Brigitte Endres-Niggemeyer *Editor*

Semantic Mashups

Intelligent Reuse of Web Resources

Springer

Semantic Mashups

Brigitte Endres-Niggemeyer
Editor

Semantic Mashups

Intelligent Reuse of Web Resources

Springer

Editor
Brigitte Endres-Niggemeyer
Hanover, Germany

ISBN 978-3-642-36402-0 ISBN 978-3-642-36403-7 (eBook)
DOI 10.1007/978-3-642-36403-7
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013936081

ACM Computing Classification (1998): H.3.5, I.2, H.5, D.2

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

At first sight, mashups are easily defined. They integrate existing web resources in order to produce helpful new resources or services. Users are in focus—they are thought to easily create mashups for a current use situation.

Mainly because of their simplicity, their reuse of existing resources, and their user orientation, mashups are so widely distributed on the web.

Mashups are semantic mashups—to different degrees. Some understanding of the incoming resources is a precondition for combining them reasonably. Selective perception is common, but evidence for non-semantic mashups is still missing. Until further notice we assume that the semantic features of mashups are ubiquitous. They are prime mashup properties, convincing by their practical value and more.

Mashups are wide-spread and their proper ordering is notoriously difficult. Koschmider et al.[1] promise to elucidate the mashup hype (sic) distinguishing mashups depending on

- what mashups display:
  - dimension 1: presentation mashups, data mashups, functionality mashups
  - dimension 2: mapping mashups, photo/video mashups, search/shopping mashups, news mashups
- where mashups are put together: server-side mashups, client-side mashups
- how mashups get input: extraction mashups, flow mashups
- mashup users: consumer mashups, business or enterprise mashups

Every real existing mashup is entitled to participate in a choice of these categories, and to add some others that Koschmider et al. do not mention. Mashups mixing client-side and server-side activity are as normal as mashups obtaining content by information extraction from text and picking up video clips. More examples are easy to imagine but not needed.

The chapters of this book render a part of real-life mashup diversity. Their basic organization is simple.

[1] http://mashup.pubs.dbs.uni-leipzig.de/files/paper14%5B1%5D.pdf.

- The two overview chapters of the beginning take readers around in the mashup environment:
  - First mashups are followed through their manifold habitats/ecosystems.
  - The second chapter concentrates on the regulations (standards, guidelines, APIs) that mashups must rely on for integrating web resources of independent producers.
- In the next sequence mashups are traced in their web home stations. The semantic mashups must be explained in more detail, whereas web environments like DBpedia or search engines are familiar to almost all readers. In contrast the Web of Things (WoT) is new and of big impact. Mashups in web contexts are considered in chapters on:
  - DBpedia mashups
  - mashups for web search engines
  - mashups for sensors and the web of things

  Chapters of this group may be particularly attractive for a developer audience.
- With the following properly application-oriented mashups, the lovers of multicolored and specialized mashup domains will get their money's worth. The authors explain mashups on:
  - mathematical knowledge
  - speech
  - emergency crisis management
  - similarity usage
  - traveling
  - in-town surroundings

Especially here, readers may be taken to fields where they run out of prior knowledge. To ease their life, the book ends with a substantial glossary and subject index.

A book covering a wide range of mashups must assemble a group of authors contributing chapters on their own research fields. The authors of this book met during the AI Mashup Challenge. It ran four times, first at the 2009 German AI Conference in Paderborn and the last three times during the Extended Semantic Web Conferences (ESWC) 2010–2012.

Fortunately the AI Mashup Challenge was well supported. Elsevier, Linguatec, O'Reilly, and Addison-Wesley sustainedly sponsored its runs. We gratefully acknowledge their contribution.

For their review of book chapters we thank Felix Burkhardt, Rui Cai, Emanuele Della Valle, Michael Hausenblas, Pascal Hitzler, Krzysztof Janowicz, Paul Librecht, Horacio Saggion, Jevon Wright, and Mao Ye. Special thanks go to Christoph Lange for his assistance in editor tasks.

