguation techniques with semi-structure Web 2.0 content to achieve more accurate semantic annotation of text and mashup result.

### 4.1 Self-Monitoring in Social Networking Sites

The Web 2.0 knowledge extraction and its applications demonstrate the power of collaborative work, and how it can be used to assess collective data resources. The quality of

such data resources is being improved via statistical and analytical methods of community behaviors. An interesting point in such collective knowledge systems is the vertical view that brings together the contributions of a specific user/organization and makes inferences about the behavior of user/organization. This vertical view, which is also referenced as Gigantic Join, brings many benefits for Web 2.0 and Mashup architecture, but on the contrary, threatens the user's privacy by disclosing the inferred facts about an individual or an organization [124]. For instance, the vertical view of a person who publishes YouTube videos plus his/her social networks might trigger a false positive alarm for relevant supervisors. There are two major concerns in such binary classifications:

- First of all, it is important to note that the ratio of an out-of-favor group (such as terrorists, child abusers, etc.) to the normal people is very small. If the sensitivity (the proportion of people that tested positive of all the positive people tested) of classifier is selected to be very high there will be lots of false positives. Consequently, many normal people will be classified as out-of-favor and specificity (the proportion of people that tested negative of all the negative people tested) decreases.
- Another important issue is the fact that as soon as a person is incorrectly blacklisted (false positive), it would be very difficult (if not impossible) to remove him/her from the list. In other words, there is a greater tendency putting the people in the black list rather than removing them from the list.

In order to hinder such scenarios, it is required an effective mechanism that can bridge the gap between information domains and aggregate value from a mix of structured and unstructured data. Due to the high degree of sensitivity that is used these days for classifiers, it is essential to have a self-judgment tool that can be used by individual/organizations to correct their facade to the public and disable incorrect interpretations about themselves. The knowledge extraction from Web 2.0 entries can be of great importance in many cases. In order to clarify the applications, a number of such use cases will be discussed in this section.

One of the interesting applications of content analysis is assistive services. The specific Web 2.0 content should provide assistance for users who create similar content. In other words, by analysis of existing content, some targeted templates and information structure patterns will be established. These templates and patterns will be used to provide ad-hoc suggestions for common content and structure for the given context. It is important to note that the assistive services are not allowed to share sensitive data with other users; instead, they would share the templates and general structure of information. A good example for this

group of services is the bookmark annotations that can be suggested to users according to the favorite tags of a specific web page.

Another use case in knowledge extraction is the resource sharing. In Web 2.0 communities and especially in SNSs, some information is being shared with other users. The “data sharing” in SNSs environments is decided by content owners and there is no holistic solution to avoid the unwanted information disclosure. There is an ever growing need for intelligent sharing of information based on the content of shared items, users’ relationships, type of users, and personal-organizational sharing policies.

The other group of use cases, which is the center of attention in this section, is self-monitoring of trust level. The data contributed by users on Web 2.0 is a good resource that can reflect the individual/organizational behavior and attitude. For example, the membership in SNSs, Facebook profiles and friend networks, shared pictures and videos, and virtual games, are all together a rich set of information that can be used to judge people or organizations. In some cases, these inferences are not correct and the individuals and organizations have no means to prevent false judgments. To address this issue, some technical and social issues will be discussed in the following sections.

All scenarios mentioned above have a common basic requirement, namely the automatic conceptualization of the targeted content. Subsequently, the conceptualized results can be used to provide assistive services, facilitate the resource sharing, or on a higher scale, be combined with other data resources and used for self-monitoring purposes.

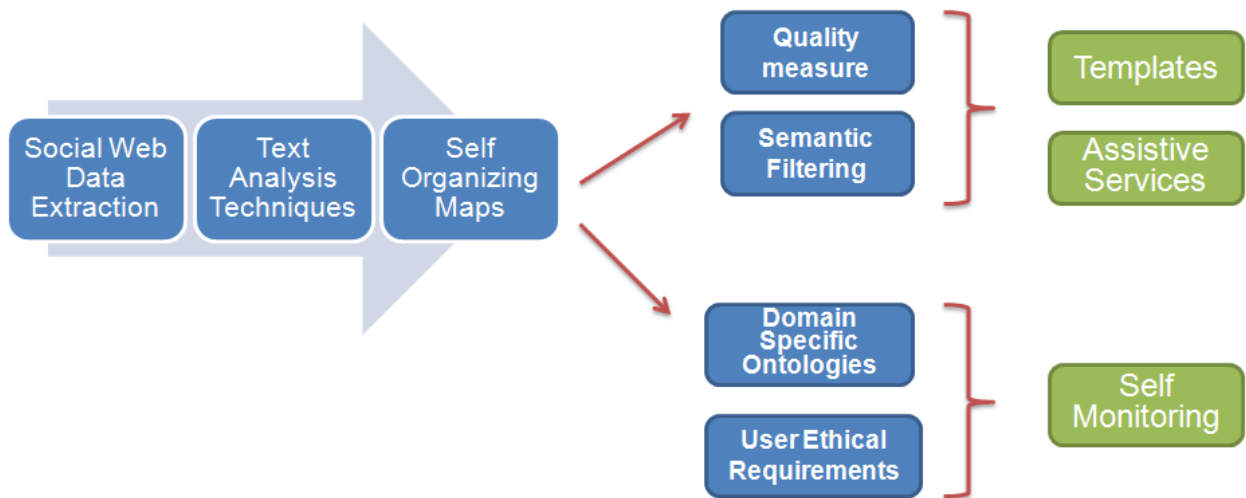
In this approach, we have tackled the automatic conceptualization challenges of Web 2.0 content by applying Semantic Web technologies. The outcomes of this approach will provide a solid basis for addressing the assistive services and resource sharing scenarios.

This approach can be summarized as the following steps:

- The data will be extracted from social web platforms either by an API of a target platform (such as Facebook API, Twitter API, etc.) or via a dedicated extractor component.
- Text analysis techniques (such as Word Sense Disambiguation) are applied to disambiguate and annotate the text with useful semantic information (Section [4.1.2](#)).
- Self-Organizing Maps (SOM) (Kohonen, 2001) are used to visualize the result and give the user an overview of his/her social network context (Section [4.1.3](#)).
- After this step, quality measures are applied to find out high quality entries that have the potential for being used as a template for assistive services.

- Another outcome of the SOM is a self-monitoring result by applying user-ethical requirements and highlighting the points of interest on the resulting SOM in a given context.

The following figure demonstrates the different steps of our solution to address the requirements of the use cases mentioned above.



**Figure 4.1:** Overall solution for self-monitoring in social network [124]

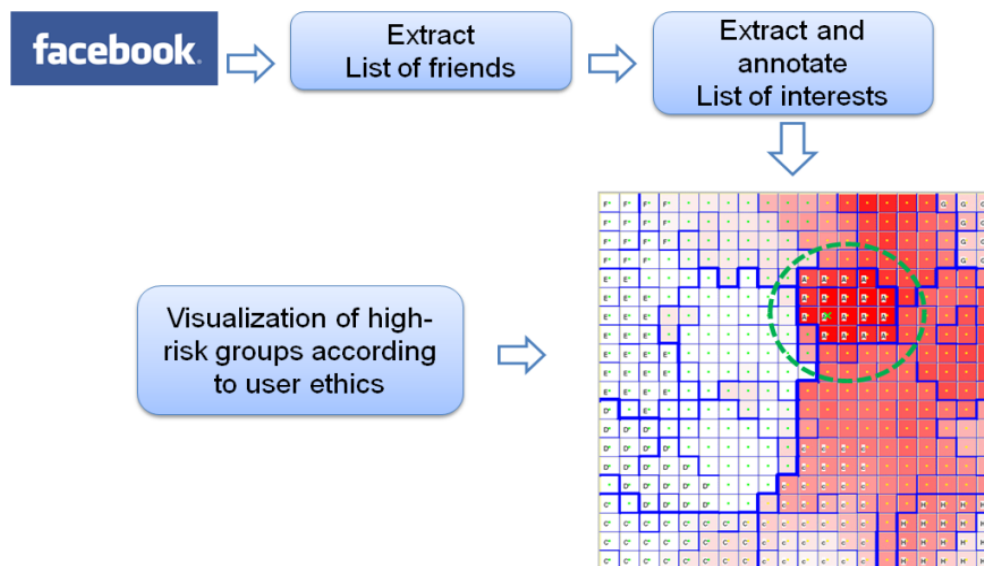
As Unhelkar et al. [125] mention trust is also directly related to ethical issues in both society and business. Ethical issues are very important, especially in a society that is constantly influenced by the rapidly changing of the technology. Ethics are the principles used to determine the purpose of organizations/enterprises' decisions.

In this research scope, it is also required that all personal resources and shared data are in secure and trust suitable for either user ethics or organization policy. User ethics can be classified as follows:

- Personal ethics
  - People have their personal preferences and lifestyles.
- Professional ethics
  - Professions have some ethical values that should be followed.
  - Organizations may also have some restrictions that should be followed by their employees.
- Social ethics
  - The community that people are living in may demand and require some social behavioral values.

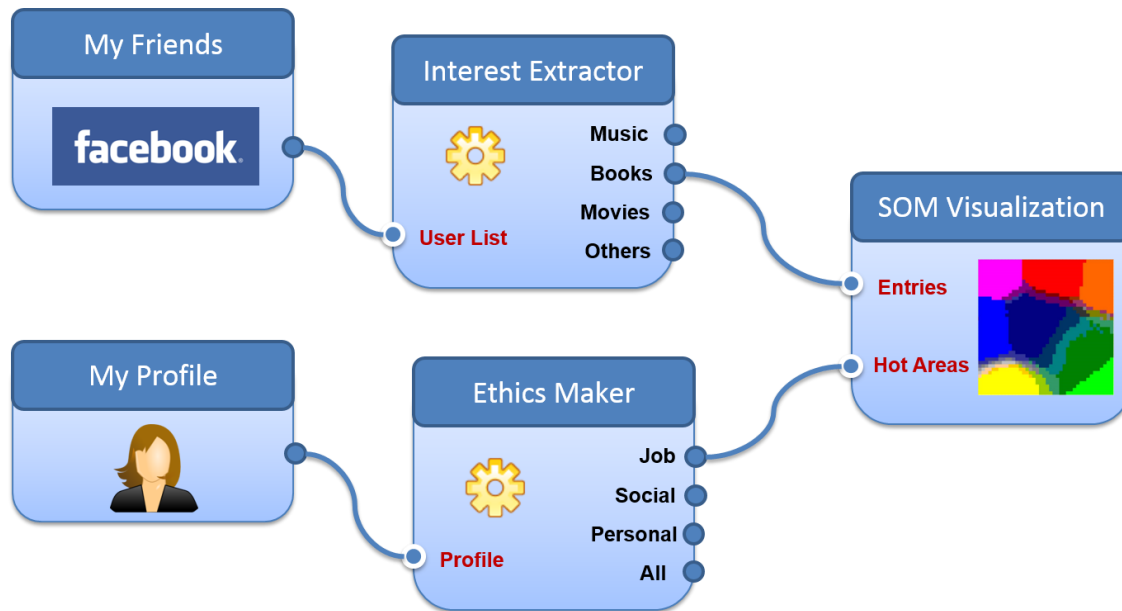
At the end of SOM process, a group of data points is merged and formed clusters. The clusters are labeled and might be further improved for creating the ontologies. An important note at this step is about security and privacy issues. After analysis of documents, sensitive contents of nodes should be removed according to domain ontology. After creating SOM, the points of personal resources need to be specified on the SOM map with corresponding ethics in the private and working lives as well. In this approach, the points of interest are those areas that are violating the above user ethics.

These ethics are encoded in user profiles, which can be combined with the semantic information of SOMs. For instance, the job ethics might prohibit the user from sharing or contributing discussions about a specific topic. In such cases, violating areas will be highlighted on the SOM, and the user may react and take the necessary actions to correct the situation. The following figure demonstrates this use case where the high risk group of friends is highlighted on the interest map of friends.



**Figure 4.2:** SOM visualization of high risk group on the friends' interest map [124]

These developed components are planning to integrate with the proposed mashup architecture in section 5.1 to facilitate user interactions. The following figure demonstrates a simple mashup to create a SOM of all books that friends of a specific user are interested in. According to job ethics of interest book of this user, the map is then highlighted to find out inappropriate connections in his/her Facebook profile.



**Figure 4.3:** Mashup solution to create a SOM visualization of high risk group of the friends' interest in Facebook [12]

In the same way, the hot areas in the SOM Visualization of organizations/enterprises may not want to be linked to sensitive data such as internal circulation documents or positive advertisement campaign of competitors' products. The identification of out-of-favor mashup data is the responsibility of users or organizations/enterprises and this decision is left up to them.

Mashup solutions facilitate the collection, integration, and publishing of data for non-expert users. As a result, the mashup solutions can be seen as new data sources that may feed other mashups, applications, services, or websites for personal/business use cases in an easy way. The further explanation of mashup solution will be presented in [Chapter 5](#). In the rest of this section, each of the steps mentioned above will be discussed in more detail.



## 4.2 Knowledge Sharing Policies

The requirement of security and trusted privacy is a critical issue in personal information management. This requirement is keeping on increasing when personal/organizational information in Web 2.0 via SNSs and digital contributions are disclosed. In Information Security and Assurance Conference, one of the major research challenges in leading cyber security is that the ability to give end-users security controls and privacy that they can understand and control for the dynamic, pervasive computing environments of the future [126].

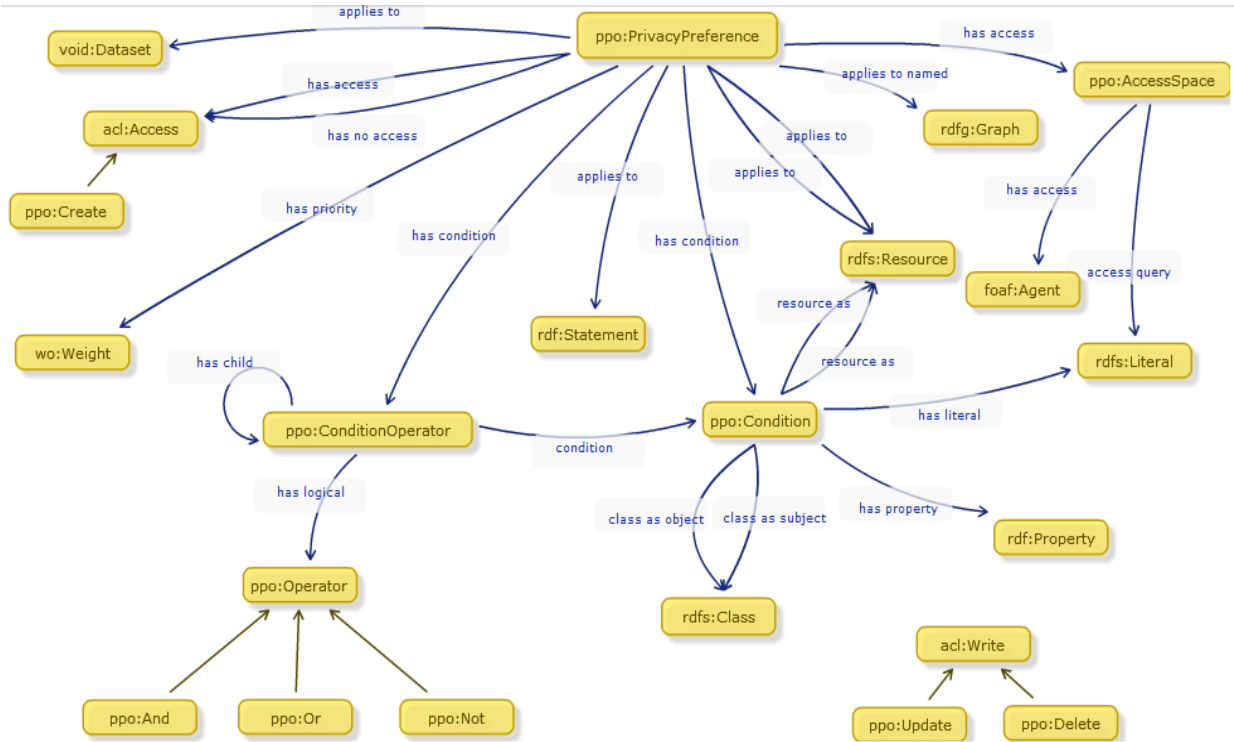
In order to facilitate knowledge sharing and reuse in an efficient and trustworthy way, it is required to support the processes of information sharing by applying appropriate policies. In addition, several domain ontologies and an efficient mechanism should be defined to enrich personal/organizational profiles, and provide privacy ontology for applying personal/organizational policies in filtering the sensitive data. Regarding to policy, for example, a privacy policy can be *“a statement or a legal document that discloses some or all of the ways a party gathers, uses, discloses, and manages a customer or client's data. Personal information can be anything that can be used to identify an individual, not limited to but including: name, address, date of birth, marital status, contact information, ID issue and expiry date, financial records, credit information, medical history, where you travel, and intentions to acquire goods and services”* [127]. The contents of this privacy policy will vary depending upon the data privacy, data protection, and applicable law. A lightweight Privacy Preference Ontology (PPO) has been proposed to enable users and prevent sensitive information from publishing linked data [128]. Like many other existing ontologies, this ontology can be further extended by adding the required concepts and entities.

In this research, policies are defined based on ontology PPO and combined with user profile in Semantic Desktop's ontologies for sharing knowledge including:

- *User ontology*: comprises the user profile, personal context and the data that might be used in the rule-making process.
- *Privacy ontology*: defines the generic concepts of an information sharing domain, such as personal information (name, address, ID issue, etc.).
- *Organization policy*: defines the organizational policies and obligations for information sharing to restrict any sensitive data such as international hiring policies, new product release policies, etc.

- *Service ontology*: describes the services, including personal services, and the description of the information that will be disclosed using such services.

The following figure depicts the privacy ontology, which supports users in setting their preference for sharing knowledge.



**Figure 4.4:** The main classes and properties of the Privacy Preferences Ontology [128]

### 4.3 Exploiting Disambiguated Information Retrieval

In Web 2.0 context, people easily annotate the shared content by free-form tagging with keywords, tags or hashtags that act like related-categories or related-topics. This feature makes the shared content more easily discoverable and browse-able by other users. However, in order to retrieve this information, these issues of free-form tagging's, such as tagging ambiguity (i.e., difference meaning), tagging heterogeneity (i.e., acronyms, synonyms, abbreviation, etc.), should be solved. Without the explicit semantic context, the process of analyzing data and putting the data to work in a business process safely is still impractical without significant human involvement and this is the point that Semantic Web can be applied to make the data machine process-able.

In order to explicitly exploit the information retrieval in a semantic way, WordNet ontology (WordNet) [129] and Word Sense Disambiguation methods [130], [131] will be utilized in conjunction with semi-structured Web 2.0 content to achieve a more accurate semantic

annotation of the personal resources [132]. Concurrently, applying the Self-Organizing Maps (SOM) [133] to automatically cluster similar inputs, which are mapped close to each other to categorize a group of any specific criteria, such as a group of experienced workers for a new project, etc.

SOM provides a unique mechanism of clustering, through which a large amount of text data is organized into a small number of meaningful clusters. It is important to note that, similar resources are clustered together if they share similar concepts, and some concepts may carry more weight compared to other concepts that appear in the same resources, based on the concept frequency of those resources. In order to improve the quality of SOM, the concepts need to be disambiguated first.

#### **4.3.1 Word Sense Disambiguation for Mashable Resources**

This research has conducted an improved Word Sense Disambiguation (WSD) method that combines the existing WSD techniques with semi-structured Web 2.0 content to achieve more accurate semantic annotations of personal resources. WSD has been considered as a fundamental research problem in machine translation [134] and in artificial intelligence [135]. It is a task of determining the sense of an ambiguous word in a given context of surrounding words, phrases, and sentences. For instance, the word “library” can be understood as “library building” or “software library” according to its context and surrounding information. By considering the context of this term and applying WSD techniques, this word can be clustered together with other relevant terms. Fortunately, the senses of English words can be easily extracted from free lexicon dictionaries such as WordNet that has been developed by Princeton University [136]. Words in WordNet are organized in hierarchy and semantically instead of alphabetically. Each node consists of a synset of words, that express the same or a closely related word. These synsets are linked together by semantic relations, such as hypernymy, hyponymy (nouns and verbs), meronymy (nouns), and antonymy (adjectives).

Almost all WSD methods require some common preprocessing steps that parse the input text and prepare it for further method-specific processes. These common pre-processes include actions such as tokenizing, stemming, and finding the meaning of the words in dictionaries or lexicons such as WordNet.

There are numbers of WSD methods for defining the sense of words in a given context. One group of such methods relies mainly on gloss-based and path-based calculations. These methods, which are also referred to as Lesk-based methods, are listed below:

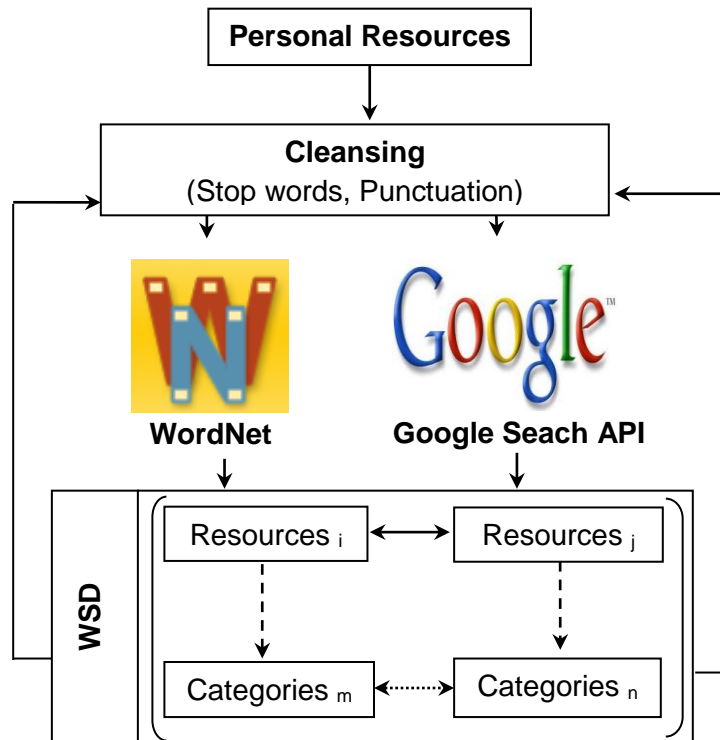
- Lesk algorithm [137] disambiguated word senses by finding their gloss in traditional dictionaries and calculating its overlap with the glosses of its surrounding words.
- Kilgariff & Rosensweig [138] applied the original Lesk and also considered the overlaps between a word's gloss with its context.
- Banerjee and Pederson [131] applied the original Lesk algorithm and used the WordNet instead of traditional dictionaries.

Some other measures, which are categorized as semantic-based approaches, compute the path distance between two concepts based on the semantic organization of corresponding words in well-known taxonomies, i.e. WordNet, in order to detect the sense of a given word. Some of the methods applying this approach are listed below:

- Leacock & Chodorow method [130] measures the similarity of word senses based on their shortest path and their maximum depth of WordNet taxonomy in which similarity score is calculated as  $-\log(\text{length}/2*d)$  where :
  - length is the shortest path between two concept  $c_1$ ,  $c_2$  (relation "is a „") and
  - d is the maximum depth of the taxonomy
- Wu & Palmer method [139] measures the depth of the two concepts in WordNet taxonomy, the depth of least common subsumer (LCS), and finally combines these figures into a similarity score as  $2 * d(\text{lcs}) / [d(c_1) + d(c_2)]$ , where:
  - $d(\text{lcs})$  is the depth of the least common subsumer (LCS) and
  - $d(c_1), d(c_2)$  are the depth of concept  $c_1$ ,  $c_2$  respectively

Among WSD methods, the algorithms of Lesk family are very suitable for the sense disambiguation of single words but they do not have the same effective when applied to a larger window of words. In contrast, semantic similarity measures compute the distance between the hierarchies of related synsets in WordNet and provide better results. The concepts of hierarchies and their weights have been used to align the ontology with domain ontologies, calculation of neighborhoods, and semantic distance of different resources.

The following figure shows the text processing process followed by cleansing and applying the WSD measures for personal resources. The final output should be annotated categories that can be used for further knowledge extraction and analysis processes.



**Figure 4.5:** Word Sense Disambiguation for mashable personal resources

In this approach, a hybrid method to accomplish the WSD is applied. In the first step, Lesk algorithm is used to find the correct sense of a word according to its context. At the end of the WSD process, words will be annotated using the most appropriate sense of words in the given context, based on WordNet senses. The annotation tags of words will be then created by combining the words and adding the correct sense to it. As a result, all words will be uniformly annotated by a well-defined taxonomy.

However, it is noticed that some words do not exist in lexicons. As an example, the commercial or technical words such as Mashup, Flickr, or Servlet cannot be found in WordNet. To address this issue, for those cases that the word is not found in WordNet, it is possible to search that word via Google Search API [140] and the first result will be considered as its gloss. In most cases, this approach will provide usable results for the next steps. For instance, for the term Mashup, WordNet returns no result; however, the first result in Google says: *"In web development, a mashup is a web page or application that combines data or functionality from two or more external sources to create a new service. ..."* that is

helpful for our purposes. In order to make the data usable, a lot of preprocessing tasks like removing of unwanted symbols, words, html tags, xml tags, punctuation marks, numeric, and stop words have to be done.

As a result of this solution, the annotation tags of words will be then created by combining the words (stem of words) and adding the correct WordNet categories to it, and all resources will be uniformly annotated by a well-defined taxonomy. The final outputs of the proposed approach are annotated resources that can be used for further actions (such as knowledge extraction and analysis processes), and express the rationale and semantic quality of mashup data. The outputs will also help to address more complex use cases such as security and privacy scenarios that need a deeper understanding of the text and its context for applying the appropriate security and privacy policies.

#### **4.3.2 SOM-based Personal Resources Clustering**

In this approach, Self-Organization Map (SOM) has been used as a powerful method for automatic clustering of high-dimensional statistical data, in which similar inputs are mapped close to each other. Based on this property, SOMs can be visualized easily, and the physical distance between concepts will depict the similarity of concepts concerning some predefined features of items in the domain under study.

A typical SOM algorithm for classification of text-based items can be summarized as follows [141]:

- Initialize input nodes, output nodes, and connection weights:
  - o Use the top (most frequently occurring)  $N$  terms as the input vector and create a two-dimensional map (grid) of  $M$  output nodes.
  - o Initialize weights  $w_{ij}$  from  $N$  input nodes to  $M$  output nodes to small random values.
- Present each document in order to:
  - o Describe each document as an input vector of  $N$  coordinates.
  - o Set a coordinate to 1 if the document has the corresponding term and to 0 if there is no such term. Each document is presented to the system several times.
- Compute distance to all nodes: compute Euclidean distance  $d_j$  between the input vector and each output node  $j$ :

$$d_j = \sum_{i=0}^{N-1} (x_i(t) - w_{ij}(t))^2$$

where  $x_i(t)$  can be 1 or 0 depending on the presence of  $i$ -th term in the document presented at time  $t$ . Here,  $w_{ij}$  is the vector representing position of the map node  $j$  in the document vector space. From a neural net perspective, it can also be interpreted as the weight from input node  $i$  to the output node  $j$ .

- Select winning node  $j^*$  and update weights to node  $j^*$  and its neighbors:
  - o Select winning node  $j^*$ , which produces minimum  $d_j$ .
  - o Update weights to nodes  $j^*$  and its neighbors to reduce the distances between them and the input vector  $x_i(t)$ :

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)(x_i(t+1) - w_{ij}(t))$$

After such updates, nodes in the neighborhood of  $j^*$  become more similar to the input vector  $x_i(t)$ . Here,  $\eta(t)$  is an error-adjusting coefficient ( $0 < \eta(t) < 1$ ) that decreases over time.

- After the network is trained through repeated presentations of all documents, assign a term to each output node by choosing the one corresponding to the largest weight (winning term).
  - o Neighboring nodes, which contain the same winning terms, are merged to form a concept/topic region (group). Similarly, submit each document as input to the trained network again and assign it to a particular concept in the map.

In this approach, SOMToolbox [142] is used to apply this SOM algorithm. SOMToolbox is an open source library for training SOM with different visualizations and quality measures. For using this toolbox, two SOM vectors must be provided for the training process of a SOM:

- *Input Vector file*: this file describes the input vectors that consist of the following primary parameters:
  - o \$XDIM: is the number of input vectors in file.
  - o \$YDIM: usually 1; this allows again form XDIM\*YDIM to provide the total number of vectors.
  - o \$VEC\_DIM: is dimensionality of vectors (weight vectors of map).
  - o Lists  $n$  vector elements of  $m$  weight vectors where  $m=XDIM$  and  $\langle VEC\_ID \rangle$  is the label of weight vector
    - $\langle x\_1\_1 \rangle \dots \langle x\_1\_n \rangle \langle VEC\_ID\_1 \rangle$
    - ...

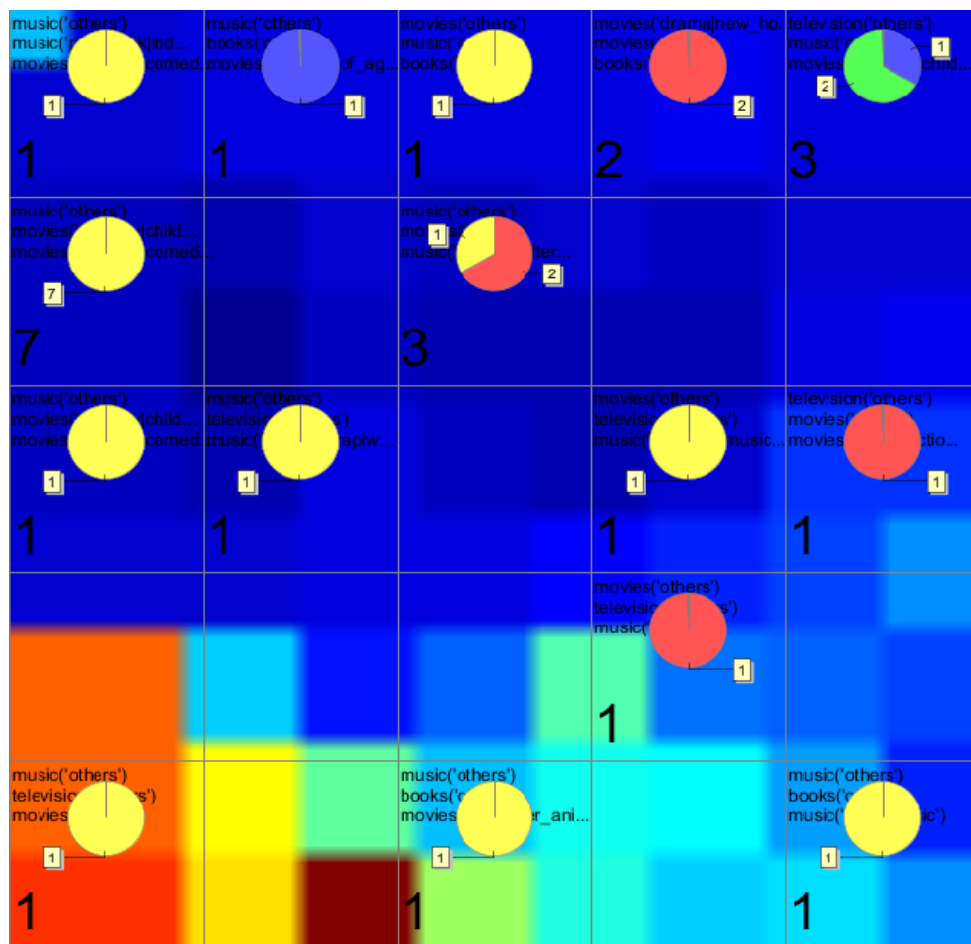




Template Vector File	<pre> \$TYPE template \$XDIM 7 \$YDIM 26 \$VEC_DIM 106 0 music('heavy_metal power_metal') 1 1 0 1 0.5 1 music('gangsta_rap west_coast_hip_hop') 1 1 0 1 0.5 2 music('avant_garde_metal') 1 1 0 1 0.5 3 movies('psychological_thriller comedy thriller') 1 1 0 1 0.5 ... </pre>
----------------------	--

**Figure 4.6:** Input and Template Vector file of SOM training for clustering friends' interest in Facebook.

The SOM Visualization result of this use case is depicted in the below figure. In this SOM map, each category of interest will be classified by each color and displayed in a circle (movies in pink, books in cyan, television in green and music in yellow). The small yellow number and the big black number indicate the number of friends that are interested in the relevant categories and are interested in the same genre of categories, respectively.



**Figure 4.7** SOM Visualization and clustering for friends' interest in Facebook

#### **4.4 Summary**

This chapter has discussed some principal requirements for the trustworthiness of mashup data. It has also conducted an improved WSD method and SOM technique to help users in personal resources clustering, self-monitoring data and avoid unwanted information in SNSs. Next chapter proposes the formulation for mashup-related concepts, a lightweight mashup language, and a semantic-based mashup system that allow making mashup personal resources in Semantic Desktops and SNSs.

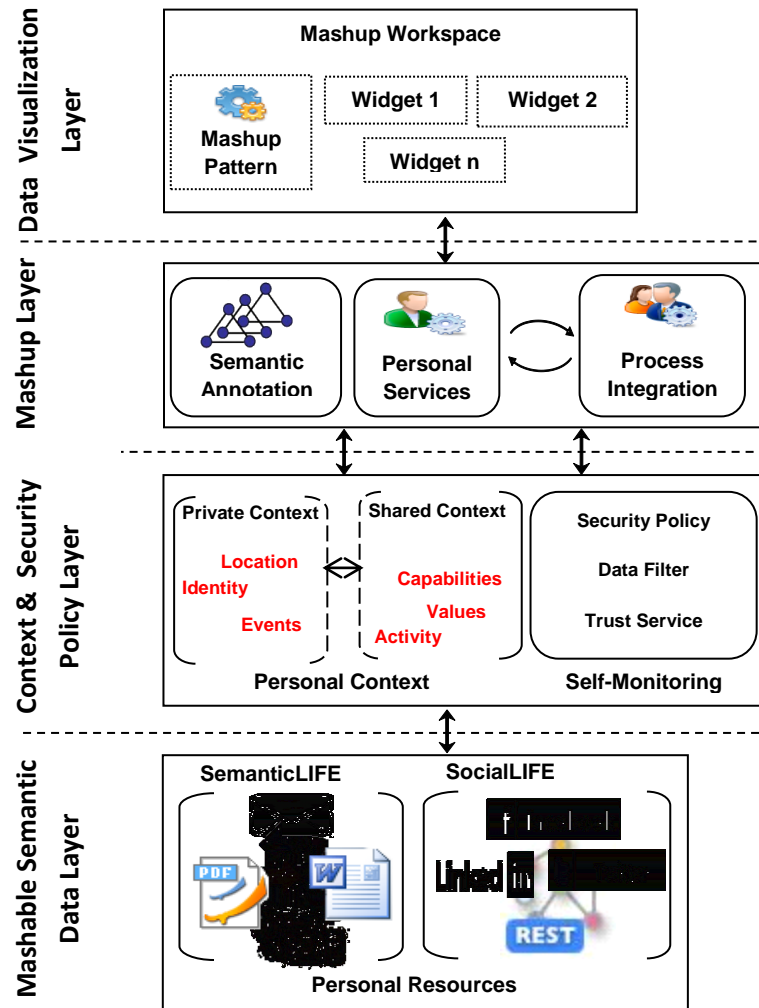
## SEMANTIC-BASED MASHUP

### 5.1 Semantic-based Mashup

Semantic-based mashup is a mashup that can combine services and APIs supported or annotated by a semantic layer [143]. To be more specific, semantic-based mashup applies semantic technologies such as semantic annotation and information extraction to improve the possibilities in choosing and matching the right input items [144]. In this section, the semantic-based mashup architecture is proposed, which aims to facilitate the integration of heterogeneous personal resources sources from SemanticLIFE and SocialLIFE within organizations/enterprises.

This framework consists of four following layers (as illustrated in Figure 5.1):

- *Mashable semantic data*: This layer mainly focuses on the retrieval of personal resources from SemanticLIFE and SocialLIFE in a structured data format, and mapping them to adaptive ontologies, RDF, or Linked Data repositories in order to make a better semantic mashup.
- *Context & Security Policy*: This layer is responsible for the efficient implementation of information security and privacy policies. More importantly, this layer includes context ontologies for describing personal services, as well as applies the privacy policies for self-monitoring the information sharing between SemanticLIFE and SocialLIFE.
- *Mashup Layer*: The mashable semantic data are mashed up in this layer. To create mashup data, the services in SemanticLIFE and SocialLIFE are referred in mashup language that will be introduced in Section [5.2](#).
- *Data visualization*: This layer enhances original contents by adding graphical representation like maps, or images (such as Google Map, Flickr). This layer constitutes the main workspaces that allow end-users to interact with the mashup platform. In addition, the usage of this main workspace is intended as a collaborative usage environment, in which multiple users can share a common widget or a mashup pattern for their common business needs.



**Figure 5.1:** Semantic-based mashup architecture for SemanticLIFE and SocialLIFE

In our framework, three following groups of services are supported:

- *SPARQL-based services*: query the semantic web data via SPARQL endpoints such as DBPedia, Events, etc.
- *Third party services*: consume the third party APIs services that do not expose RDF data, for example Flickr, Google Map, Weather ...
- *Personal services*: query personal resources via custom personal services or via SPARQL endpoints of the data repository of Semantic Desktops.

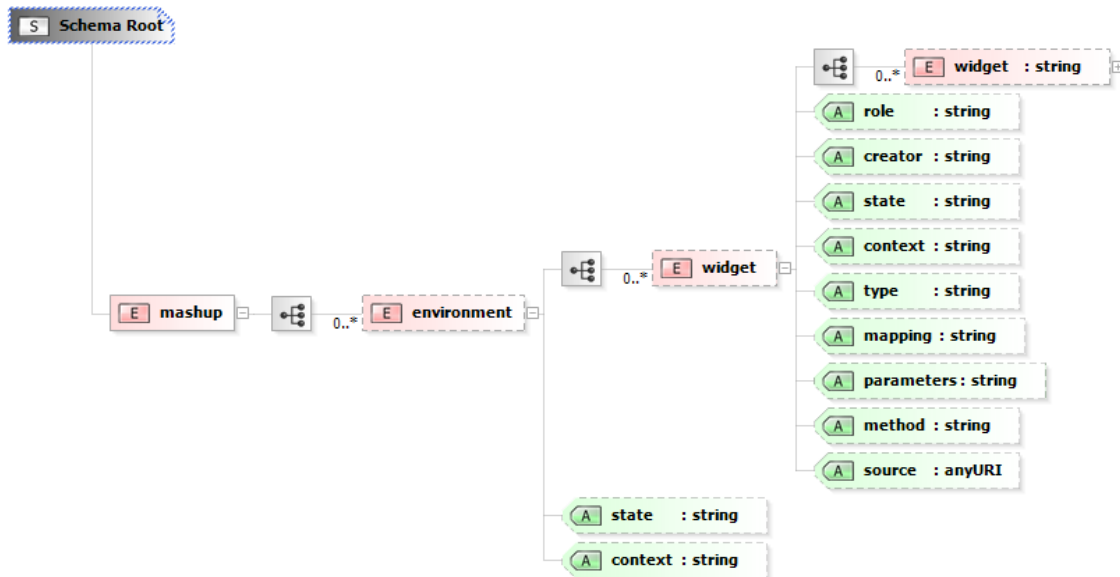
## 5.2 Personal Resources Mashup Language

A lightweight mashup language has been proposed in this research to support our semantic-based mashup system in personal resources mashups. Although a number of mashup languages have been developed (such as MashQL [145], WSML [146], EMMML [89]), each mashup language provides a specific mechanism that enables users building web data mashups. WSML supports end-user working from their browser by embedding scripts into HTML; whereas MashQL is restricted to web data sources represented in RDF, and uses SPARQL as the query language. EMMML is a rather expressive language, but certainly quite complicated and not easy for the users to apply. Besides, EMMML has not supported ontology or customized ontology mapping yet, it only supports constructing a SPARQL query and executing it.

In this framework, the proposed personal resources mashup language (PRML) aims to:

- Create a simple mashup language, which helps developers create a widget for mashup data in a semantic way.
- Develop new composite application in mashup process to fit personal needs based on workflow of connectable widgets.
- Make easily new widget that inherits the predefined widget.

The PRML has four main parts, namely mashup, environment, widget, and parameter as the following schema:



**Figure 5.2:** The schema of personal resources mashup language (PRML)

The root element of PRML schema is *<mashup>* element. The sub element of *<mashup>* contains *<environment>* element that helps user know exactly which environment context they are working with.

- Attribute *context* could be '*SemanticLIFE*', '*SocialLIFE*' or '*Organizations/Enterprises*' or even more.
- Attribute *state* could be '*checked*' or '*unchecked*', meaning to be whether shown or not in the mashup workspace.