Lucca, Italy Brigitte Endres-Niggemeyer

Contents

Chapter 1
The Mashup Ecosystem

Brigitte Endres-Niggemeyer

Abstract The web is growing quickly, substructures are coming up: a {social, semantic, etc.} web, or the {business, services, etc.} ecosystem which includes all resources of a specific web habitat. In the mashup ecosystem, developers are in intense scientific activity, what is easily measured by the number of their recent papers. Since mashups inherit an opportunistic (participatory) attitude, a main point of research is enabling users to create situation-specific mashups with little effort. After an overview, the chapter highlights areas of intensive discussion one by one: mashup description and modeling, semantic mashups, media mashups, ubiquitous mashups and end-user related development. Information is organized in two levels: right under the headings, a block of topic-related references may pop up. It is addressed to readers with deeper interest. After that, the text for everybody explains and illustrates innovative approaches. The chapter ends with an almost fail-safe outlook: given the growth of the web, the ecosystem of mashups will keep branching out. Core mashup features such as reuse of resources, user orientation, and versatile coordination (loose coupling) of components will propagate.

1.1 The Mashup Ecosystem

On the Ecosystem: [10, 14, 18, 28, 71, 109, 126, 152, 155, 171, 187, 196, 208, 209, 224, 229, 230, 239, 240, 242]

Mashups are advancing on the internet, the web, and the semantic web. They have no problems to adapt to the cultures in the web [10], performing on the semantic web as on the internet or web in general. Their count is going up. They expand their services into new areas. They take root. Their simple principle of building upon work of others is gaining acceptance. As far as one can see mashups will remain on the move. In [208] Spivack illustrates how he anticipates the web and the semantic web will go on developing (see Fig. 1.1). Corresponding to the fast expansion of the web, people tend to define substructures: a social web, a web of services, a semantic

B. Endres-Niggemeyer (✉)
Heidegrün 36, 30179 Hanover, Germany
e-mail: brigitteen@googlemail.com

B. Endres-Niggemeyer (ed.), *Semantic Mashups*,
DOI 10.1007/978-3-642-36403-7_1,

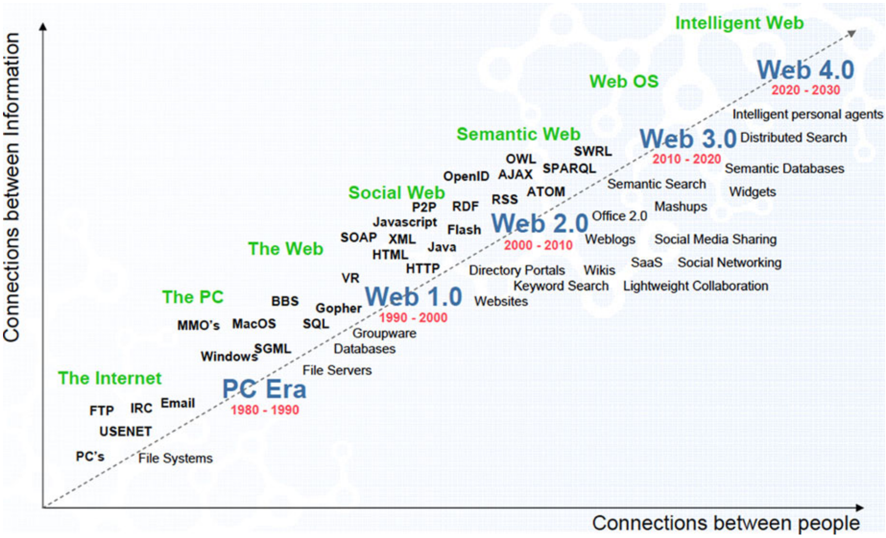


Fig. 1.1 Timeline of internet and web development (from [208]). Notice mashups coming up towards 2010

web, a mobile web, a web of things, and so on. The subwebs overlap as shown in Fig. 1.2. Like smaller geographical or organizational units, let us say the regions of a country, these subwebs partition the web universe, so that local communities can concentrate on the concerns of their own subunit.