The third level contains required *<widgets>* and additional attributes:

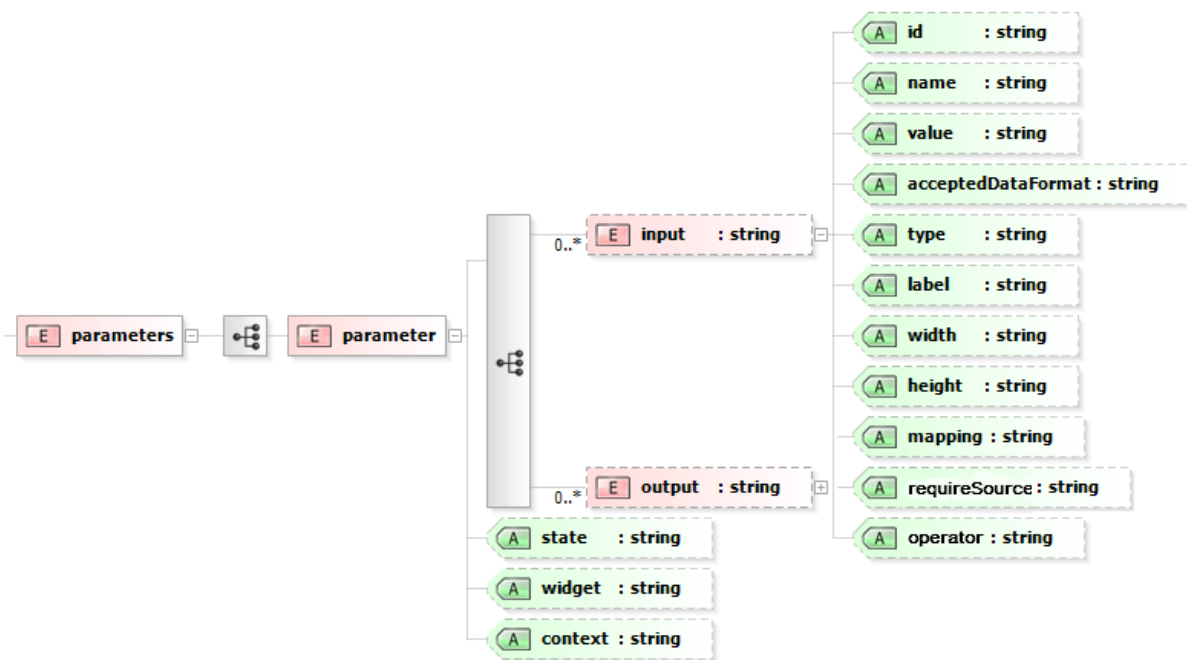
- Attribute '*state*' is set '*checked*' or '*unchecked*', meaning the widget is chosen for mashup or not.
- Attribute '*context*' and '*type*' are the name and type of personal context respectively that users require.
- Attribute '*service*' identifies the service type of that widget (for example, event, financial service, etc.).
- Attribute '*source*' indicates the service source (for example, SPARQL endpoint, source of third party APIs service, etc.).
- Attribute '*role*' is assigned a privilege to a specific user or a group of users who can use the widget.
- Attribute '*mapping*' supports mapping to ontology resource or properties, for example, *mapping='rdf:type Place'* is mapping the input/output with the Place concept of the target ontology.
- Attribute *parameters* indicates the additional parameters of the widget (input, output).

The fourth level *<parameters>* contains one or more *<input>*, *<output>* elements.

- Attribute '*name*' is the name of the input/output parameter.
- Attribute '*type*' is the type of the input/output parameter (String, Number, Array, Boolean, Date).
- Attribute '*value*' is the value of the input/output parameter.
- Attribute '*mapping*' supports mapping to the ontology resource or properties.

- Attribute '*acceptedData/dataFormat*': is the type of data format that is accepted by the output and input port. The data format can be JSON, XML, Sparql-results+xml or object.
- Attribute '*operator*' identifies an operator to construct constraints (for instance, >, <, or regular expression, etc.)
- Attribute '*requireSource*' indicates that the input requires data from another source or not.
- Attributes '*label*', '*width*', and '*height*': are referred to the additional values of the input/output layout that will be used in UI container

The following figure indicates the overview schema for widget parameters of PRML.



**Figure 5.3:** The schema of widget parameters of PRML.

## 5.3 Semantic Mashup Formulation

For our semantic-based mashup system, some mashup concepts and rules are defined to match with the proposed mashup language for semantic annotation.

### 5.3.1 Definition and Rule

#### Definition 1. Widget

A widget is a tuple of  $\langle I, R, P \rangle$  where  $I = \{i_1, i_2, \dots, i_p\}$  is a set of Input ports,  $R$  is the result set and  $P$  is the process (or web service) running inside the widget to consume the inputs  $I$  and produce the result  $R$ . Each one of these elements has a set of metadata that describes the functions and properties of that element. For instance, the input ports and output results have the following metadata:

- format: accepted data format rule
- mapping: mapping rule attribute
- value: value of input/output
- extras: additional parameters such as layout, id, name

The process may include further metadata such as:

- additional parameters for the widget, such as id, name, width, height...
- process description
- type of process (e.g., SPARQL, Javascript based, Web Service, etc.)

In order to use a widget, they should be instantiated and called by the context engine. One such context could be a Mashup environment.

#### Definition 2. Mashup

A mashup  $M$  is a set of 4-tuple,  $M = \{ \langle w_i, c_{ij}, w_j, l_{w_j} \rangle \mid \forall i, j = 0 \dots n, i \neq j \}$  where

- $w_i, w_j \in W$ : instance of widgets ( $W$  is the set of available widgets)
- $c_{ij}$ : is the connector between two widgets that denotes the dataflow between output port of widget  $w_i$  and one of the input ports  $l_{w_j}$  of widget  $w_j$ .

In order to help users to design mashups in a semantic way, some rules for the mashup process are defined as follows.



Let  $D = \{d_1, d_2, \dots, d_q\}$  be a set of data formats, and  $O = \{o_1, o_2, \dots, o_r\}$  be a set of ontology types.

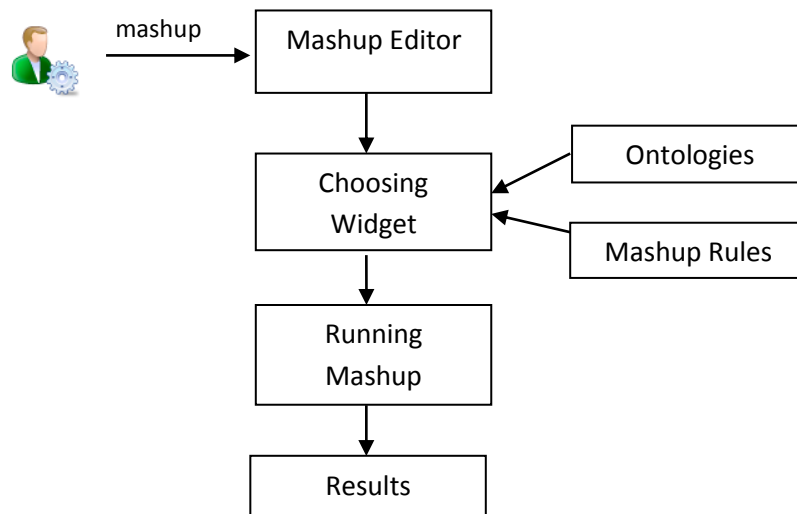
**Rule 1: Feasible Connection**

A feasible connection between two widgets  $w_i$  and  $w_j$  is a connection  $c_{ij}$  that input data format ( $d_i$ ) or mapping ontology type ( $o_i$ ) of the output port on widget  $w_i$  side is compatible with  $d_j$  or  $o_j$  on the input port at widget  $w_j$  side.

**Rule 2: Avoiding loops**

If the Mashup contains a 4-tuple  $\langle w_i, c_{ij}, w_j, l_{w_j} \rangle$  then there should be no a 4-tuple  $\langle w_j, c_{ji}, w_i, l_{w_i} \rangle$ .

The process for applying these rules in mashup is depicted in the figure below.



**Figure 5.4:** The process for applying ontologies and mashup rules in the mashup.

### 5.3.2 Realization of Definitions and Rules

In this section, the realization of the definitions and rules mentioned above is described in PRML syntax and explained in more details.

For each widget  $w_i$ , input/output parameters can accept a data format  $d_i \in D$ . This rule is described via property *acceptedDataFormat*=' $d_i$ ' of input/output parameters. The following example depicts an input that only accepts data 'String'.

```
<input acceptedDataFormat='String' name='event_name' />
```

For each widget  $w_j$ , input/output parameters can be mapped to an ontology type  $o_i \in O$ . This rule is described via property *mapping*="o<sub>i</sub>". The following sample input is mapped to a financial ontology related to posted date (DTPOSTED) of bank statement.

```
<input name="start_date" type='date' mapping='ofx:DTPOSTED' />
```

For forming a mashup  $M = \langle w_i, c_{ij}, w_j, lw_j \rangle$ , this sample describes the widgets, their inputs and connection as follows:

Widget  $w_i$  'Bank Statement' has three inputs and one output port, the data source of this widget is retrieved via the SPARQL endpoint of bank statement repository. This widget is defined in the following PRML syntax:

```
<widget role='user' context='Bank Statement' mapping='slife: BankStatement'
source='http://localhost:8080/ repositories/bank_statement'>
  <parameters>
    <parameter>
      <input name="account" type='number' mapping='ofx:ACCTID'
        label='Bank Nr.' />
      <input acceptedDataFormat='Date' name="start_date" type='date'
        mapping='ofx:DTPOSTED' label='From Date' />
      <input acceptedDataFormat='Date' name="end_date" type='date'
        mapping='ofx:DTPOSTED' label='To Date' />
      <output dataFormat='sparql-results+xml' name='bank_statements'
        mapping='ofx:STMTTRN' />
    </parameter>
  </parameters>
</widget>
```

Widget  $w_i$  connects with widget  $w_j$  'Calendar', which shows the bank statement in a calendar viewer, is defined in the following syntax:

```
<widget context='Calendar' parameters='Calendar.paras' service='CalendarViewer'>
  <parameter>
    <input name='calendars' requiredSource='true'
      acceptedDataFormat='sparql-results+xml' />
  </parameter>
</widget>
```

The connection  $c_{ij}$  is established via output port named 'bank\_statements' of  $w_i$  and input port named 'calendar' of  $w_j$ .

### 5.3.3 Widget-based Query Generation

In order to query remote RDF resources, the query is constructed based on widget parameters as follows:

Suppose  $P = \{p_1, p_2, \dots, p_n\}$  is a set of input parameters of the widget and  $R$  is the output result of the widget. Each input parameter  $p_i$  ( $i=0 \dots n$ ) has its optional properties such as name, mapping, type, value and operator in turn.

A query Q can be defined as:  $Q = \langle S, W, F \rangle$  where

- $S = \langle \text{SELECT}, R, P \rangle$ : SELECT statement with relevant parameters
- $W = \langle \text{WHERE}, R, P \rangle$ : WHERE clause with relevant parameters
- $F = \langle \text{FILTER}, P \rangle$ : FILTER constrains with relevant required parameters

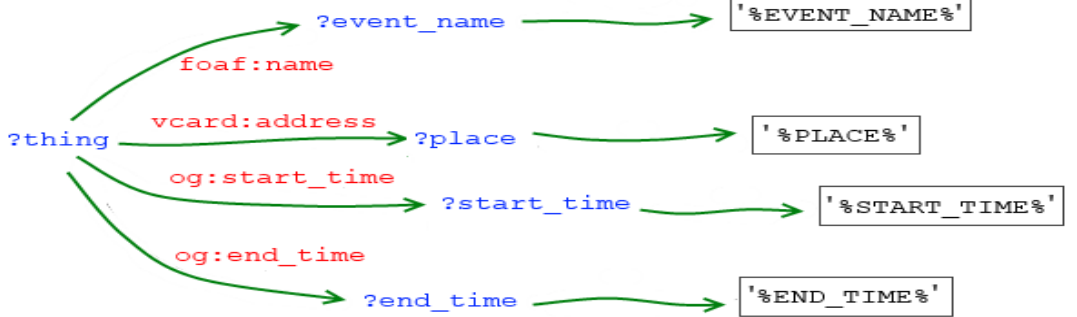
Query Q is generated by the following principles:

- Enumerating parameters' names for the SELECT part: the SPARQL variables of SELECT statement are formed by output R and relevant parameters' names. The SELECT statement must be "SELECT ?R ?p<sub>1</sub>[name] ... ?p<sub>n</sub>[name]" .
- Taking output R, names, and mappings of parameters to form the triple patterns for WHERE clause. The WHERE clause must be "WHERE {?R ?p<sub>1</sub>[mapping] ?p<sub>1</sub>[name] . ... . ?R ?p<sub>n</sub>[mapping] ?p<sub>n</sub>[name] }"
- Taking names, values, and operators of parameters to add filter expressions for the FILTER constrains: depending on relevant parameters' operators, the syntax of the FILTER will be alternative. For example, if the operator of p<sub>i</sub> is 'regex', the string matching syntax must be "FILTER (regex(?p<sub>i</sub>[name], ?p<sub>i</sub>[value]))"; otherwise, the syntax must "FILTER (?p<sub>i</sub>[name] ?p<sub>i</sub>[operator] ?p<sub>i</sub>[value])".

Widget Parameters	<pre> &lt;parameter&gt;   &lt;input name='p1[name]' mapping='p1[mapping]' type='p1[type]'     value='p1[value]' operator='p1[operator]' /&gt;   &lt;input name='p2[name]' mapping='p2[mapping]' type='p2[type]'     value='p2[value]' operator='p2[operator]' /&gt;   ...   &lt;input name='pn[name]' mapping='pn[mapping]' type='pn[type]'     value='pn[value]' operator='pn[operator]' /&gt;   &lt;output name='R' /&gt; &lt;/parameter&gt; </pre>
Parsing parameters into SPARQL	<pre> With p<sub>i</sub> ∈ P, i=0...n SELECT ?R ?p1[name] ?p2[name]... ?pn[name] WHERE {?R ?p1[mapping] ?p1[name] .       ?R ?p2[mapping] ?p2[name] .       ...       ?R ?pn[mapping] ?pn[name] .       &lt;!-- if p<sub>i</sub>[operator]= 'regex'--&gt;       FILTER (regex(?p<sub>i</sub>[name], ?p<sub>i</sub>[value]))       &lt;!-- else --&gt;       FILTER (?p<sub>i</sub>[name] ?p<sub>i</sub>[operator] ?p<sub>i</sub>[value]) } </pre>

**Figure 5.5:** Query generation based on widget parameters

The following example depicts the process of generating a SPARQL query from widget parameters for an event service.

Widget parameters	<pre> &lt;parameter&gt;   &lt;input name='event_name' mapping='foaf:name' value='%EVENT_NAME%' operator='regex' /&gt;   &lt;input name='place' mapping='vcard:address' value='%PLACE%' operator='=' /&gt;   &lt;input name='start_time' value='%START_TIME%' mapping='og:start_time' operator='&gt;=' /&gt;   &lt;input name='end_time' dataFormat='Date' operator='&lt;=' mapping='og:end_time' value='%END_TIME%' /&gt;   &lt;output name='thing' /&gt; &lt;/parameter&gt; </pre>
Parsing parameters into SPARQL	 <pre> SELECT ?thing ?event_name ?place ?start_time ?end_time WHERE {   ?thing foaf:name ?event_name .   ?thing vcard:address ?place .   ?thing og:start_time ?start_time .   ?thing og:end_time ?end_time   . FILTER ( regex(?event_name,'%EVENT_NAME%'))   . FILTER ( ?place ='%PLACE%' )   . FILTER ( ?start_time &gt;='%START_TIME%' )   . FILTER ( ?end_time  &gt;='%END_TIME%' ) } ORDER BY ?thing </pre>

**Figure 5.6:** An example of query generation based on widget parameters

### 5.3.4 Mashup Algorithm

Suppose that a set of widgets  $\{w_i \dots w_j\}$  is used in a mashup composition  $M$ . If  $w_j$  is requested to execute, then it is necessary to check whether  $w_j$  requires output from another widget  $w_i$  or not. If  $w_j$  has no required input, then  $w_j$  is able to execute. Otherwise,  $w_i$  (the widget that provides the required input) needs to be called and so on. For this reason, a mashup algorithm for possible loop detection is required.

Algorithm for mashup in our case is based on the acyclic directed graphs, in which each widget of mashup is considered as a vertex of a graph, and each connection between two widgets is considered as an edge of a graph. For iterating through all widgets of a mashup (nodes of a graph), it is considered to apply the very basic algorithm for graph that is Depth first-search (DFS) algorithm to ‘visit’ the widgets. In case of mashup, the output value of any

selected widget will be the input value of the connected widgets via the relevant port. The mashup M and the visited widget will be handled by the main recursive function *processWidget(M,widget)*. Each visited widget will perform the corresponding process inside by the function *executeProcess()* and return the value via the output port. This return value could be the input value of the connected widget (if available).

Instead of checking all edges at once in DFS algorithm, the edges are checked in two phases by differentiating discovery edges and back edges via input ports or output port respectively. Discovery edges are those edges connecting a vertex to another descendant one, and back edges are those edges connecting a vertex to another ancestor one.

The steps of mashup algorithm can be described in the following recursive pseudo code for processing a mashup:

---

**Algorithm** processWidget(M,widget)

---

**Input:** mashup M, an instance widget  $\in$  M

```

1: if widget.inputPorts > 0 then
2:   for each inputPort  $\in$  widget
3:     //get connected widget
4:     previousWidget = inputPort.connectedWidget
5:     if previousWidget is executed then
6:       //get value of connected widget
7:       widget.inputPort.value = previousWidget.outputPort.value
8:     else
9:       //recursive-process connected widget
10:      widget.inputPort.isVisited = true
11:      processWidget(M,previousWidget)
12:   end for
13: end if
14: widget.isExecuted = true;
15: //executeProcess function: execute process inside widget (SPARQL, service, etc)
16: widgetResult = executeProcess()
17: for each outputPort  $\in$  widget
18:   nextWidget = outputPort.connectedWidget
19:   if nextWidget.inputPort is visited then
20:     //get value of current widget
21:     nextWidget.inputPort.value = widgetResult
22:   else
23:     //recursive-process connected widget
24:     nextWidget.inputPort.isVisited = true
25:     processWidget(M,nextWidget)
26:   end if
27: end for
28: return widgetResult

```

---

**Algorithm 5.1:** Mashup algorithm

## 5.4 Semantic Mashup Patterns

In this research, semantic mashup pattern is a set of services/widgets, including SPARQL services, semantic personal services or other external web services that can be executed to make data mashups by the proposed mashup framework. Semantic mashup patterns can be seen as the predefined patterns that are constructed from SemanticLIFE and SocialLIFE for using and sharing of widgets.

It is possible to use mashup patterns to explore the accumulation of personal resources for following practical purposes:

- *Personalization*: mashup widgets are personalized based on personal context. With some preferences or sensitive contents, widgets are private or shareable. For example, users want to share their interests but keep their banking statements in private.
- *Real-time monitoring*: allows users/organizations to observe the interested real-time data from mashups via SNSs or from other users' shared widgets.
- *Reuse & Collaboration*: widgets are created and saved in a repository for later use or for collaborating in workflows.

The following mashup pattern, which is defined in PRML, depicts a sample pattern to retrieve personal events from SemanticLIFE and pictures from Flickr, which are annotated with the relevant places of personal events.

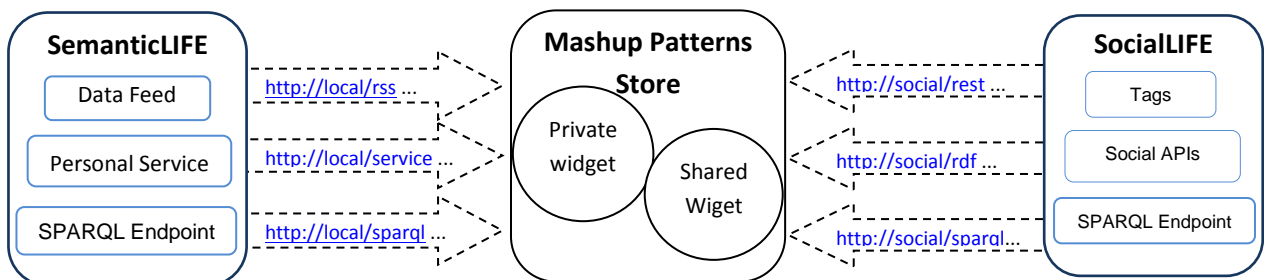
```

< mashup>
  < environment context='SemanticLIFE'>
    < widget role='user' creator='user' context='Calendar Events'
      service='CalendarEvents' type="calendar_event"
mapping='slife:Calendar'
      source='http://localhost:8080/repository/'>
    < parameters>
      < parameter>
        < input acceptedDataFormat='String' type='string'
          name='event_name' mapping='foaf:name' operator='regex' />
        < input acceptedDataFormat='String' type='string'
          name='place' mapping='vcard:address' operator='regex' />
        < input acceptedDataFormat='Date' type='date'
          name='start_time' mapping='event:start_time' operator='>=' />
        < input acceptedDataFormat='Date' type='date'
          name='end_time' mapping='event:end_time' operator='<=' />
        < output name='events' mapping='rdf:type Events'
          dataFormat='sparql-results+xml' />
      < /parameter>
    < /parameters>
  < /widget>
< /environment>
< environment context='SocialLIFE'>
  < widget role='user' type='flickr'
source='http://api.flickr.com/services/rest/?method=flickr.photos.search'>
    < parameters>
      < parameter>
        < input name='tags' type='string'
          requiredSource='true' acceptedDataFormat='entry'>
        < input name='api_key' type='string' value='%FLICKR_API_KEY%' />
        < input name='per_page' type='string' value='5' />
      < /parameter>
    < /parameters>
  < /widget>
< /environment>
< / mashup>

```

**Figure 5.7:** An example of mashup pattern to retrieve personal resources.

The semantic mashup patterns can be simply reused and archived in a store of mashup patterns that depicts in the following figure:



**Figure 5.8:** Mashup patterns store of personal resources

## **5.5 Summary**

This chapter has proposed some formulations for mashup-related concepts such as semantic mashup, widget, and mashup rules. In addition, this chapter has also proposed a lightweight mashup language and a semantic-based mashup system that support end-users to design semantic-aware mashup dataflow. The next chapter will present the implementation prototype of our mashup system, some significant use cases, and evaluation results.