Fig. 1.2 Subwebs of the web (from [90])

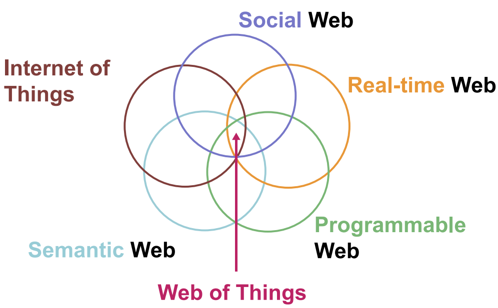
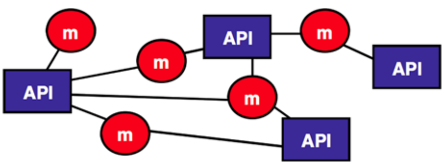


Fig. 1.3 The mashup ecosystem linking mashups and APIs. Mashups in *red circles*, APIs in *blue squares* (from [240])



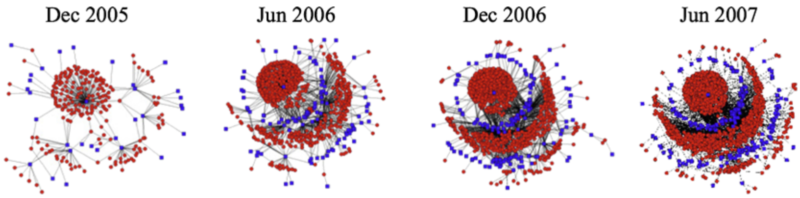


Fig. 1.4 The mashup ecosystem evolution. Mashups in *red*, APIs in *blue* (source: [240]) (Color figure online)

Mashups are a relatively new web concept. Their history begins with DJ mashups of songs and with web 1.0 portals. The oldest mashup on Programmable Web[1] was added in 2005. The mashup ecosystem [240] may be seen as linking mashups and web APIs (see Fig. 1.3). [196] conceives it as configuration of service providers, mashup authors, and users without any central authority. The mashup ecosystem also appears as a specific software ecosystem (details in [28]). Thus the mashup ecosystem integrates mashups and their cohabitants wherever they may be spread on the web. Like a biological ecosystem, it interconnects all species that are needed for its functioning, such as users, tools or script languages. The affinity of mashups to composite web services [71, 171] is evident, so that methods from both sides cross the border without trouble.

The mashup ecosystem is growing quickly. Some evidence available for instant inspection is shown in Fig. 1.4. [229–231] describe a growth model in detail. Success factors for mashups are the activation of end-users as creators/designers, the attractivity of the most popular APIs (all readers will guess right: Google Maps, Twitter, YouTube and so on—more on the ProgrammableWeb hit list[2]), and the simple technique of copying—the reuse of existing resources. The mashup ecosystem shares the innovation rate of the web and its service ecosystem (also called internet/web of services—more detailed description in [14, 187]). [126, 155] explain the computational marketplace ecosystem. It serves mashups, too—why should mashups pick up their APIs anywhere on the web instead of going straight to the service market for shopping?

As the whole service ecosystem, the mashup ecosystem is assumed to follow a pattern of open innovation (Fig. 1.5), branching out, advancing into new domains, reaching more developers and users, and so on.

In the following we inspect the mashup ecosystem where the discussion is particularly active and innovative: mashup description and modeling, semantic mashups, media mashups, ubiquitous mashups, and end-user related development.

[1] http://www.programmableweb.com.

[2] http://www.programmableweb.com/apis/directory/1?sort=mashups.