## IMPLEMENTATION RESULTS AND EVALUATION

### 6.1 Implementation Results

As a proof of concept and evaluation of the proposed approach, a prototype has been developed based on Adobe Flex [147]. Adobe Flex is a free, open source application framework for the development and deployment of cross-platform applications on all major browsers, desktops, and devices.

For preparing data for mashup use cases, some financial data are converted from OFX format into RDF triple and stored in the SemanticLIFE repository (as described in Section 3.3.1). Besides, some other SocialLIFE information resources such as Twitter tweets, Facebook interests, Flickr images, etc. will be used.

#### 6.1.1 Personal Resources Retrieval from SocialLIFE

Some user-generated contents in SNSs are available to download or access in structured data, feeds, or other data formats that are open to everyone such as Open Graph of Facebook. These structured data are the goldmine of potential mashup data that can be mashed up and used by different applications. However, there are also some SNS sources that are either unstructured or lack of the required information. Some examples of such data resources are users' tweets in Twitter or users' interests in Facebook. Even if developers are able to capture those data, they are not readily available for analysis or reuse.

In order to retrieve some unstructured personal resources from SNSs and preparing data for mashup, a data retrieval component is implemented for extracting data from major SNSs platforms such as Facebook, Flickr, Youtube, Twitter, and MindMeister [148]. The unstructured data are transformed into structured data, which are leveraged in various ways (i.e., mashup in our study). The retrieved data are also stored locally and used as the buffered feeds to improve the processing performance.

For personal resources retrieval from SocialLIFE, personal resources from Facebook have been retrieved and integrated with Freebase to annotate resources with the relevant metadata in the following use case.

### **Collecting and aggregating social data from Facebook & Freebase**

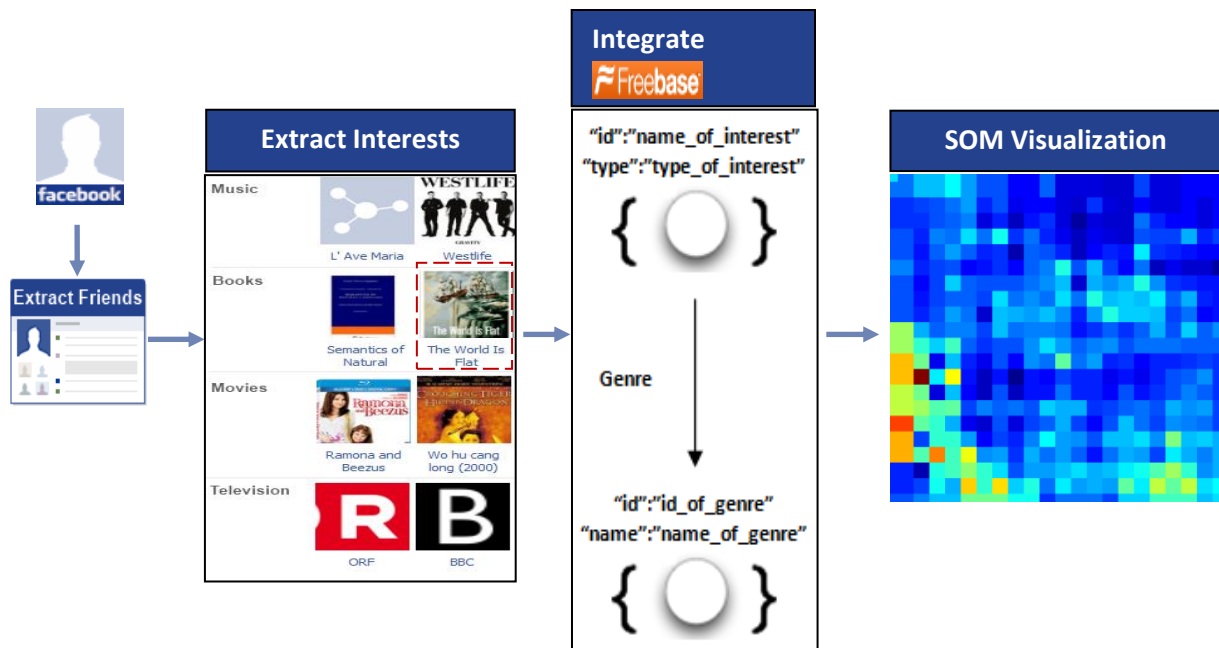
Facebook is a social networking service and has more than 500 million monthly active users, 900 million objects and 30 billion pieces of contents that people interact with [149]. It provides a social channel to enable users to create their personal Home Page (like profiles, photos, events, etc.), and create their social connection (such as their friendships, joining an interested group or community). It also supports a development platform that enables developers to interact with Facebook data through its graph API and open graph protocol [150]. Because of Facebook's popularity, other enterprises and developers may include Facebook's Social Plugins into their websites or applications to make their pages more social. The simplest example is the "Like Button" that enables users to like any links, movies, books, etc. These characteristics make Facebook become a large online personal data repository.

In Facebook, many users may lose the overview of what their friends are doing and this may lead to incorrect judgments about them when the interest of a friend is out-of-favor. For this purpose, the user's network of friends is scanned and all interests will be extracted. In the next step, the interest items are automatically annotated with relevant information such as book category, film genre, etc. In this case, Facebook users' profiles and friends' interests are considered as the main entry and the main data for integration respectively. Depending on the friends' security and privacy settings, the integration application can access the personal information, including name, gender, interest (music, books, movies, etc.), and some other information. The solution for collecting and aggregating social data from Facebook & Freebase can be summarized as follows:

- After Facebook's users are logged in, the user profile and his/her friend list will be extracted by Facebook Java API [150].
- Interest types (music, books, movies, television) of each Facebook friend will be retrieved and integrated with Freebase for their annotation in turn. The integration solution with Freebase will be described in next section.

In addition, this solution is also a preparing step for another mashup solution that consumes the Facebook friends' interest data to create a Self-Organising Map (SOM) for self-monitoring in social networks. This approach was described with further details in [Chapter 4](#). In the later mashup solution, the output result will be delivered to the SOM component to visualize and give the users an overview of his/her social network context.

The following figure depicts the steps for personal resources retrieval from social data in Facebook & Freebase.



**Figure 6.1:** An example of personal resources retrieval from social data in Facebook & Freebase

### Integrating Freebase for retrieval of data annotation

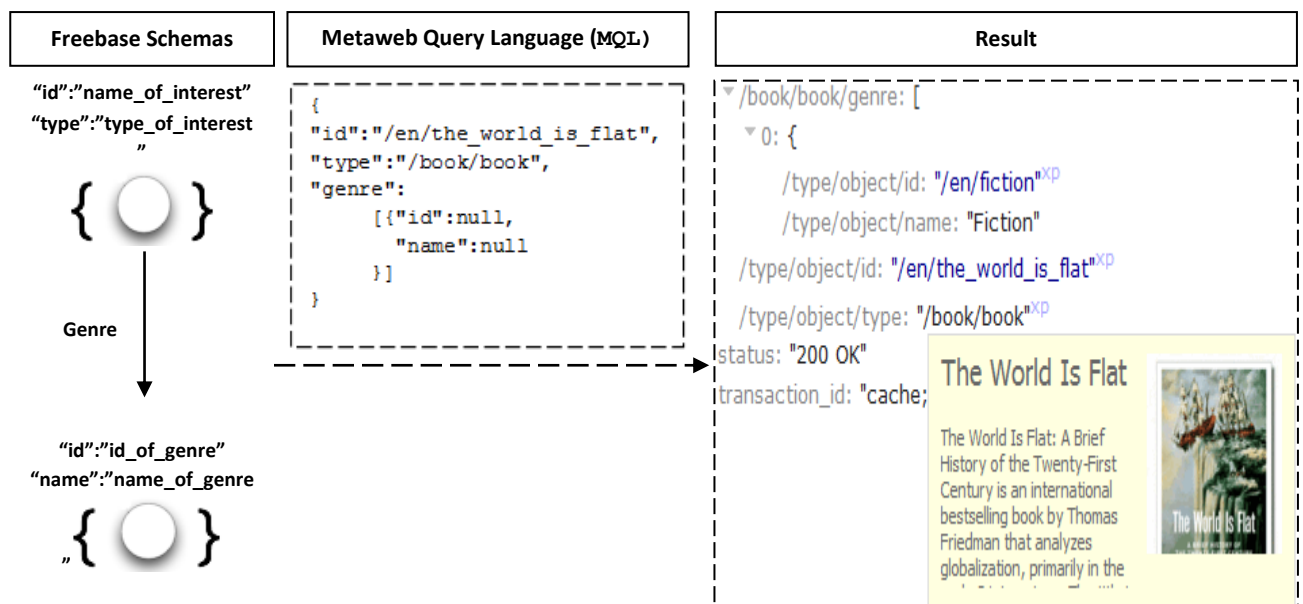
Freebase was developed to be a large public structured knowledge base, which supports human and machine-readable data in the semantic way. In Freebase, data are structured in schema and expressed through domains, types, and properties. Freebase contains more than ten million topics of people, places, and things with thousands of types [151]. Each topic is linked to other related topics and annotated with important properties like movie genres, book subjects, etc. With Freebase, users can query and disambiguate entities in varied ways by searching for IDs, properties, or text. Every retrieved entity is available in JSON or RDF, which makes it easy to analyze.

Our solution for retrieval of personal data annotation can be summarized as follows:

- Each user interest entry in Facebook will be classified in an appropriate type of interest, then it will be searched in Freebase based on its schemas, which are expressed via types and properties [152]. For example, the “book” is expressed in type “book/book” and has some properties such as id, name.

- The user interest entry will be queried to Freebase by using Freebase API via Metaweb Query Language (MQL) [153]. MQL allows developers to incorporate their applications with data from Freebase database. For example, the following query  
[https://api.freebase.com/api/service/mqlread?query={\"query\":{\"type\":\"type\\_of\\_interest\",\"name\\_of\\_interest\":\"The World is flat\",\"genre\":\[\]}}](https://api.freebase.com/api/service/mqlread?query={\) is supposed to search the genre of a book named “The World is flat”.
- The details of interest entry are extracted from Freebase result.

The following figure depicts how an object (i.e. user interest entry) is annotated and retrieved by querying in Freebase Schema.



**Figure 6.2:** Query Freebase schema for data annotation

### 6.1.2 Mashup Workspace

With this framework, users can design mashups in a mashup workspace based on pre-built widgets. The mashup workspace has an associated layout, which includes some core components such as mashup widget tree, mashup editor, and mashup portal. These components help users to design and execute mashup in a consistent front-end layout as the following figure.

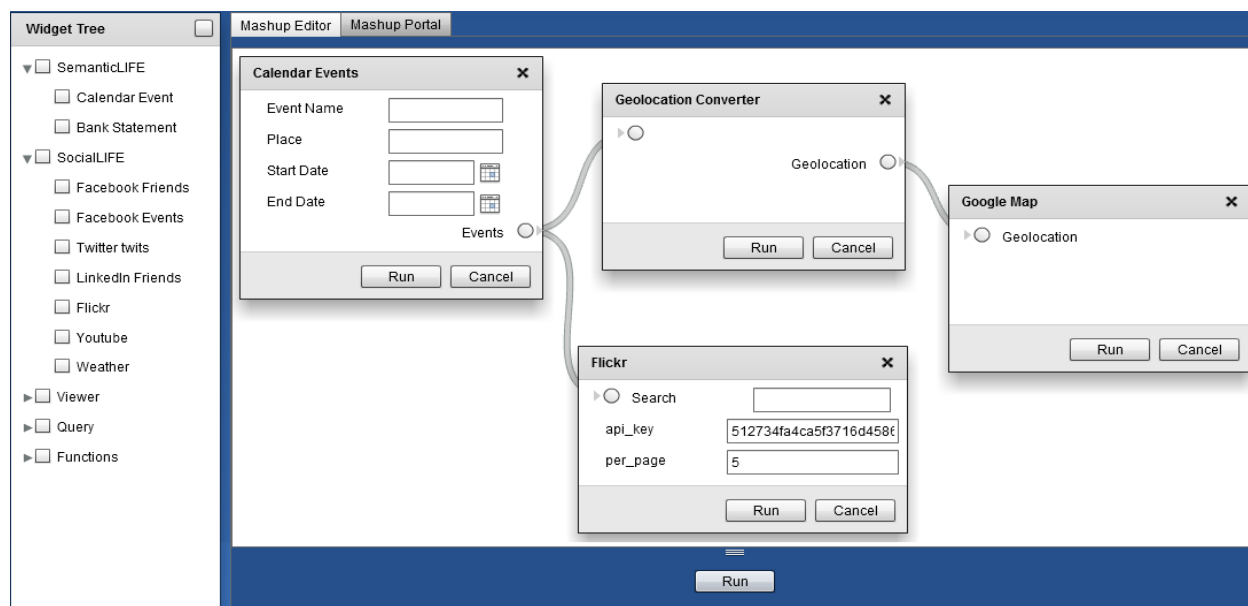


Figure 6.3: Mashup Workspace

### 6.1.3 Widget Tree

To represent widgets for the mashup design, a tree structure is used in a hierarchical view. In this tree structure form, each widget is described in PRML syntax, in which the widget will refer to the corresponding service, ontologies, and parameters. Some widgets will implement the simple or complex logic service, which could be the personal services or the third party social services (such as Google Map, Flickr, etc.). Some widgets may be translated to SPARQL to query RDF data from the SPARQL endpoints. The widgets can be assigned to a special privileged role to a particular user or a group of users.

In addition, end-users can combine with some other widget types:

- *Viewers*: present the mashup data in various formats such as in Geographic Map, Grid, or Diagram.

- *Functions*: process the output data to adapt requirements with other widgets. For instance, in order to show the location of the events' place in map users need to use 'Location converter' to extract the latitude and longitude of the relevant place.

The following figure shows an example of widget tree with relevant configuration in PRML that represent the widgets for personal resources mashups in SemanticLIFE and SocialLIFE.

<pre> &lt;environment context='SemanticLIFE'&gt;   &lt;widget role='user' creator='user'     context='Calendar Event' service='CalendarEvent'     type="calendar_event" mapping='slife:Calendar'     source='http://localhost:8080/repositories/sparql'     parameters='CalendarEvent.paras' /&gt;   &lt;widget role='user' creator='user'     context='Bank Statement' type="bank_statement"     mapping='slife:BankStatement'     source='http://localhost:8080/repositories/sparql'     parameters='BankStatement.paras' /&gt; &lt;/environment&gt;  &lt;environment context='SocialLIFE'&gt;   &lt;widget role='user' context='Facebook Friends'     type='FacebookFriends' service='FacebookFriends'     parameters='FacebookFriends.paras' /&gt;   &lt;widget role='user' context='Facebook Events'     type='FacebookEvents' service='FacebookEvents'     parameters='FacebookEvents.paras' /&gt;   &lt;widget role='user' context='LinkedIn Friends'     type='LinkedInFriends'     parameters='LinkedInFriends.paras' /&gt;   &lt;widget role='user' context='Flickr'     type='flickr'     mapping='http://localhost/sociallife-items.owl#Image'     source='http://api.flickr.com/services/rest/?method=flickr.photos.search'     parameters='flickr.paras' /&gt;   &lt;widget role='user' service='Youtube'     type="youtube" mapping='rdf:type Video'     parameters='Youtube.paras' /&gt;   &lt;widget role='user' context='Weather'     type="weather" mapping='rdf:type Weather'     parameters='Weather.paras' /&gt;   ... &lt;/environment&gt; </pre>	<div style="border: 1px solid gray; padding: 5px;"> <b>Widget Tree</b> <input type="checkbox"/> </div> <ul style="list-style-type: none"> <li>▼ <input type="checkbox"/> SemanticLIFE       <ul style="list-style-type: none"> <li><input type="checkbox"/> Calendar Event</li> <li><input type="checkbox"/> Bank Statement</li> </ul> </li> <li>▼ <input type="checkbox"/> SocialLIFE       <ul style="list-style-type: none"> <li><input type="checkbox"/> Facebook Friends</li> <li><input type="checkbox"/> Facebook Events</li> <li><input type="checkbox"/> LinkedIn Friends</li> <li><input type="checkbox"/> Flickr</li> <li><input type="checkbox"/> Youtube</li> <li><input type="checkbox"/> Weather</li> <li>▶ <input type="checkbox"/> Viewer</li> <li>▶ <input type="checkbox"/> Query</li> <li>▶ <input type="checkbox"/> Functions</li> </ul> </li> </ul>
--	---

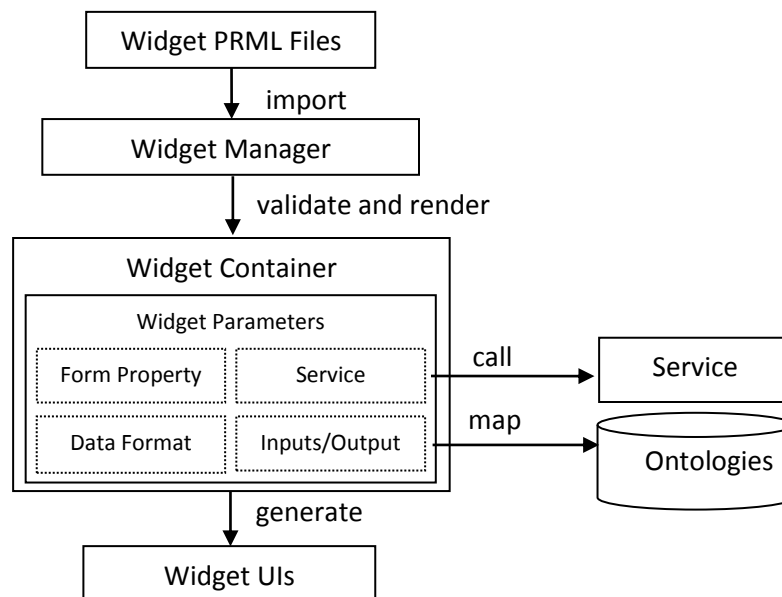
**Figure 6.4:** Widget tree for personal resources mashups in SemanticLIFE and SocialLIFE.

#### 6.1.4 Widget UI

Widget UI is an instance for a specific widget in the widget tree. In the proposed mashup system, Widget UI is both a graphical user interface and a software component with a specific function. Widget UI is defined in PRML syntax that can be rendered and described as a user interface with functional form elements. In this section, a convenient and flexible Widget UI generation mechanism is provided for parsing the predefined widget from PRML files into a built-in form.

##### Widget UI generation mechanism

The mechanism for Widget UI generation is described as in the following figure and steps:



**Figure 6.5:** Widget UI generation mechanism

- First, the user interface configuration and functionality of the widgets are defined in PRML files.
- Then the widget PRML files are imported into the widget manager.
- The widget container will validate and render those PRML files:
  - o Widget parameters are in turn rendered into form properties, services, data format, and input/output.
  - o Service parameter will be called by the relevant service.
  - o Input/output parameters will be mapped with the appropriate ontologies.
- Finally, Widget UIs are generated if the validation is successful.

Service and user interface of a Widget UI are rendered by a widget container with the following parameters:

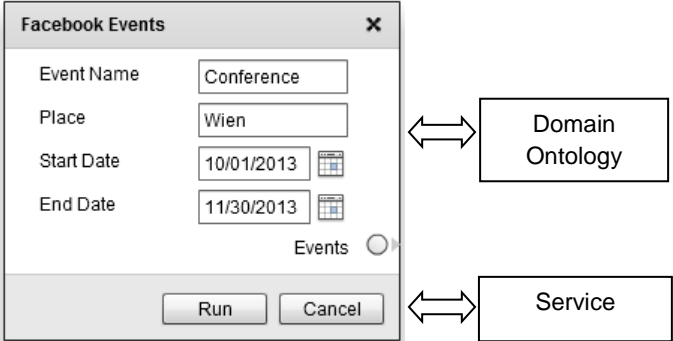
- *Form properties types*: support String, Number, Array, Boolean, Date.
- *Mapping type*: support mapping to ontology resource or properties. For example, in order to map a location with a place ontology, the following syntax is used: *mapping='rdf:type Place'* if the place ontology of RDF Schema, or *mapping='dbpedia-owl: Place'* if the place ontology of DBpedia.
- *Data Format*: is the type of data format that input or output port accepts. The data format can be JSON, XML, or object.
- *Service*: is referred to a specific viewer (e.g. Calendar Viewer, Flickr Viewer, etc.), data operation (e.g. Location converter, etc.), or service (e.g. personal service, third party service, or SPARQL-based service).

The following example describes the generation steps of Widget UI with the appropriate ontologies and service from a PRML file:

- Widget's parameters are defined in PRML file: for example, FacebookEvents requires four input parameters, namely '*event\_name*', '*place*', '*start\_time*' and '*end\_time*'; and one output parameter namely '*events*'.
- Widget parameters are rendered into form elements that are mapped with appropriate domain ontologies and service. Each input parameter is represented as a form element such as textbox, combo box or date field. For example, the FacebookEvents widget is executed by the built-in service '*FacebookEvents*', input parameter '*place*' represents a textbox and is mapped with the ontology '*vcards:address*', input parameter '*start\_date*' represent a date field, and is mapped with the ontology '*og:start\_time*', etc.
- Widget UI is then parsed into a SPARQL query with relevant variables according to the widget-based query generation in section [5.3.3](#).



The following figure depicts more details for the example mentioned above:

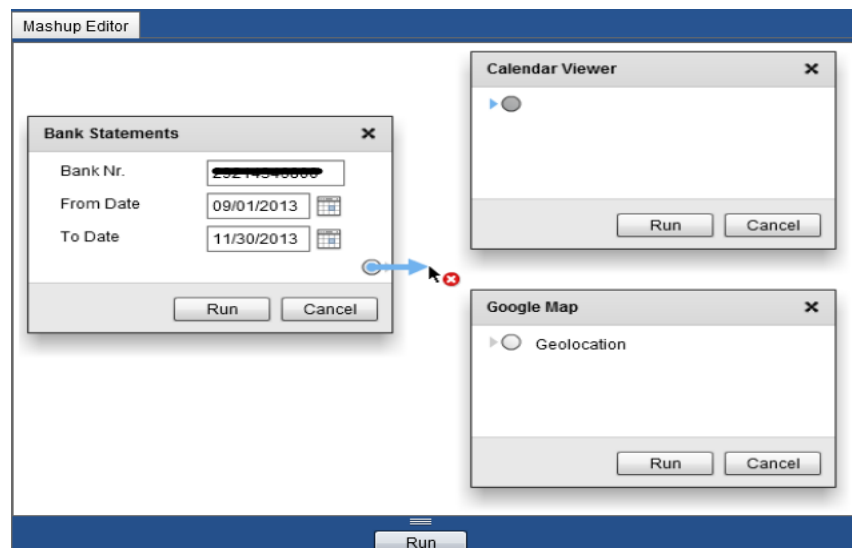
Defining Widget Parameters	<pre> &lt;parameter widget='FacebookEvents'&gt;   &lt;input acceptedDataFormat='String' type='string' name='event_name'     mapping='foaf:name' label='Event Name' operator='regex'     value='%EVENT_NAME%' /&gt;   &lt;input acceptedDataFormat='String' type='string' name='place'     mapping='vcard:address' label='Place' operator='regex'     value='%PLACE%' /&gt;   &lt;input acceptedDataFormat='Date' type='date' name='start_time'     mapping='og:start_time' label='Start Date' operator='&gt;='     value='%START_TIME%' /&gt;   &lt;input acceptedDataFormat='Date' type='date' name='end_time'     mapping='og:end_time' label='End Date' operator='&lt;='     value='%END_TIME%' /&gt;   &lt;output mapping='rdf:type Events' label='Events' name='events' /&gt; &lt;/parameter&gt; </pre>
Mapping Ontology and Service with Widget UI	
Mapping parameters from Widget UI into SPARQL	<pre> SELECT ?thing ?event_name ?place ?start_time ?end_time WHERE {?thing foaf:name ?event_name . ?thing vcard:address ?place . ?thing og:start_time ?start_time . ?thing og:end_time ?end_time . FILTER ( regex(?event_name,'Conference') ) . FILTER ( regex(?place,'Wien') ) . FILTER ( ?start_time &gt;='2013-10-01') . FILTER ( ?end_time &lt;='2013-11-30') } ORDER BY ?thing </pre>

**Figure 6.6:** Parsing FacebookEvents widget' parameters into Widget UI and relevant SPARQL query.

### 6.1.5 Mashup Editor

The mashup editor is used to design the visual mashup by dragging and dropping widgets from the widget tree into the mashup editor. The drag-and-drop widgets will be rendered in corresponding widget UIs that represent the data/service operations. Widgets can be connected via feasible connections according to the first rule in section [5.3.1](#). For each feasible connection, when users choose an output port to find the connectable input port, the mashup editor will highlight the input ports of other widgets that have the same matching data or ontology type.

For instance in the following figure, the output of 'Bank statements' widget has the feasible connection with the input of 'Calendar Viewer' widget, which is highlighted but the input port of 'Google Map'.



**Figure 6.7:** Mashup editor with highlighted feasible connection

### 6.1.6 Mashup Portal

Mashup portal is a workplace that displays widget UIs and delivers mashup data to users. Mashup portal is built by composing separate widget UIs in the mashup design phase of mashup editor. Each widget UI provides the mashed up data from diverse sources or services. After executing mashed up services, their results are rendered and shown as appropriate widget UIs in a uniform way where users can access and preview their mashup data. Mashup portal also allows users to customize or navigate their view of information. The following figure illustrates how the mashup portal is displayed.

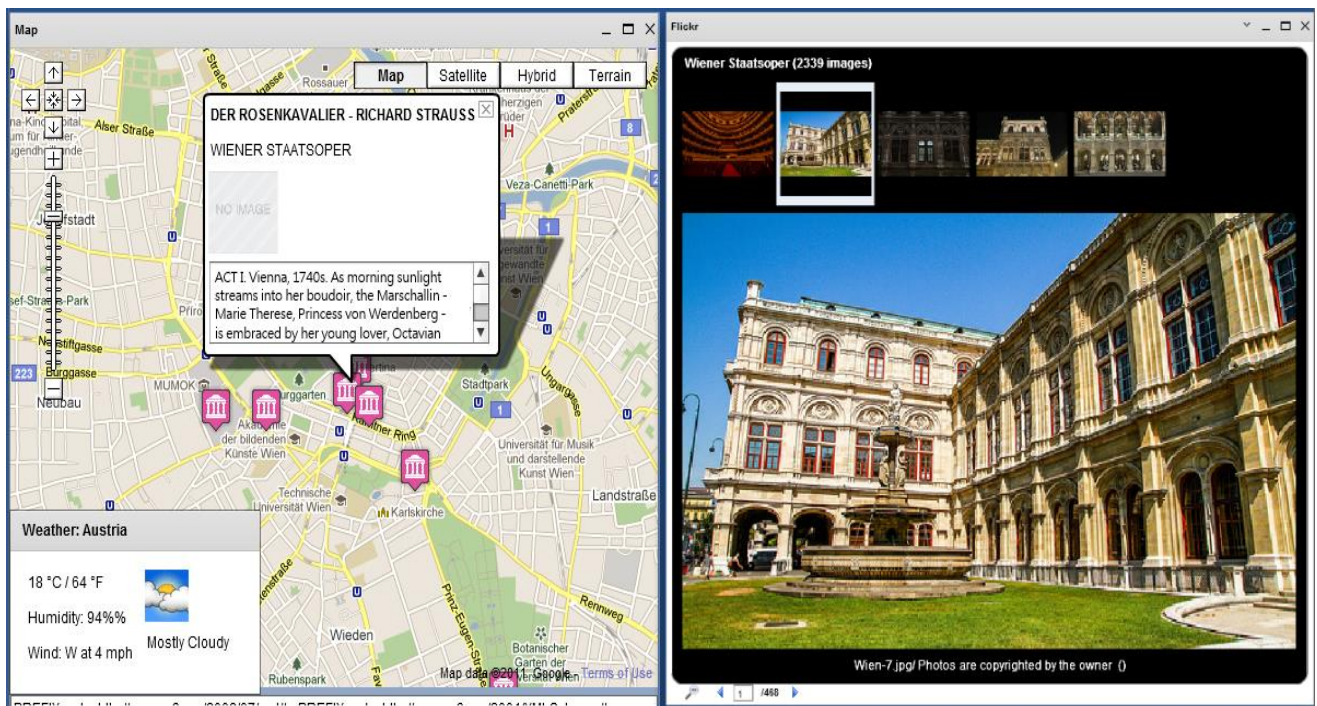
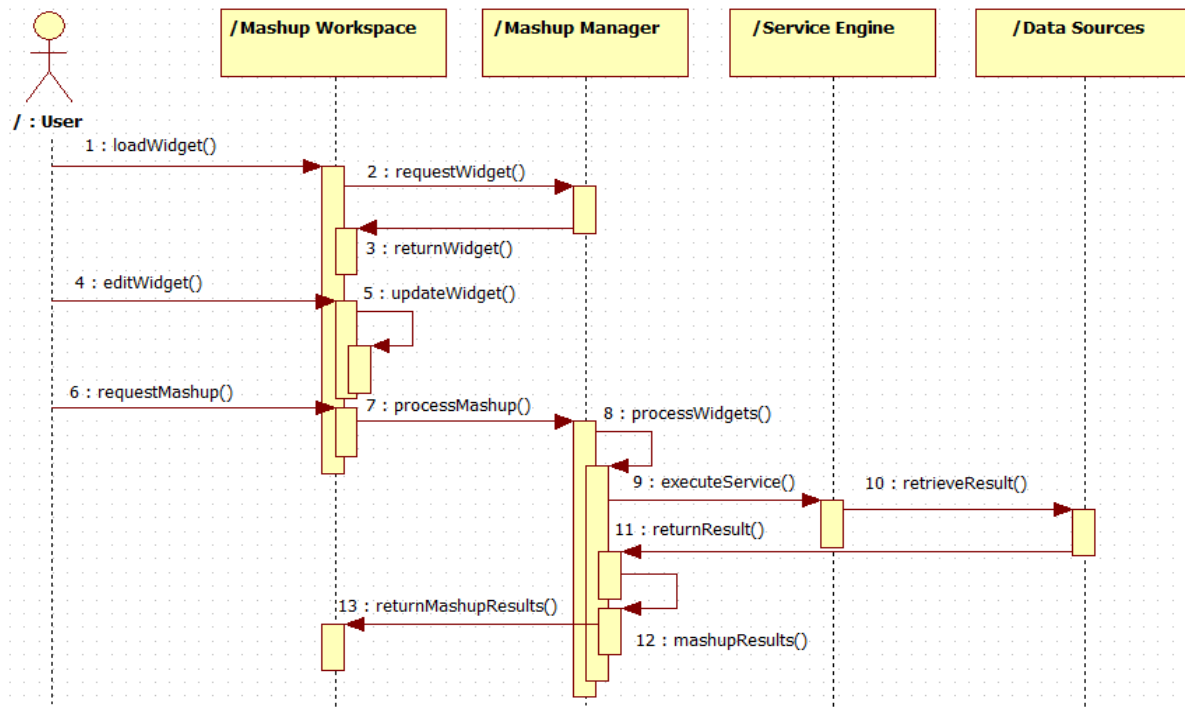


Figure 6.8: Mashup Portal with some sample widget UIs

### 6.1.7 Mashup Sequence Diagram

The following figure shows the UML sequence diagram that summarizes the sequence activities for designing and running data mashups in our framework.



**Figure 6.9:** Sequence diagram of designing and running mashup

In the above sequence diagram, after receiving the loaded widget message, the mashup workspace will request the widgets from the widget manager. The relevant widget is then returned to users for editing. Once users request mashup to run, the mashup workspace will send the processing mashup message to the mashup manager. The mashup manager will process each widget in turn by executing the corresponding service. The retrieval result will be mashed up to return mashup data.

During the design phase, users can edit (update or delete) the loaded widget, or load the new one. In addition, users can run a specific widget to preview the mashup data instead of running the whole mashup workflow.

### 6.1.8 Mashup Use Cases

As a proof of concept, the proposed approach has been applied to some mashup use cases that integrate personal resources in SemanticLIFE and SocialLIFE in our mashup platform.

#### Use Case 1: Personal Finance Mashup

According to a statement of Tim Berners-Lee, *“There is lots of data we all use every day, and it is not part of the web. I can see my bank statements on the web, and my photographs, and I can see my appointments in a calendar. But can I see my photos in a calendar to see what I was doing when I took them? Can I see bank statement lines in a calendar?”*

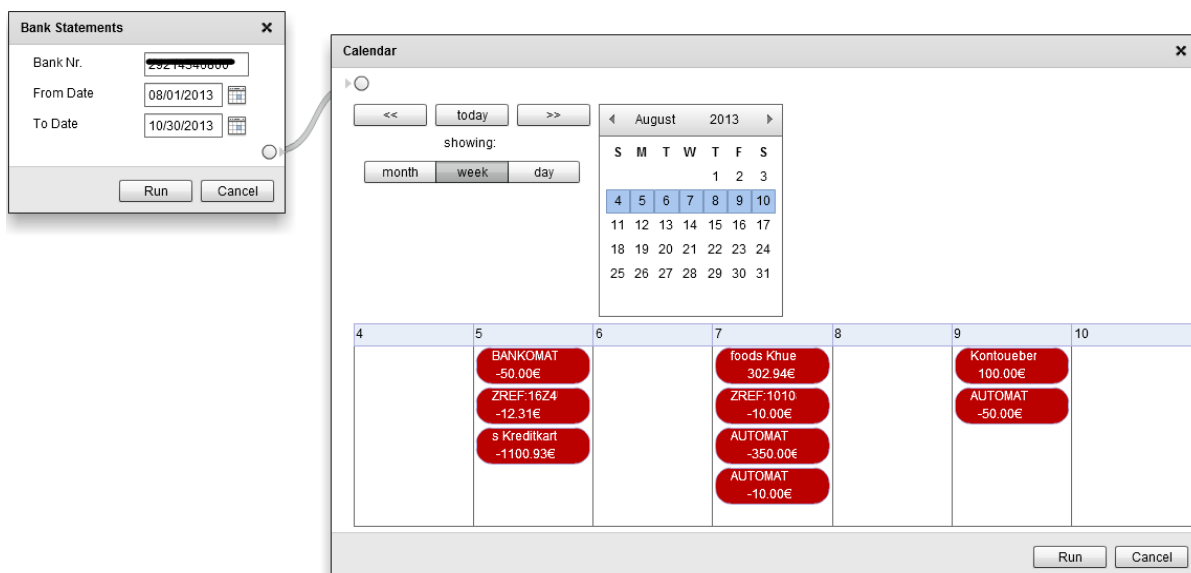
*Why not? Because we do not have a web of data. Because data is controlled by applications, and each application keeps it to itself”*

Besides, you might have many bank accounts and want to keep a record of different bank transactions from those accounts in a single view such as a calendar view.

For addressing this use case, the mashable bank statements have been prepared as described in section [3.3.1](#). In mashup editor, two following basic widgets are used:

- *Bank Statements widget*: retrieves bank statements for a bank account in a specific time.
- *Calendar widget*: shows details of bank statements in a single and flexible view.

The following figure depicts for the personal finance mashup.



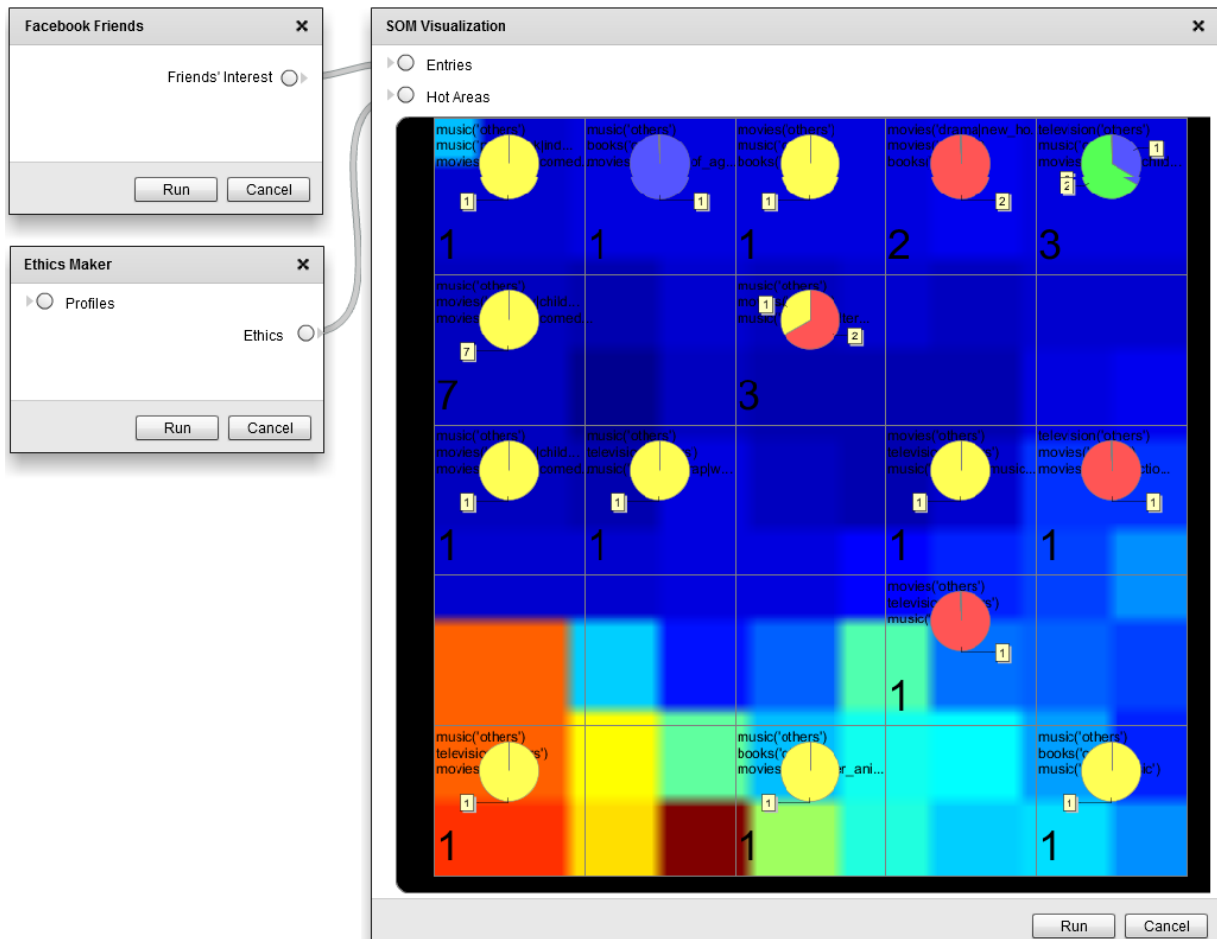
**Figure 6.10:** Personal finance mashup for showing bank statements in calendar view

## **Use case 2: Self-monitoring in SocialLIFE**

In Facebook environment, each user usually has a number of friends in his/her network, where each friend has a collection of interests including books, movies, music, etc. Many users may lose the overview of what their friend are doing and this might lead to incorrect judgments when the interest of a friend is out-of-favor. A simple mashup is requested to create a SOM of all types of interests, in which friends of a specific user are interested. The SOM map is then highlighted according to user ethics of relevant interest of this user. The high risk groups of friends are highlighted on the interest map of friends to help the user find out the inappropriate connections in his/her Facebook profile.

For realizing this self-monitoring use case, the user's network of friends in Facebook is scanned and all interests will be extracted:

- At first, the data from Facebook API for a specific user has been extracted. The following categories of interest have been considered: books, music, movies, and television.
- In the next step, the interest items are annotated with relevant categories such as books, music types, movies genres, or television shows. For a better classification, the genres of the music bands, television shows, movies and books are used instead of their titles (for example, movie Heroes has the genre of Drama/Sci-Fi). The mashable resources for this use case have been retrieved and extracted as described in details in section [6.1.1](#).
- Finally, a classification of friends according to their interest will be displayed on SOM visualization as depicted in the following figure.