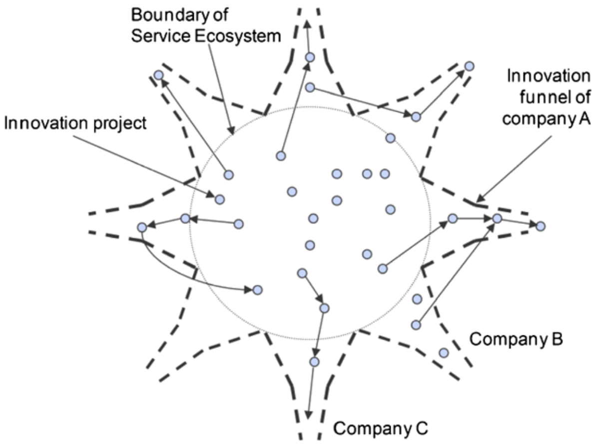


Fig. 1.5 Innovation expanding from a service ecosystem (source: [187])

1.2 Mashup Description and Modeling

On Model Descriptions: [8, 47, 59, 61, 62, 66, 69, 78, 80, 96, 101, 105, 112–114, 117, 151, 164, 165, 177, 179, 190, 191, 221]

Mashups came later and as lightweight web applications into an environment where enterprise WSDL/SOAP web services with their more elaborated scheme were already established. In particular for enterprise mashups in intranets, the standards of earlier web services were and are kept up, while consumer mashups are being watched less for WSDL/SOAP compliance. Possibly mashups may, however, relax the climate for enterprise services. [79] states that

> enterprise mashups must realize the benefits already touted by end-user mashups.

This would summon earlier monoliths to adapt to the more flexible and abstract mashup concept.

Much effort is observed in modeling and description of mashups. Many mashup developers pursue the functional standards of the SOA-based web services habitat. In parallel, enterprise services and mashups begin to exploit web features such as semantic annotation, so that both parties are sharing more common ground.

A choice from the competing modeling and description activities on the market:

- Web Mashup Scripting Language (WMSL—[190])
- Enterprise Mashup Markup Language (EMML)[3] of the Open Mashup Alliance (OMA)
- Mashup Component Description Language (MCDL—[78])
- Universal model of components and composition [62]
- Universal model based on MetaObject Facility (MOF)[4] standards [180]

[3] http://www.openmashup.org/omadocs/v1.0/index.html.

[4] http://www.omg.org/mof/.

- UML2 model for a set of integrated mashups [80]
- ResEval Mash [113, 114] with a domain-specific description language (DSL)

The first and the last approach are chosen for closer inspection:

- The WMSL AM-AO use case because of its OWL ontology alignment of web services
- ResEval because of its two-level model with an abstract and a domain-related layer and the requirement-driven interface

From the mashup quality models [185], PEUDOM [39] is selected for a more detailed description.

Context awareness and personalization are main modeling issues as well. As they mostly happen in a ubiquitous environment, they will be dealt with there.

1.2.1 AM-AO: Web Mashup Scripting with OWL Ontology Use

Imagine that AM (Air Mobility) and AO (Air Operations) cooperate. The AM system is responsible for missions like mid-air refueling and the movement of vehicles while the AO system is primarily concerned with offensive and defensive missions [192]. Each party has an ontology of its own [78, 190, 191].

A Web Mashup Scripting Language (WMSL) script permits end-users to combine AM and AO services. WMSL uses both its own script language and standard