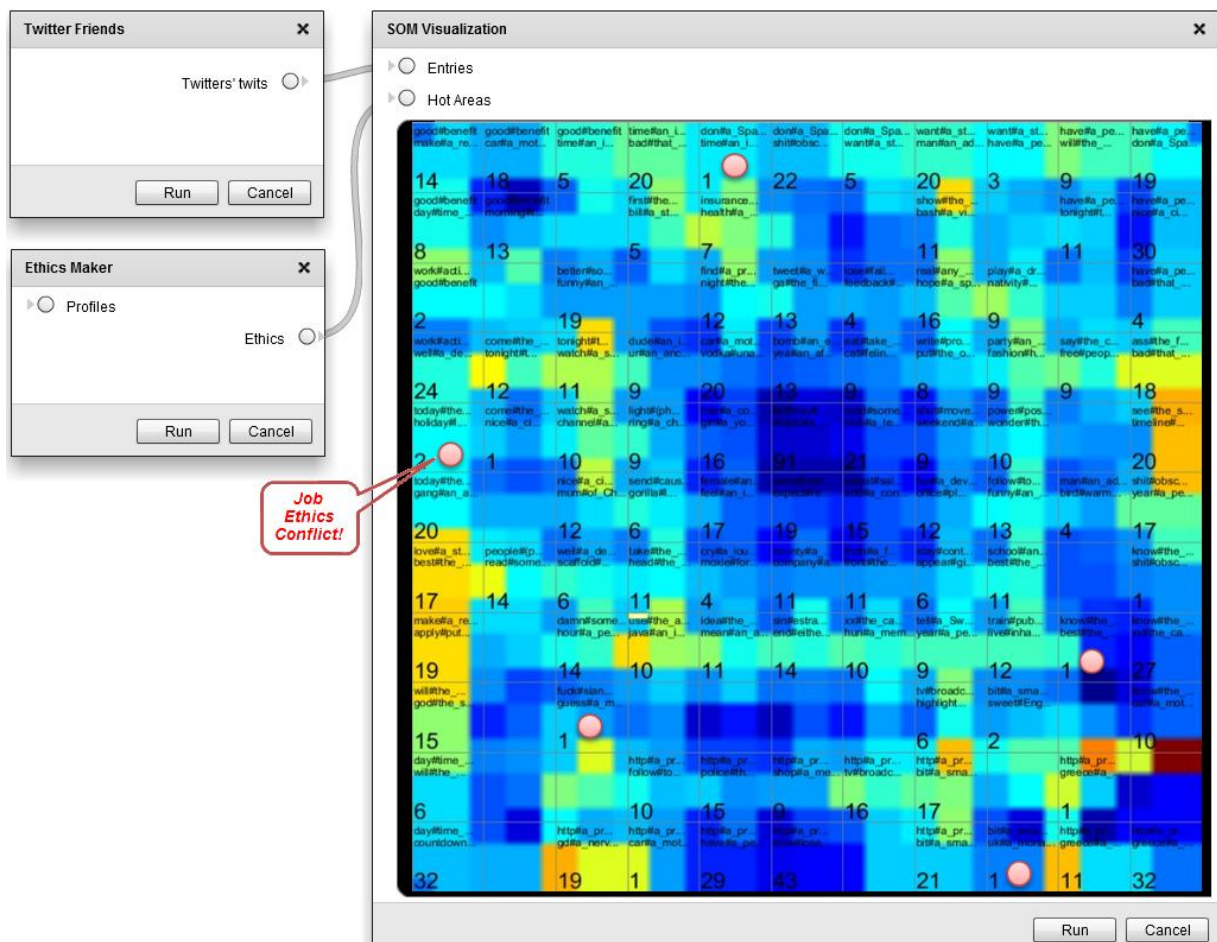
**Figure 6.11:** SOM visualization and clustering of friends' interest from Facebook.

In this use case, the following widgets have been used:

- *Interest of friends in Facebook:* this widget retrieves the annotated interest of friends in Facebook.
- *Ethics Maker:* according to user's ethics, the high risk groups of friends are highlighted on the interest map of friends to help the user find out inappropriate connections in his/her Facebook profile.
- *SOM Visualization:* the interest of each friend and classification of friends according to their interest are visualized in a widget. In this SOM map, each category will be classified by a different color and displayed in a circle (movies in pink, books in cyan, television in green and music in yellow). The small yellow number and the big black number indicate the number of friends who are interested in the relevant category and in the same genre of categories, respectively.



Similarly, the proposed approach has also been applied to Twitter use case. In this use case, the tweets of a user are extracted from Twitter using Twitter API [154]. In the next step, the words are disambiguated and visualized in a SOM. This step uses the top 1000 frequent words that have been occurring in the input tweets. In this use case, the highlighted red points are those areas that are violating the job ethics in organizations/enterprises context. The result of this process is depicted in the following figure.



**Figure 6.12:** SOM visualization and clustering of friends' tweets from Twitter for self-monitoring.



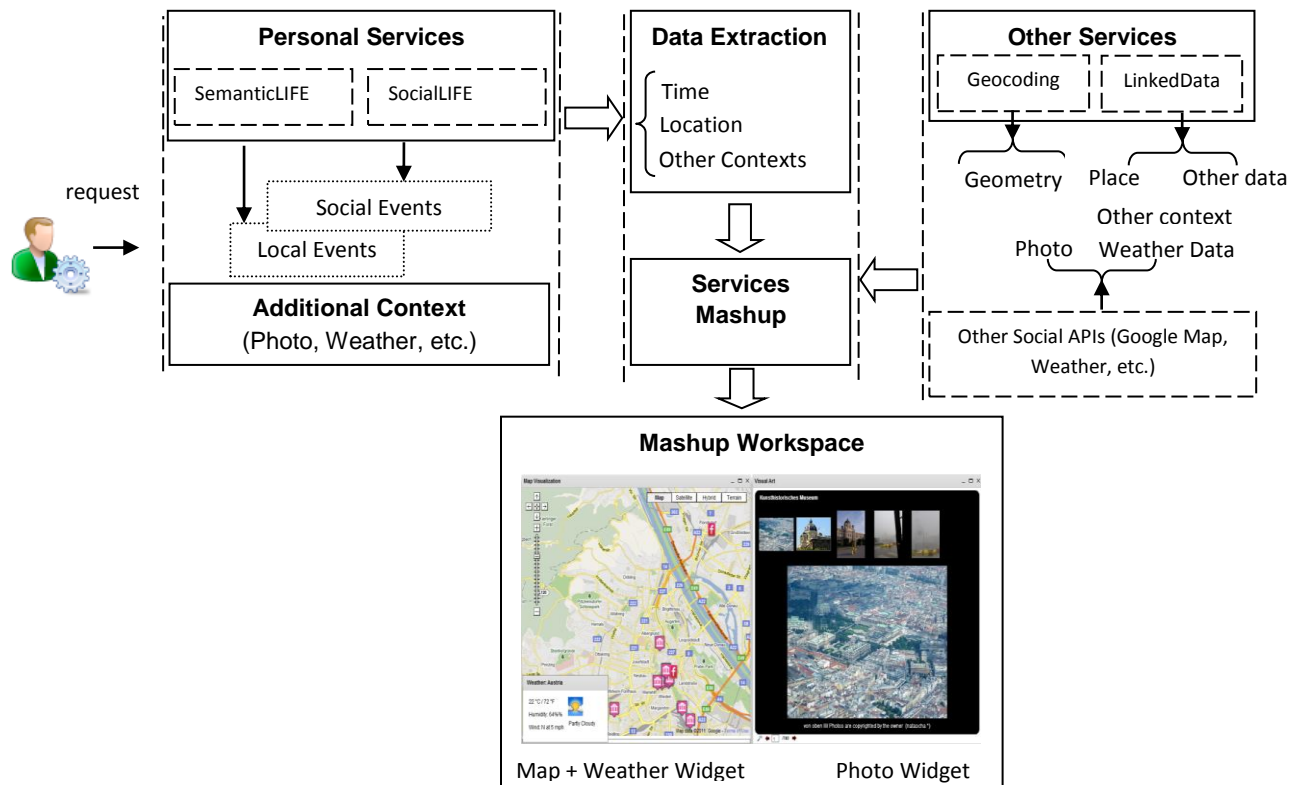
## Use case 2: Personalized Mashup

*I want to check events in my local calendar as well as from my SNSs in a specific time. For a specific event, show me some famous tourist attractions in the location of that event, including some photos (if available), other additional contextual information (weather condition, political status, my social friends, etc.), and show all information on my mashup portal.*

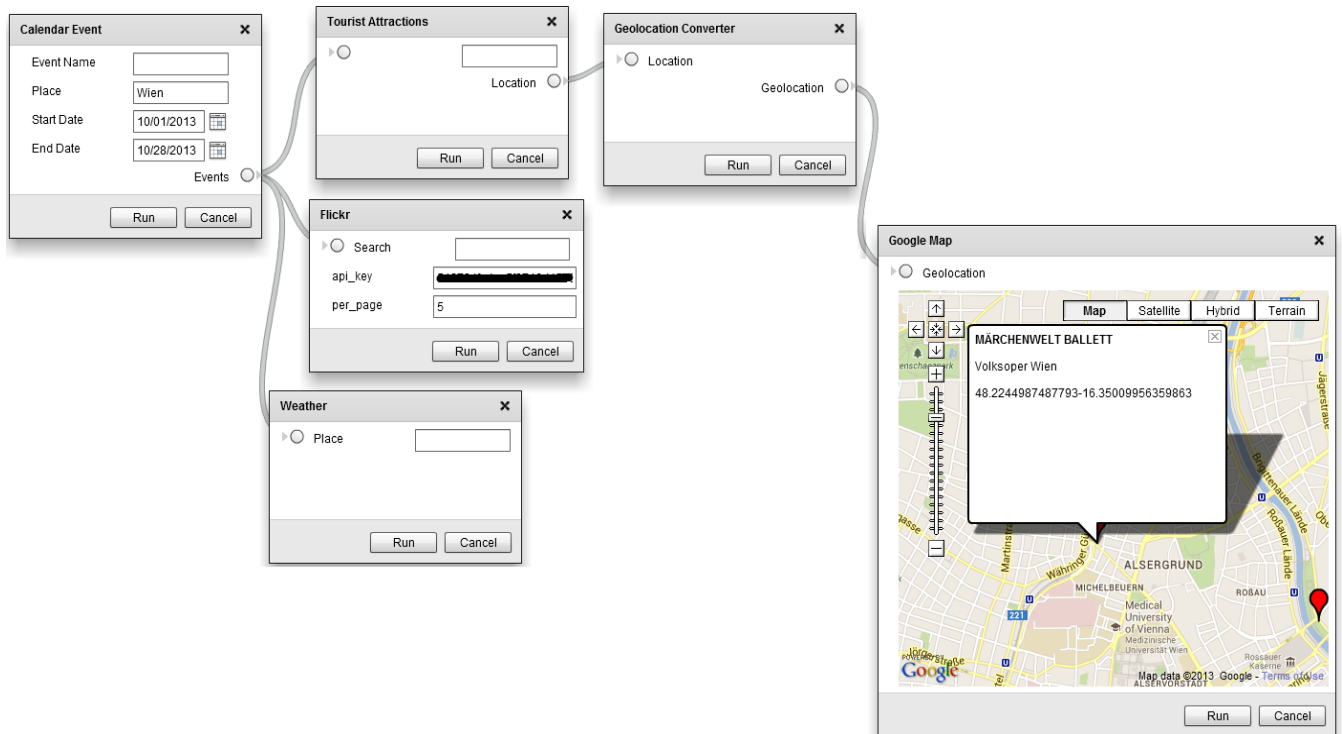
In this context, it is necessary to analyze and map the above information into appropriate resources and services as follows:

- The next event (time, location) can be retrieved in the user's event from both SemanticLIFE and SocialLIFE (i.e. Facebook in this case).
- The famous tourist attractions near the retrieved event location are queried from SNSs API (Freebase, DBPedia, Flickr APIs) and shown on the map if their geocoding data is available.
- If the weather condition is required, the weather widget will provide forecast data for the given location (Google Weather API).

The following figure illustrates the overview for this use case.



**Figure 6.13:** A personalized mashup use case on demand



**Figure 6.14:** Design personalized mashup in Mashup Editor

With such a personalized mashup, the mashup editor will contain six widgets (as depicted in the above figure):

- *Calendar Events widget*: retrieves personal calendar events in SemanticLIFE or SocialLIFE (i.e., Facebook events). This widget will return the events with relevant information (e.g. locations or organizer).
- *Tourist attractions widget*: calls third party service to return information about Places based on the input location. This service may be a query to DBpedia via its dedicated SPARQL Endpoint.
- *Geolocation converter*: is a widget that converts the place address into the geolocation format for viewing in Google map.
- *Google map widget*: shows the obtained places in map visualization.
- *Flickr widget*: is a third party service of Flickr to search photos that match some criteria, in this use case, the matching condition will be the place name.
- *Weather widget*: shows the weather forecast of the given place.

## 6.2 Mashup Framework Evaluation

Several evaluation frameworks [155], [156], [157], [158], [159] have tried to evaluate existing mashup tools and existing approaches in different dimensions. In the scope of this research, the mashup framework is evaluated preliminarily using the following evaluation features.

### 6.2.1 Mashup Framework Components

The primary feature of mashup framework is the requirement of components [159], [160] [161], in which three basic requirements are integration of existing services and information, data aggregation in server-side, and information presentation in client-side. In addition, mashup framework is required providing easy integration of existing mashup components as well as an efficient allocation of mashup components. These components can be mashed up via generic APIs or SPARQL Endpoint that retrieve data from different sources.

Our approach is also conducted in a similar way to deal with the major design characteristics of mashup applications:

- *Integrate existing services and information:* SPARQL query language and REST services are the key services in our framework for querying semantic data that linked data stored in RDF format and third party services from SNSs, respectively.
- *Aggregate data on the server side:* some common format standards such as JSON, XML are used to provide a simpler format for data aggregation on the server side.
- *Present information on the client side:* the results getting from the server side can be visualized on the client side as image, map, or data grid.

### 6.2.2 Data Retrieval Strategy

Data retrieval strategy is a fundamental step in mashup programming since it makes the data available for being used in mashups [157]. Some mashup tools use screen scraping and access the document object model of the web pages [47], [48], [64]. In some cases, some frameworks require scripting and code handling. As a result, users need to understand RSS, XML, gadgets, JSON, RDF feeds, and mashup query languages [51], [53], [83], [86], [145].

This research has conducted data retrieval and annotate in a semantic way with pre-built widgets that allow users to simply drag and drop them into mashup editor without prior knowledge of RDF, SPARQL, etc.

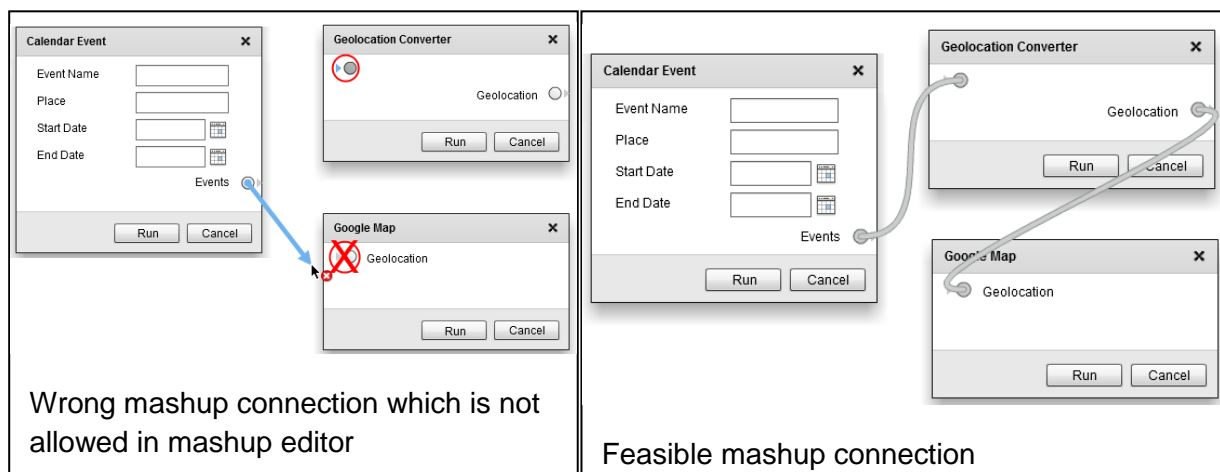
### 6.2.3 Mashup Development Cycle

Mashup framework should focus on the discovery mashable components as the core elements of the development process by enabling the reuse of existing resources in new combinations [159], [161]. With our UI generation mechanism of widgets, developers can reuse or extend existing widget and services for further development.

### 6.2.4 Simple User Interaction Mechanism

One of the main objectives of mashup framework is to provide a graphical and simple user interaction that abstracts the users from the underlying resources and the corresponding technical interfaces [162]. Although the current mashup developments focus mainly on the data aggregation that aims at automating or semi-automating mashup development to serve non-programmers, the end-users are still required a basic level of programming knowledge, such as identifying parameters for operators, loops problems, if-else-then, strings comparison, or some SPARQL syntax [50], [84].

Most of the current mashup tools provide a mashup window containing SPARQL queries to support in querying RDF data sources and web feeds. Some mashup tools use a specific mashup language that is not easy to understand for end-users without prior knowledge of that language or extensive programming skills. In our solution, end-users can be non-programmers but can easily find the correct widgets on the fly. The following figure depicts the semantic-aware dataflow of input/output connections between widgets. This feature allows users to find feasible widget connections easily.



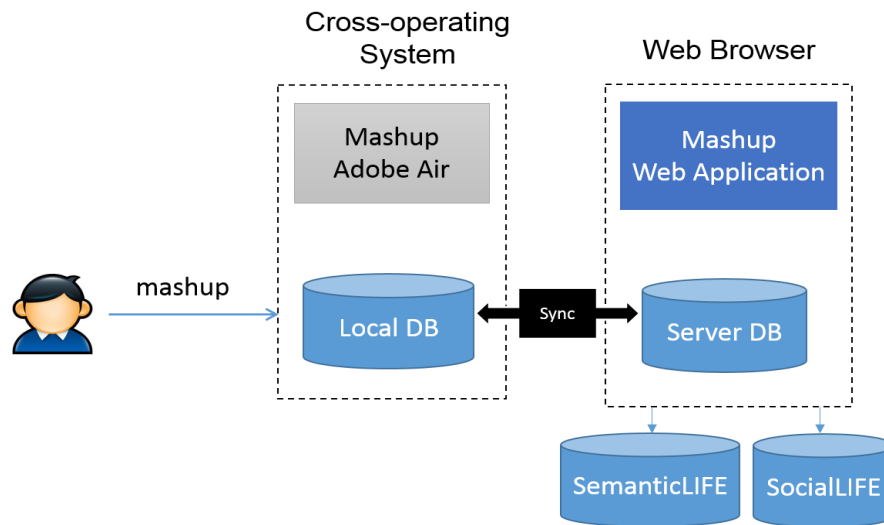
**Figure 6.15:** Semantic-aware dataflow

### 6.2.5 Security and Privacy Policy

The major risks in creating mashup are security and privacy policies [157], [162], [122]. Due to involvement of servers, the security and privacy policies such as cookies issues should be handled with care. This always happens in case of server-based mashup tools [50]. If mashup applications are developed and used as browser add-ons [64], [66], they can address these issues. However, those browser add-ons are required to solve the problem of supporting multiple web browsers.

In our implementation, an open source application framework of Adobe Air and Flex is applied. This platform supports common design patterns, which are suitable for all major browsers, desktops and multiple devices. Our framework has also applied WSD and SOM techniques to support users in personal resources clustering, self-monitoring mashup data and avoid unwanted information in SNSs via a visualization way.

The following figure demonstrates the perspective of running in multiple environments for our mashup platform.



**Figure 6.16:** Running mashup platform in multiple environments perspective.

### 6.2.6 Integrating SemanticLIFE and SocialLIFE data

The last major goal of this mashup framework is integrating SemanticLIFE and SocialLIFE data. With this feature, mashup technology is used as an effective way to facilitate the integration of personal resources on Semantic Desktops and SNSs.

The following table shows some existing features and points out some advanced features of our semantic-based mashup system compared with other mashup products. The most noticeable features in our system are semantic-aware dataflow, self-monitoring mashup, and integrating Semantic Desktops and SNSs (integrate SemanticLIFE and SocialLIFE in particular).