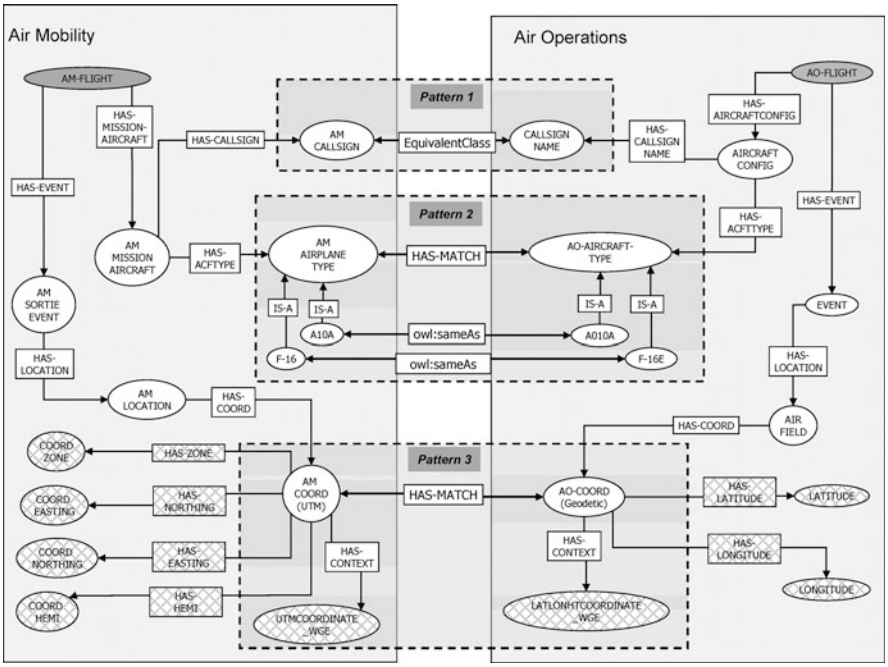


Fig. 1.6 Alignment of diverging OWL ontologies of Air Mobility and Air Operations (source: [192])

HTML commands/tags. The scripts deal with input of resources (WSDL files, schemas, ontologies, and WSML scripts), with the alignment of concepts, and with workflow.

WMSL embeds mapping relations in HTML. Look at the encoding for a concept alignment (compare pattern 1 in Fig. 1.6) in the AM and AO ontology:

```
<dl class = "owl-equivalentClass">
<dt> <a href= "http://mitre.org/owl/1.1/AM#CallSign">
AM#CallSign</a><dt>
<dt> <a href= "http://mitre.org/owl/1.1/AO#CallSignName">
AO#CallSignName</a><dt>
<dl>
```

In Fig. 1.6 the OWL ontologies of Air Mobility (AM) and Air Operations (AO) are reconciled by three mediating patterns. Pattern 1 uses the simple equivalence of two concepts with different names whereas the match in pattern 2 depends on the 'owl:sameAs' identity of subconcepts on both sides.

1.2.2 Domain-Specific Description and Modeling: ResEval Mash

While most mashup tools are domain-independent, the ResEval Mash [113, 114] is dedicated to a specific task with an own body of knowledge: research evaluation. The authors combine a generic mashup meta-model with a domain-specific description language (DSL) as a sublanguage. The DSL specifies a class of mashups, in the present case for research evaluation, using terms specified in cooperation with domain experts. The more abstract generic mashup meta-model is addressed by IT developers, e.g. for entering new components, whereas a graphical user interface with a visual DSL (Fig. 1.7) helps domain experts to set up their mashups for concrete tasks.

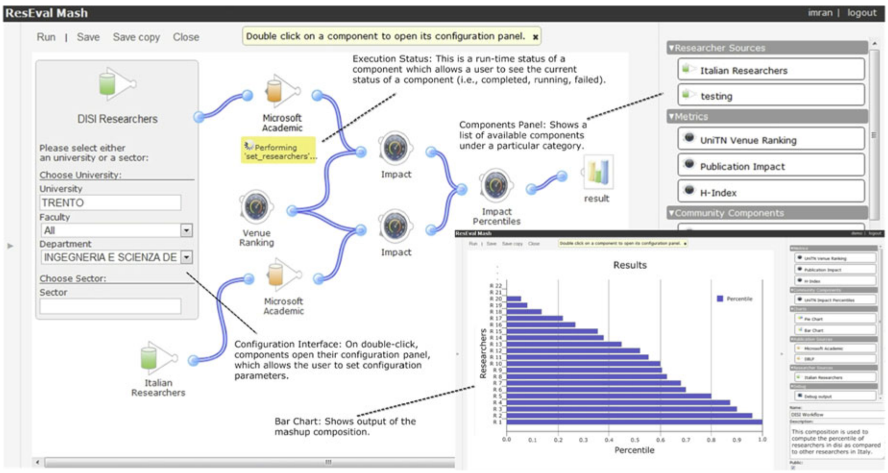


Fig. 1.7 ResEval Mash: The user interface (from [113])

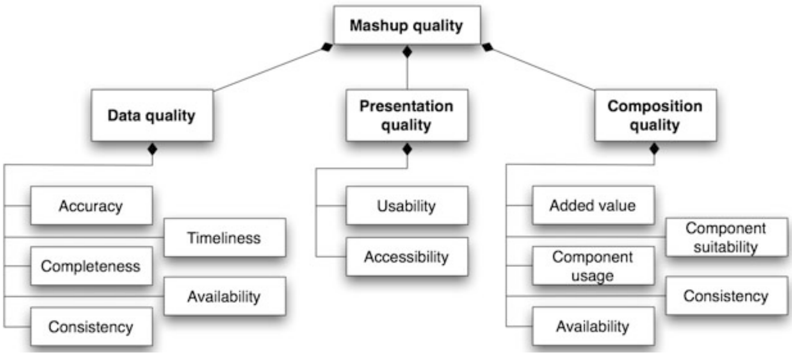


Fig. 1.8 Mashup quality model (from [38])

```
<component id="googlemaps" name="GoogleMaps"
        description="Geolocalization service.">

        <operation name="centerMap" description="."
                similarity-meaning="CenterMap"
                similarity-verb="center" similarity-object="map">

                <param name="latitude" description="Latitude"
                        direction="input" type="xsd:decimal"
                        similarity-meaning="Geography.owl#Latitude"/>
                <param name="longitude" description="Longitude"
                        direction="input" type="xsd:decimal"
                        similarity-meaning="Geography.owl#Longitude"/>
        </operation>
        ....
        <event name="positionOnClick" description="..."
                similarity-meaning="ProvidePosition"
                similarity-verb="provide" similarity-object="position">

                <param name="latitude" description="Latitude"
                        direction="input" type="xsd:decimal"
                        similarity-meaning="Geography.owl#Latitude"/>
                <param name="longitude" description="Longitude"
                        direction="input" type="xsd:decimal"
                        similarity-meaning="Geography.owl#Longitude"/>
        </event>
        ....
```

```
<qualityAttributes>
        <reputation>1</reputation>
        <languages>
                <language>javascript</language>
        </languages>
        <dataFormats>
                <dataFormat>JSON</dataFormat>
                <dataFormat>XML</dataFormat>
                <dataFormat>VML</dataFormat>
                <dataFormat>KML</dataFormat>
        </dataFormats>
        <security>no authentication</security>
        <timeliness>0.9</timeliness>
        <accuracy>0.9</accuracy>
        <completeness>0.8</completeness>
        <availability>1</availability>
        ...
</qualityAttributes>
```

basic quality features

Fig. 1.9 Google Maps description: Code sample for event-based handling on the *left* and quality attributes on the *right* (source: [38])

1.2.3 Mashup Quality—The PEUDOM Mashup Tool

On Quality: [2, 12, 23, 36–38, 40, 45, 176, 185, 247, 249, 255]

Mashup content largely decides on mashup quality, so that external resources have a big impact on it. The rest of a mashup's quality results from good component integration and a well-designed visualization interface. The quality assessment of a mashup as a whole is puzzled together from the quality scores of its parts, so that it is complex enough for an explicit quality description or model.

The mashup quality model [38] displayed in Fig. 1.8 organizes its features in three dimensions: data of the components, the presentation on the user interface, and