Features	Our semantic-based mashup system	Yahoo Pipes	JackBe	Dapper	DERI Pipes
Data Retrieval	√	√	√	√	√
Data Aggregation	√	√	√		√
Data Flow	√	√	√		√
Semantic data	√				√
Semantic-aware dataflow	√				
Reuse/extend mashup component	√	√	√	√	√
Self-monitoring mashup	√				
Integrate Semantic Desktops and SNSs	√				

**Table 6.1:** Advanced features in semantic-based mashup system compared with other mashup products.

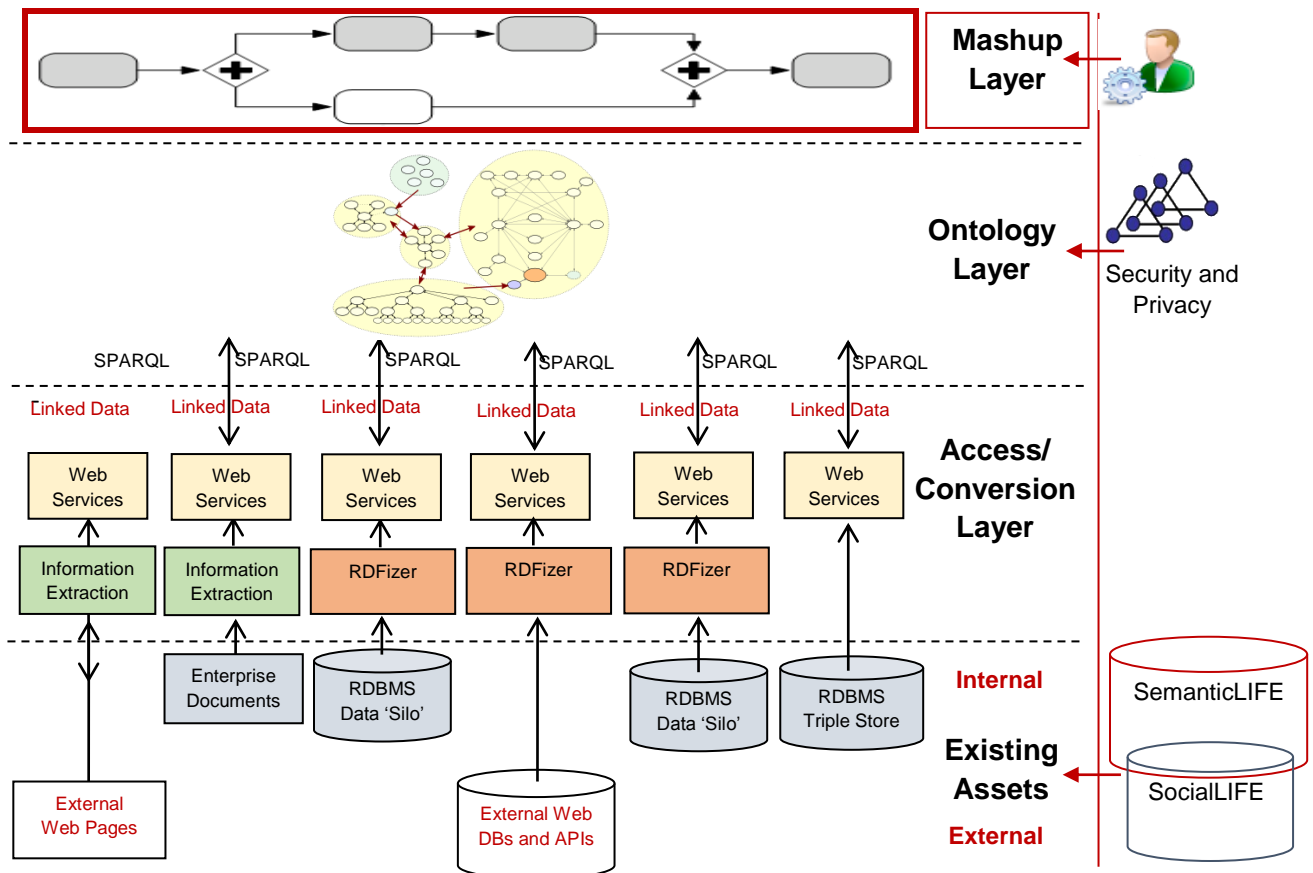
## CONCLUSION AND FUTURE WORK

### 7.1 Conclusion

Considering the information overload issues both on Semantic Desktops and SNSs, this research aims to use semantic metadata for information integration and semantic-based mashups to benefit the personal resources. This research proposed a lightweight mashup language and semantic-based mashup framework for creating personal resources mashup solutions. In addition, a solution of self-monitoring for mashable resources is proposed. Although the implementation of this mashup framework is in progress, and some more components are still needed, the implemented prototype shows the efficiency of the proposed approach. The research contributions of this research work can be summarized as follows:

- Integrating personal resources in SemanticLIFE and SocialLIFE
  - o Expanding the scope of SemanticLIFE into the web of data instead of isolated data silos on the desktop.
  - o Bridging the gap between SemanticLIFE and SocialLIFE in order to integrate and reuse existing personal resources in different use cases.
- Building semantic-based mashup system
  - o *Lightweight mashup language*: creates a simple mashup language to help users to develop widgets based on their required context in a semantic way.
  - o *Main components of a mashup framework*: the major components of a mashup framework such as mashup editor, mashup portal have been developed. These components facilitate the integration of existing services and personal resources.
  - o *Simple mashup mechanism*: supports end-users to create mashup solutions based on pre-built widgets and create new widgets that inherit predefined widgets.

- *Semantic-aware mashup dataflow*: users may easily create mashups and find the connectable ports via the semantic-aware input/output ports of widgets on the fly.
- *Self-monitoring mashup data*: support users in self-monitoring their social data to get rid of unintentional risks or sensitive information leakages.
- Utilizing Semantic Web, Linked Data, and mashup technologies towards the layered approach of Open Semantic Enterprise (OSE) and semantic information integration. Our proposed solutions can be adapted to OSE as depicted in the following figure.



**Figure 7.1:** The vision of adapting mashup application in the layered approach of Open Semantic Enterprise.



In this section, the research questions are revisited to show how the proposed solution can address these challenging issues.

**RQ1.** *How to apply a semantic-driven approach that integrates personal life items in Semantic Desktops and SNSs in order to benefit individual, collaborative work and better solve business processes in organizations/enterprises?*

To answer this research question, Chapter 3 presented the mashable resources solution by applying semantic and Linked Data technologies for linking the personal resources in Semantic Desktops and SNSs. In addition, a number of existing vocabularies of LOD cloud and adapted ontologies have been used such as FOAF, Geolocation, DBpedia's datasets, and OFX – ontology for personal finance. With the combination of these technologies, a better linking and sharing mechanism between data resources and information have been provided in which the shared information is not just textual contents but also multimedia contents (such as images, videos, etc. ).

**RQ2.** *How to secure mashable resources that can be combined with personal/organization policies in order to protect and filter sharing data in a collaborative environment of enterprise?*

To address the security-related issues in data mashups, Chapter 4 applied the WSD and SOM techniques to help users in self-monitoring and avoiding unwanted information leakage, as well as identifying trustworthy mashable resources. In addition, with pre-built widgets that have been developed in our framework, users can have a better overview of information resources.

**RQ3.** *How to create a semantic-based unified mashup model to support end-users in the fast creation of data mashups and to fulfill users' requirements on demand for enterprise?*

Chapter 5 described a semantic-based mashup framework where end-users and mashup developers collaborate to design mashup on demand. To illustrate how mashup can be designed and executed, some example use cases were described and implemented. Those use cases typically prove the semantic-based mashup possibilities in mashup framework as presented in Chapter 6.

## **7.2 Future Work**

In the next studies, the development of our proposed system should be continued to improve the mashup language and self-monitoring components. Automatic widget composition would also be a major task in the future research of semantic mashup. The mashup sharing issues should be done in order to support end-users in collaborating to build data mashups in enterprises.

Besides, more third party APIs of SNSs should be exploited and implemented in order to enrich personal resources and the mashup repository as well. In addition, further experiments for performance and usability evaluation should be conducted. Finally, more mashup scenario should be carried out to benefit the business value of knowledge worker resources, towards Open Semantic Enterprise context in future.

## Bibliography

- [1] L. Baird and I. Meshoulam, "Managing Two Fits of Strategic Human Resource Management," *Academy of Management Review*, vol. 13, 1988.
- [2] D. Pauleen, "Personal knowledge management: Putting the 'person' back into the knowledge equation," *Online Inf. Rev.*, vol. 33, no. 2, pp. 221–224, 2009.
- [3] T. Berners-Lee, H. James, and L. Ora, "The Semantic Web," 2001. [Online]. Available: <http://www.scientificamerican.com/article.cfm?id=the-semantic-web>. [Accessed: 26-Mar-2008].
- [4] D. E. O'Leary, "Enterprise knowledge management," *Computer (Long. Beach. Calif.)*, vol. 31, no. 3, pp. 54–61, 1998.
- [5] A. McAfee, "Enterprise 2.0: The Dawn of Emergent Collaboration," *MIT Sloan Management Review*, 2006. [Online]. Available: <http://sloanreview.mit.edu/article/enterprise-the-dawn-of-emergent-collaboration/>. [Accessed: 11-Sep-2012].
- [6] D. Hinchcliffe, "Enterprise 2.0: Finding success on the frontiers of social business," 2009. [Online]. Available: <http://blogs.zdnet.com/Hinchcliffe/?p=744>. [Accessed: 25-Sep-2012].
- [7] D. Hinchcliffe, "14 Reasons Why Enterprise 2.0 Projects Fail," 2009. [Online]. Available: <http://blogs.zdnet.com/Hinchcliffe/?p=718>. [Accessed: 25-Sep-2012].
- [8] E. Metter, T. Perrin, V. Gyster, and R. Lamson, "Enterprise 2.0 and HR - Realizing the Potential," *IHRIM J.*, vol. XII, no. 5, 2008.
- [9] A. Corriveau, "The Informal Organization." [Online]. Available: [http://www.businesswire.com/portal/site/google/index.jsp?ndmViewId=news\\_view&newsId=20070731005934&newsLang=en](http://www.businesswire.com/portal/site/google/index.jsp?ndmViewId=news_view&newsId=20070731005934&newsLang=en). [Accessed: 15-Aug-2012].
- [10] Tim Berners-Lee, "Linked Data," 2006. [Online]. Available: <http://www.w3.org/DesignIssues/LinkedData.html>. [Accessed: 22-Sep-2012].
- [11] C. Newmark, "Craigslist," 2008. [Online]. Available: [www.craigslist.org](http://www.craigslist.org). [Accessed: 04-Dec-2013].
- [12] A. Anjomshoaa, "Integration of Personal Services into Global Business," Vienna University of Technology, Vienna, Austria, 2009.

- [13] M. Ahmed, H. H. Hoang, M. S. Karim, S. Khusro, M. Lanzenberger, K. Latif, E. Michlmayr, K. Mustofa, H. T. Nguyen, A. Rauber, A. Schatten, T. M. Nguyen, A. M. Tjoa, and T. A. M. Ahmed M, Hoang H H, Karim M S, Khusro S, Lanzenberger M, Latif K, Michlmayr E, Mustofa K, Nguyen H T, Rauber A, Schatten A., Tho M. N., “SemanticLIFE’ - A Framework for Managing Information of A Human Lifetime.,” in *iiWAS*, 2004, vol. 183, no. October 2004.
- [14] Isidro Laso Ballesteros, “New Collaborative Working Environments 2020 - Report on industry-led FP7 consultations,” Bruxelles, 2006.
- [15] T. O’Reilly, “What is Web 2.0?,” 2005. [Online]. Available: <http://oreilly.com/web2/archive/what-is-web-20.html>. [Accessed: 26-Jun-2011].
- [16] A. McAfee, “Enterprise 2.0, version 2.0.” [Online]. Available: [http://andrewmcafee.org/2006/05/enterprise\\_20\\_version\\_20/](http://andrewmcafee.org/2006/05/enterprise_20_version_20/).
- [17] B. Hyland, “Preparing for a Linked Data Enterprise,” in *Linking Enterprise Data SE - 3*, D. Wood, Ed. Springer US, 2010, pp. 51–64.
- [18] D. Fichter, “What is a Mashup?,” 2005. [Online]. Available: <http://books.infotoday.com/books/Engard/Engard-Sample-Chapter.pdf>.
- [19] JackBe, “A Business Guide to Enterprise Mashups,” 2008. [Online]. Available: [http://mdc.jackbe.com/downloads/JackBe\\_business\\_guide\\_to\\_enterprise\\_mashups.pdf](http://mdc.jackbe.com/downloads/JackBe_business_guide_to_enterprise_mashups.pdf). [Accessed: 15-Aug-2012].
- [20] J. G. Breslin, Alexandre Passant, and Stefan Decker, *The Social Semantic Web*. Springer, 2009.
- [21] G. Bader, A. Anjomshoaa, and A. M. Tjoa, “Privacy Aspects of Mashup Architecture,” *2010 IEEE Second Int. Conf. Soc. Comput.*, pp. 1141–1146, Aug. 2010.
- [22] ProgrammableWeb.com, “ProgrammableWeb - Mashups, APIs, and the Web as Platform,” 2013. [Online]. Available: <http://www.programmableweb.com/>.
- [23] S. Peenikal, “Mashups and the Enterprise,” *White Paper*, 2009.
- [24] V. Hoyer and K. Stanoevska-Slabeva, “Design Principles of Enterprise Mashups,” in *Wissensmanagement*, 2009, pp. 242–253.
- [25] D. E. Simmen, M. Altinel, V. Markl, S. Padmanabhan, and A. Singh, “Damia - Data Mashups for Intranet Applications,” in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data - SIGMOD ’08*, 2008, p. 1171.

- [26] P. De Vrieze, L. Xu, A. Bouguettaya, J. Yang, and J. Chen, "Process-Oriented Enterprise Mashups," *2009 Work. Grid Pervasive Comput. Conf.*, pp. 64–71, May 2009.
- [27] M. Ogrinz, *Mashup Patterns - Designs and Example for the Modern Enterprise*. Addison Wesley, 2009, p. 428.
- [28] G. Bader, A. Anjomshoaa, and Am. M. Tjoa, "A Context-Aware Mashup Integration Guideline for Enterprise 2.0," in *Multidisciplinary Research and Practice for Information Systems SE - 2*, vol. 7465, G. Quirchmayr, J. Basl, I. You, L. Xu, and E. Weippl, Eds. Springer Berlin Heidelberg, 2012, pp. 17–30.
- [29] M. K. Bergman, "Seven Pillars of the Open Semantic Enterprise," 2010. [Online]. Available: <http://www.mkbergman.com/859/seven-pillars-of-the-open-semantic-enterprise/>. [Accessed: 16-Jan-2011].
- [30] H. B. Guentner Georg, "The Open Semantic Enterprise - Enterprise Data meets Web Data," *2nd B2B Software Days*. Vienna, Austria.
- [31] P. F. Drucker, *The Effective Executive: The Definitive Guide to Getting the Right Things Done*. Harperbusiness Essentials, 1967.
- [32] P. F. Drucker, *Management Challenges for the 21st Century*. Harper Business, 1999, p. 207.
- [33] Doculabs, "Social Computing and Collaboration for the Enterprise - Enabling Knowledge Worker Productivity," *White Paper*, 2010.
- [34] M. Platt, "Web 2.0 in the Enterprise," *Architecture Journal*, 2007. [Online]. Available: <http://msdn.microsoft.com/en-us/library/bb735306.aspx>. [Accessed: 07-Mar-2012].
- [35] D. Quan, D. Huynh, and D. R. Karger, "Haystack: A Platform for Authoring End User Semantic Web Applications," in *Second International Semantic Web Conference (ISWC 2003)*, vol. ISCW, D. Fensel, K. Sycara, and J. Mylopoulos, Eds. Sanibel Island, FL, USA: Springer, 2003, pp. 738–753.
- [36] G. Tummarello, C. Morbidoni, M. Nucci, D. Elettronica, and I. Artificiale, "Enabling Semantic Web Communities with DBin: An Overview," in *The Semantic Web - ISWC 2006 SE - 69*, vol. 4273, I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, Eds. Springer Berlin Heidelberg, 2006, pp. 943–950.
- [37] J. Iturrioz, S. F. Anzuola, and O. Díaz, "Turning the Mouse into a Semantic Device: The seMouse Experience," in *The Semantic Web: Research and*

*Applications SE* - 34, vol. 4011, Y. Sure and J. Domingue, Eds. Springer Berlin Heidelberg, 2006, pp. 457–471.

- [38] DFKI, “Gnowsis - Semantic Desktop environment.” [Online]. Available: <http://www.gnowsis.org/>. [Accessed: 25-Sep-2012].
- [39] T. Groza, S. Handschuh, K. Moeller, G. A. Grimnes, L. Sauermann, E. Minack, C. Mesnage, M. Jazayeri, G. Reif, and R. Gudjonsdottir, “The NEPOMUK Project - On the way to the Social Semantic Desktop,” in *Proceedings of International Conferences on new Media technology (I-MEDIA-2007) and Semntic Systems (I-SEMANTICS-07)*, Graz, Austria, September 5-7., 2007, pp. 201–210.
- [40] A. Cheyer, J. Park, and R. Giuli, “IRIS: Integrate. Relate. Infer. Share,” *1st Work. Semant. Desktop. 4th Int. Semant. Web Conf.*, 2005.
- [41] X. L. Dong and A. Halevy, “A Platform for Personal Information Management and Integration,” in *Second Biennial Conference on Innovative Data Systems Research (CIDR 2005)*, 2005.
- [42] S. Handschuh and S. Decker, “The Semantic Desktop at Work : Interlinking Notes,” in *Proceedings of the 7th International Conference on Semantic Systems (I-Semantics '11)*, 2011, pp. 17–24.
- [43] M. Ahmed, “Context-Based Privacy Management of Personal Information Using Semantic Desktop : SemanticLIFE Case Study,” in *Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services (iiWASS '08)*, 2008, no. c, pp. 214–221.
- [44] C. Dan, “Guiding Principles for the Open Semantic Government,” 2010. [Online]. Available: [http://docwiki.citizen-dan.org/index.php/Guiding\\_Principles\\_for\\_the\\_Open\\_Semantic\\_Government](http://docwiki.citizen-dan.org/index.php/Guiding_Principles_for_the_Open_Semantic_Government). [Accessed: 25-Sep-2012].
- [45] S. Bridges, J. Schiffl, and S. Polovina, “OpenSEA : A Framework for Semantic Interoperation between Enterprises,” in *Next Generation Data Technologies for Collective Computational Intelligence Studies in Computational Intelligence*, vol. 352, 2011, pp. 61–86.
- [46] A. Passant, P. Laublet, J. G. Breslin, and S. Decker, “Enhancing Enterprise 2.0 Ecosystems Using SemanticWeb and Linked Data Technologies: The SemSLATES Approach,” no. Linking Enterprise Data, D. Wood, Ed. Boston, MA: Springer US, 2010, pp. 79–102.

- [47] D. F. Huynh, D. R. Karger, and R. C. Miller, "Exhibit: lightweight structured data publishing," in *Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 737–746.
- [48] MIT, "SIMILE Project," 2011. [Online]. Available: <http://simile.mit.edu/>. [Accessed: 20-Sep-2012].
- [49] Yahoo Dapper, "Dapper: The Data Mapper," 2012. [Online]. Available: <http://open.dapper.net/>. [Accessed: 22-Sep-2012].
- [50] Yahoo Pipes, "Pipes: Rewire the web," 2012. [Online]. Available: <http://pipes.yahoo.com/pipes/>. [Accessed: 22-Sep-2012].
- [51] JackBe, "JackBe Presto," 2012. [Online]. Available: <http://jackbe.com/products/presto>. [Accessed: 22-Sep-2012].
- [52] S. Mayers and M. Lee, "Mac OS X Automation with Automator and AppleScript," in *Learn OS X Lion SE - 31*, Apress, 2011, pp. 593–627.
- [53] WSO2, "WSO2 Mashup Server—where to now?," 2012. [Online]. Available: <http://wso2.com/blogs/thesource/2012/12/wso2-mashup-serverwhere-to-now/>. [Accessed: 22-Sep-2012].
- [54] M. Altinel, P. Brown, S. Cline, R. Kartha, E. Louie, V. Markl, L. Mau, Y.-H. Ng, D. Simmen, and A. Singh, "Damia: a data mashup fabric for intranet applications," in *Proceedings of the 33rd international conference on Very large data bases*, 2007, pp. 1370–1373.
- [55] IBM, "Software withdrawn: IBM Mashup Center," 2012. [Online]. Available: [http://www-01.ibm.com/common/ssi/rep\\_ca/8/897/ENUS912-068/ENUS912-068.PDF](http://www-01.ibm.com/common/ssi/rep_ca/8/897/ENUS912-068/ENUS912-068.PDF). [Accessed: 22-Sep-2012].
- [56] Synergex, "Serena Business Mashups," 2008. [Online]. Available: [http://pvcs.synergex.com/products/serena\\_business\\_mashups.aspx](http://pvcs.synergex.com/products/serena_business_mashups.aspx). [Accessed: 20-Oct-2012].
- [57] Romulus, "MyCocktail Mashup Builder," 2010. [Online]. Available: <http://www.ict-romulus.eu/web/mycocktail>. [Accessed: 20-Oct-2012].
- [58] Omelette, "Omelette Project," 2009. [Online]. Available: <http://www.ict-omelette.eu/home>. [Accessed: 20-Sep-2012].
- [59] M. Imran, F. Kling, S. Soi, F. Daniel, F. Casati, and M. Marchese, "ResEval Mash : A Mashup Tool for Advanced Research Evaluation," *Proc. 21st Int. Conf. companion World Wide Web (WWW '12 Companion)*, pp. 361–364, 2012.

- [60] ServFace, "ServFace - Service Annotations for User Interface Composition," 2010. [Online]. Available: <http://www.servface.eu/>. [Accessed: 25-Sep-2012].
- [61] Google, "Google Mashup Editor," 2009. [Online]. Available: <https://developers.google.com/mashup-editor/>. [Accessed: 25-Sep-2012].
- [62] M. Popfly, "Microsoft Popfly," 2007. [Online]. Available: [http://en.wikipedia.org/wiki/Microsoft\\_Popfly](http://en.wikipedia.org/wiki/Microsoft_Popfly). [Accessed: 25-Sep-2012].
- [63] R. J. Ennals and M. N. Garofalakis, "MashMaker: Mashups for the Masses," in *Proceedings of the 2007 ACM SIGMOD international conference on Management of data - SIGMOD '07*, 2007, p. 1116.
- [64] R. Ennals, E. Brewer, M. Garofalakis, M. Shadle, and P. Gandhi, "Intel Mash Maker: Join the Web," *ACM SIGMOD Rec.*, vol. 36, no. 4, p. 27, Dec. 2007.
- [65] M. Kaply, "Operator - Add-ons for Firefox," 2010. [Online]. Available: <https://addons.mozilla.org/en-us/firefox/addon/operator/>. [Accessed: 25-Sep-2012].
- [66] D. Huynh, S. Mazzocchi, and D. Karger, "Piggy Bank: Experience the Semantic Web Inside Your Web Browser," in *The Semantic Web – ISWC 2005 SE - 31*, vol. 3729, Y. Gil, E. Motta, V. R. Benjamins, and M. Musen, Eds. Springer Berlin Heidelberg, 2005, pp. 413–430.
- [67] MIT, "Piggy Bank," 2011. [Online]. Available: [http://simile.mit.edu/wiki/Piggy\\_Bank](http://simile.mit.edu/wiki/Piggy_Bank). [Accessed: 20-Oct-2012].
- [68] T. Berners-lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets, "Tabulator: Exploring and analyzing linked data on the semantic web," *Proc. 3rd Int. Semant. Web User Interact. Work.*, 2006.
- [69] B. Hartmann, L. Wu, K. Collins, S. R. Klemmer, and S. R. Klemmer, "Programming by a Sample: Rapidly Prototyping Web Applications with d.mix," in *Proceedings of the 20th annual ACM symposium on User interface software and technology - UIST '07*, 2007, pp. 241–250.
- [70] S. Abiteboul, O. Greenshpan, and T. Milo, "Modeling the mashup space," *Proceeding 10th ACM Work. Web Inf. data Manag. - WIDM '08*, p. 87, 2008.
- [71] S. Ikeda, T. Nagamine, and T. Kamada, "Application Framework with Demand-Driven Mashup," *J. Univers. Comput. Sci.*, vol. 15, no. 10, pp. 2109–2137, 2009.



- [72] M. Vasko and S. Dustdar, "Introducing collaborative Service Mashup design," in *Lightweight Integration on the Web (ComposableWeb'09)*, 2009, no. April, pp. 51–62.
- [73] R. Gurram, B. Mo, and R. Gueldemeister, "A Web Based Mashup Platform for Enterprise 2.0," in *Web Information Systems Engineering – WISE 2008 Workshops SE - 17*, vol. 5176, S. Hartmann, X. Zhou, and M. Kirchberg, Eds. Springer Berlin Heidelberg, 2008, pp. 144–151.
- [74] O. S. Inc, "OpenLink iSPARQL," 2006. [Online]. Available: <http://dbpedia.org/isparql/>. [Accessed: 20-Sep-2012].
- [75] D. F. Huynh, R. C. Miller, and D. R. Karger, "Potluck: Data mash-up tool for casual users," *Web Semant. Sci. Serv. Agents World Wide Web*, vol. 6, no. 4, pp. 274–282, Nov. 2008.
- [76] E. Hyvönen, K. Viljanen, J. Tuominen, and K. Seppälä, "Building a National Semantic Web Ontology and Ontology Service Infrastructure –The FinnONTO Approach," in *The Semantic Web: Research and Applications SE - 10*, vol. 5021, S. Bechhofer, M. Hauswirth, J. Hoffmann, and M. Koubarakis, Eds. Springer Berlin Heidelberg, 2008, pp. 95–109.
- [77] A. P. Sheth, K. Gomadam, W. State, and A. P. S. G. Lathem, "SA-REST: Semantically Interoperable and Easier-to-Use Services and Mashups," no. December, pp. 84–87, 2007.
- [78] B. Biörnstad and C. Pautasso, "Let It Flow: Building Mashups with Data Processing Pipelines," in *Service-Oriented Computing - ICSOC 2007 Workshops*, E. Di Nitto and Matei Ripeanu, Eds. Vienna, Austria: Springer Berlin Heidelberg, 2009, pp. 15–28.
- [79] M. Albinola, L. Baresi, M. Carcano, and P. Milano, "Mashlight : a Lightweight Mashup Framework for Everyone," in *2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009)*, 2009.
- [80] F. Daniel, F. Casati, S. Soi, J. Fox, D. Zancarli, and M.-C. Shan, "Hosted Universal Integration on the Web: The mashArt Platform," in *Service-Oriented Computing SE - 51*, vol. 5900, L. Baresi, C.-H. Chi, and J. Suzuki, Eds. Springer Berlin Heidelberg, 2009, pp. 647–648.
- [81] B. Lu, Z. Wu, C. Zhou, and H. Chen, "sMash : Semantic-based Mashup Navigation for Data API Network," in *Proceedings of the 18th international conference on World wide web - WWW '09*, 2009, pp. 1133–1134.
- [82] O. Greenshpan, T. Milo, N. Polyzotis, and S. Abiteboul, "Autocompletion for mashups," *Proc. VLDB Endow.*, vol. 2, no. 1, pp. 538–549, Aug. 2009.

- [83] H. Knublauch, "SPARQLMotion," 2010. [Online]. Available: <http://sparqlmotion.org/>. [Accessed: 24-Sep-2012].
- [84] D. Le-Phuoc, A. Polleres, G. Tummarello, and C. Morbidoni, "Rapid prototyping of semantic mash-ups through semantic web pipes," *Proc. 18th Int. Conf. World wide web - WWW '09*, p. 581, 2009.
- [85] M. Mostarda and D. Palmisano, "MU : an hybrid language for Web Mashups \*," in *International World Wide Web Conference (WWW2009)*, 2009.
- [86] YQL Yahoo, "Yahoo! Query Language - YDN," 2007. [Online]. Available: <http://developer.yahoo.com/yql/>. [Accessed: 22-Sep-2012].
- [87] M. Jarrar and M. D. Dikaiakos, "MashQL : A Query-by-Diagram Topping SPARQL," in *Proceedings of the 2nd international workshop on Ontologies and information systems for the semantic web (ONISW '08)*, 2008, pp. 89–96.
- [88] H. Hobel, J. Heurix, A. Anjomshoaa, and E. Weippl, "Towards Security-Enhanced and Privacy-Preserving Mashup Compositions," in *Security and Privacy Protection in Information Processing Systems SE - 22*, vol. 405, no. Security and Privacy Protection in Information Processing Systems IFIP Advances in Information and Communication Technology, L. Janczewski, H. Wolfe, and S. Shenoj, Eds. Springer Berlin Heidelberg, 2013, pp. 286–299.
- [89] OMA, "Open Mashup Alliance FAQ," 2009. [Online]. Available: [http://mdc.jackbe.com/downloads/Open\\_Mashup\\_Alliance\\_FAQ\\_for\\_Customers.pdf](http://mdc.jackbe.com/downloads/Open_Mashup_Alliance_FAQ_for_Customers.pdf). [Accessed: 15-Aug-2013].
- [90] Apache, "Apache Rave," 2012. [Online]. Available: <https://rave.apache.org/>. [Accessed: 04-Dec-2013].
- [91] C. Messias, R. Spolon, M. A. Cavenaghi, R. S. Lobato, M. Angel, C. Vaz, and E. Vargas, "Utilizing cross-domain SOAP web services using client- side languages in an Enterprise Mashup Platform," in *Proceedings of the XV Brazilian Symposium on Multimedia and the Web (WebMedia '09)*, 2009, pp. 3–6.
- [92] E. M. Maximilien, A. Ranabahu, and S. Tai, "Swashup," in *Companion to the 22nd ACM SIGPLAN conference on Object oriented programming systems and applications companion - OOPSLA '07*, 2007, p. 797.
- [93] D. H. Hansson, "Ruby on Rail," 2005. [Online]. Available: <http://rubyonrails.org/>. [Accessed: 04-Dec-2013].
- [94] H. Hotta, T. Nozawa, and M. Hagiwara, "A Design of Client Side Information Management Method for Web Services Collaboration," in *Proceeding WI-IATW*

*'07 Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops*, 2007, pp. 95–98.

- [95] S. Zarandioon, D. (Daphne) Yao, and V. Ganapathy, “OMOS: A Framework for Secure Communication in Mashup Applications,” *2008 Annu. Comput. Secur. Appl. Conf.*, pp. 355–364, Dec. 2008.
- [96] F. De Keukelaere, S. Bhola, M. Steiner, S. Chari, and S. Yoshihama, “SMash: Secure Component Model for Cross-Domain Mashups on Unmodified Browsers,” in *Proceeding of the 17th international conference on World Wide Web - WWW '08*, 2008, p. 535.
- [97] R. Hasan, M. Winslett, R. Conlan, B. Slesinsky, and N. Ramani, “Please Permit Me: Stateless Delegated Authorization in Mashups,” in *2008 Annual Computer Security Applications Conference (ACSAC)*, 2008, pp. 173–182.
- [98] J. Magazinius, A. Askarov, and A. Sabelfeld, “A lattice-based approach to mashup security,” *Proc. 5th ACM Symp. Information, Comput. Commun. Secur. - ASIACCS '10*, p. 15, 2010.
- [99] R. Fox, J. Cooley, and M. Hauswirth, “Collaborative development of trusted mashups,” *Proc. 12th Int. Conf. Inf. Integr. Web-based Appl. Serv. - iiWAS '10*, p. 255, 2010.
- [100] M. Herbert, T. Thieme, J. Zibuschka, H. Roßnagel, and F. Iao, “Secure Mashup-Providing Platforms - Implementing Encrypted Wiring,” in *Current Trends in Web Engineering SE - 9*, vol. 7059, A. Harth and N. Koch, Eds. Springer Berlin Heidelberg, 2012, pp. 99–108.
- [101] J. Zibuschka, M. Herbert, and H. Roßnagel, “Towards Privacy-Enhancing Identity Management in Mashup-Providing Platforms,” in *Data and Applications Security and Privacy XXIV SE - 18*, vol. 6166, S. Foresti and S. Jajodia, Eds. Springer Berlin Heidelberg, 2010, pp. 273–286.
- [102] V. Bush, “As We May Think,” 1945. [Online]. Available: <http://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/>. [Accessed: 01-Nov-2012].
- [103] J. G. Breslin, A. Passant, and S. Decker, *The Social Semantic Web*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.
- [104] B. Solis, “Introducing The Conversation Prism,” 2008. [Online]. Available: <http://www.briansolis.com/2008/08/introducing-conversation-prism/>. [Accessed: 04-Nov-2012].

- [105] L. Drăgan, R. Delbru, T. Groza, S. Handschuh, and S. Decker, "Linking Semantic Desktop Data to the Web of Data," *Semant. Web – ISWC 2011 SE - 3*, vol. 7032, pp. 33–48, 2011.
- [106] T. Groza, L. Drăgan, S. Handschuh, and S. Decker, "Bridging the Gap between Linked Data and the Semantic Desktop," *Semant. Web - ISWC 2009 SE - 52*, vol. 5823, pp. 827–842, 2009.
- [107] D. B. and L. Miller, "FOAF Vocabulary Specification 0.97," 2010. [Online]. Available: <http://xmlns.com/foaf/spec/>. [Accessed: 07-Mar-2012].
- [108] A. Passant, "RDF export of Flickr profiles with FOAF and SIOC." [Online]. Available: <http://apassant.net/blog/2007/12/18/rdf-export-of-flickr-profiles-with-foaf-and-sioc/>.
- [109] Kanzaki, "Flickr photo info to RDF image description," 2007. [Online]. Available: <http://www.kanzaki.com/works/2005/imgdsc/flickr2rdf>. [Accessed: 25-Sep-2012].
- [110] C. B. C. Bizer, "Flickr wrapper," 2009. [Online]. Available: <http://wifo5-03.informatik.uni-mannheim.de/flickrwrapper/>. [Accessed: 11-May-2012].
- [111] MetaBrainz, "MusicBrainz - The Open Music Encyclopedia," 2003. [Online]. Available: <http://musicbrainz.org/>. [Accessed: 15-Aug-2012].
- [112] Jamendo, "Jamendo - The #1 platform for free music," 2006. [Online]. Available: <http://www.jamendo.com/en/>. [Accessed: 25-Sep-2012].
- [113] OFX, "Open Financial Exchange," 2007. [Online]. Available: <http://www.ofx.net/>. [Accessed: 15-Aug-2013].
- [114] W3C, "SPARQL Query Language for RDF," 2008. [Online]. Available: <http://www.w3.org/TR/rdf-sparql-query/>. [Accessed: 11-Aug-2012].
- [115] A. Alowisheq, D. E. Millard, T. Tiropanis, and A. M. Alowisheq David Tiropanis, Thanassis, "EXPRESS: EXPressing REStful Semantic Services Using Domain Ontologies," in *8th International Semantic Web Conference (ISWC 2009)*, vol. 5823, VA, USA: Springer Berlin Heidelberg, 2009, pp. 941–948.
- [116] S. Speiser and A. Harth, "Integrating Linked Data and Services with Linked Data Services," in *The Semantic Web: Research and Applications SE - 12*, vol. 6643, G. Antoniou, M. Grobelnik, E. Simperl, B. Parsia, D. Plexousakis, P. Leenheer, and J. Pan, Eds. Springer Berlin Heidelberg, 2011, pp. 170–184.
- [117] S. Speiser and A. Harth, "Towards Linked Data Services," in *International Semantic Web Conference (ISWC 2010)*, 2010, pp. 5–8.

- [118] B. Norton and R. Krummenacher, "Consuming Dynamic Linked Data," in *1st International Workshop on Consuming Linked Data (COLD2010)*, 2010, pp. 1–12.
- [119] R. Verborgh, T. Steiner, D. Van Deursen, S. Coppens, E. Mannens, R. Van De Walle, and J. G. Vallés, "RESTdesc — A Functionality-Centered Approach to Semantic Service Description and Composition," in *9th Extended Semantic Web Conference*, 2012, pp. 4–5.
- [120] S. Crites, F. Hsu, and H. Chen, "OMash: Enabling Secure Web Mashups via Object Abstractions," in *Proceedings of the 15th ACM conference on Computer and communications security - CCS '08*, 2008, p. 99.
- [121] R. Hashimoto, N. Ueno, and M. Shimomura, "A design of usable and secure access-control APIs for mashup applications," in *Proceedings of the 5th ACM workshop on Digital identity management - DIM '09*, 2009, p. 31.
- [122] C. Yanchun and W. Xingpeng, "A Security Risk Evaluation Model for Mashup Application," *2009 Int. Conf. Inf. Manag. Innov. Manag. Ind. Eng.*, pp. 212–215, 2009.
- [123] Edgar Weippl, "Secure 2.0 - securing the information sharing on web 2.0," *Austrian FIT-IT project*. [Online]. Available: <http://www.ifs.tuwien.ac.at/node/6570>. [Accessed: 22-Sep-2012].
- [124] A. Anjomshoaa, K. Vo-Sao, A. Tahamtan, A. M. Tjoa, and E. Weippl, "Self-monitoring in social networks," *International Journal of Intelligent Information and Database Systems*, vol. 6, no. 4. p. 363, 2012.
- [125] B. Unhelkar, A. Ghanbary, and H. Younessi, *Collaborative Business Process Engineering and Global Organizations*. IGI Global, 2009, p. 383.
- [126] CRA and ACM, "Leading cyber security experts identify key research challenges," *Press Release: Leading Cyber Security Experts Identify Key Research Challenges*, 2003. [Online]. Available: <http://archive.cra.org/Activities/grand.challenges/security/press.release.html>.
- [127] M. McCormick, "New Privacy Legislation," 2003. [Online]. Available: <http://www.ica.bc.ca/kb.php3?pageid=2326>.
- [128] O. Sacco and A. Passant, "A Privacy Preference Ontology ( PPO ) for Linked Data," in *Linked Data on the Web Workshop at WWW2011*, 2011.
- [129] C. Fellbaum, "WordNet and wordnets," in *Encyclopedia of Language and Linguistics*, 2005, Second Edition., pp. 665–670.

- [130] C. Leacock and M. Chodorow, "Combining Local Context and WordNet Similarity for Word Sense Identification," *An Electron. Lex. Database*, pp. 265–283, 1998.
- [131] S. Banerjee and T. Pedersen, "Extended gloss overlaps as a measure of semantic relatedness," in *Proceedings of the 18th international joint conference on Artificial intelligence (IJCAI'03)*, 2003.
- [132] A. Anjomshoaa, K. Vo-Sao, A. M. Tjoa, E. Weippl, and M. Hollauf, "Context Oriented Analysis of Web 2.0 Social Network Contents MindMeister Use-case," *The 2nd Asian Conference on Intelligent Information and Database Systems*. Springer LNCS/LNAI, Hue City, Vietnam, 2010.
- [133] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, no. 1–3, pp. 1–6, 1998.
- [134] W. Weaver, "Translation," in *Machine Translation of Languages*, Reprinted ., New York: John Wiley & Sons, 1949, pp. 15–23.
- [135] J. C. Mallery, "Thinking about foreign policy: Finding an appropriate role for artificial intelligence computers," in *Master's thesis, M.I.T. Political Science Department*, 1988.
- [136] P. University, "WordNet - A lexical database for English," 2005. [Online]. Available: <http://wordnet.princeton.edu/>.
- [137] M. Lesk, "Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone," in *SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation*, 1986, pp. 24–26.
- [138] A. Kilgariff and J. Rosenzweig, "Framework and Results for English SENSEVAL," *Comput. Hum.*, vol. 34, no. 1–2, pp. 15–48, 2000.
- [139] Z. Wu and M. Palmer, "Verb semantics and lexical selection," in *32nd. Annual Meeting of the Association for Computational Linguistics*, 1994, pp. 133–138.
- [140] G. Developers, "Google Custom Search API," 2011. [Online]. Available: <https://developers.google.com/custom-search>. [Accessed: 22-Sep-2012].
- [141] H. Chen, C. Schuffels, and R. Orwig, "Internet Categorization and Search: A Self-Organizing Approach," *J. Vis. Commun. Image Represent.*, vol. 7, no. 1, pp. 88–102, Mar. 1996.
- [142] R. Mayer and A. Rauber., "SOMToolbox," 2011. [Online]. Available: <http://www.ifs.tuwien.ac.at/dm/somtoolbox/>. [Accessed: 20-Nov-2012].

- [143] A. Malki and S. M. Benslimane, "Building Semantic Mashup," in *Proceedings ICWIT 2012*, 2012, pp. 40–49.
- [144] B. Endres-Niggemeyer, *Semantic Mashups Intelligent Reuse of Web Resources*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.
- [145] M. Jarrar, "A Query Formulation Language for the Data Web," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 783–798, Feb. 2012.
- [146] M. Sabbouh, J. Higginson, S. Semy, and D. Gagne, "Web mashup scripting language," in *Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 1305–1306.
- [147] Adobe, "Adobe Flex - open source framework," 2004. [Online]. Available: <http://www.adobe.com/products/flex.html>. [Accessed: 25-Sep-2012].
- [148] MeisterLabs, "Mind Mapping Software - Create Mind Maps online," 2008. [Online]. Available: <http://www.mindmeister.com/>. [Accessed: 12-Aug-2012].
- [149] Facebook, "Statistic," 2011. [Online]. Available: <http://www.facebook.com/press/info.php?statistics>. [Accessed: 20-Nov-2012].
- [150] Facebook, "Facebook Graph API," 2011. [Online]. Available: <http://developers.facebook.com/docs/reference/api/>. [Accessed: 20-Nov-2012].
- [151] Freebase, "Freebase API — Google Developers," 2010. [Online]. Available: <https://developers.google.com/freebase/>. [Accessed: 22-Sep-2012].
- [152] Freebase, "Freebase Schema," 2010. [Online]. Available: <http://www.freebase.com/schema>.
- [153] Freebase, "Freebase Developers," 2010. [Online]. Available: <http://wiki.freebase.com/wiki/Developers>. [Accessed: 22-Sep-2012].
- [154] Twitter, "The Twitter REST API," 2010. [Online]. Available: <https://dev.twitter.com/docs/api>. [Accessed: 04-Nov-2011].
- [155] S. Aghaee and C. Pautasso, "An Evaluation of Mashup Tools Based on Support for Heterogeneous Mashup Components," in *Current Trends in Web Engineering*, 2012.
- [156] I. Pahlke, R. Beck, and M. Wolf, "Enterprise Mashup Systems as Platform for Situational Applications," *Bus. Inf. Syst. Eng.*, vol. 2, no. 5, pp. 305–315, Aug. 2010.

- [157] S. S. Minhas, P. Sampaio, and N. Mehandjiev, "A Framework for the Evaluation of Mashup Tools," *2012 IEEE Ninth Int. Conf. Serv. Comput.*, pp. 431–438, Jun. 2012.
- [158] W. Al Sarraj, "A Usability Evaluation Framework for Web Mashup Makers for End-Users," in *Disseratation, Web and Information System Engineering Lab, Department of Computer Science, Faculty of Science & Bio-Engineering Sciences*, Vrije Universiteit Brussel, 2012, p. 200.
- [159] N. Carrier, T. Deutsch, C. Gruber, M. Heid, and L. L. Jarrett, "The business case for enterprise mashups," *Web 2.0 technology solutions, IBM White Paper*, no. August, 2008.
- [160] V. Hoyer, K. Stanoesvka-Slabeva, T. Janner, and C. Schroth, "Enterprise Mashups: Design Principles towards the Long Tail of User Needs," *2008 IEEE Int. Conf. Serv. Comput.*, pp. 601–602, Jul. 2008.
- [161] W. Ketter, M. Banjanin, R. Guikers, A. Kayser, R. E. Westernacher, P. Jibes, B. V. J. B. V, and B. V Jibes, "Introducing an agile method for enterprise mash-up development," in *In Proc. of the IEEE Conf. on Commerce and Enterprise Computing*, 2009.
- [162] V. K. S.-S. Hoyer and K. Stanoevska-Slabeva, "Generic Business Model Types for Enterprise Mashup Intermediaries," in *Value Creation in E-Business Management SE - 1*, vol. 36, M. Nelson, M. Shaw, and T. Strader, Eds. Springer Berlin Heidelberg, 2009, pp. 1–17.