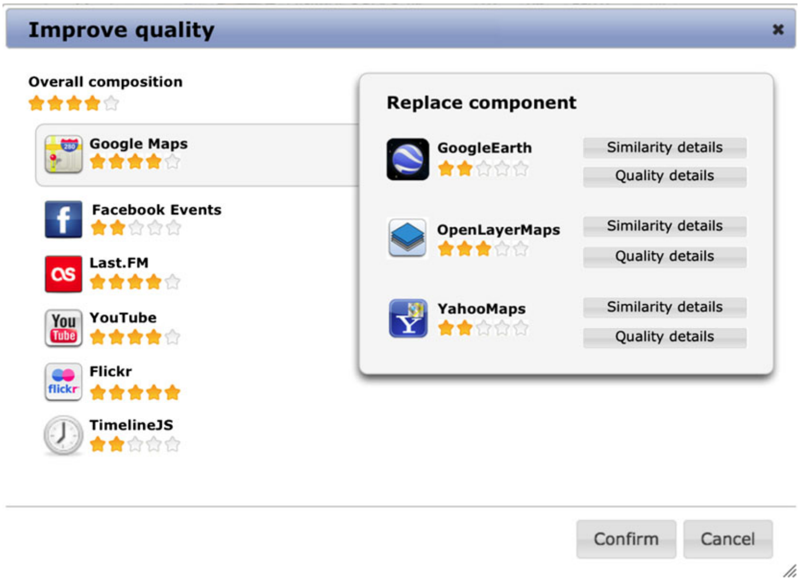


Fig. 1.10 Google Maps with alternatives on the PEUDOM user interface, the alternatives being ranked by quality (source: [38])

the composition quality. Though quality criteria are context-dependent, the quality features for incoming API data are widely shared. This is because accuracy, timeliness, completeness, availability, and consistency are crucial for all mashups that reuse web services. In case of input problems, like missing data delivery from a chosen API, the mashup has to react, e.g. by switching to a substitute resource.

For quality assessment, component descriptions must be instantly available from a repertory [38, 185]. Figure 1.9 displays a sample component description in XML format for Google Maps, the most popular API of the web. It reappears on the user interface of the PEUDOM mashup tool (Fig. 1.10) as the first option with a set of possible replacements. The alternative map services are ranked according to their quality features (cf. left column and content of the green box in Fig. 1.9). The PEUDOM ranking mechanism combines several probabilistic technologies.

1.3 Semantic Mashups

On Semantics: [4, 9, 19, 20, 24, 25, 46, 48–50, 74, 78, 87, 88, 115, 116, 119, 131, 134, 137, 138, 143–145, 148–150, 157, 159, 160, 166, 172–174, 190, 193, 198, 200, 201, 222, 232]

Semantic mashups are at home in the semantic web, although they may also reach outside resources. As regular inhabitants they share the common semantic features of the semantic-web ecosystem. If one conceives the semantic web as being characterized by semantic annotations (markup or metadata) coded in RDF or OWL, the background of basic semantic mashups is all set.

By using semantic annotations, neutral mashups permute into semantic mashups. Where metadata, e.g. from an ontology, state which items web services offer (see Fig. 1.6 above), semantic mashups indeed improve the possibilities to choose and match the right input items [157, 198]. Semantic mashups with this profile convince many netizens. W3C provides SAWSDL[5] for semantic annotation of WSDL components via ontology referencing.

The restricted view on mashups explained a moment ago is having its defenders (e.g., [150]). A mashup can be semantic to different degrees. Lightweight semantics is well known in the semantic web.

The further reaching claim about the semantic web is that it achieves a deeper understanding of meaning than other ecosystems would enable. To speak the truth, some penetration into the meaning of content occurs almost everywhere in the web, albeit it may be very limited. Thus the semantic web only emphasizes a feature that was and is widely distributed, only we see that in the semantic web, meaning and semantics score much higher.

Now the illustration and application to semantic mashups:

- Meaning is handled almost in all web applications, but to different degrees. All mashups that deal with symbolic data, from interpreted fact databases to virtual reality, are assumed to be semantic unless they prove the contrary. Developers who feel to have a semantic-free mashup are invited to present it. Until this happens, one can put the non-semantics issue aside.
- Treating meaning is by no means restricted to markup, annotation, and metadata. Take information extraction as an example. It may use metadata, but just as well syntactic or semantic templates. Or look at mashups interpreting data via semantic rules or probabilistic methods (inspect the Black Swan below). Semantic mashups are semantic because they apply semantic methods—all available ones. Semantic mashups can contribute much more semantics than an alignment of data sources via metadata.

[119] presents an example. The authors innovate the classical RDF triple-store-based bookshop scenario (see below) with more internal intelligence. Mashup knowledge is stored in an ontology, Pellet[6] is used for reasoning.

WSDL/SOAP-oriented (semantic) web services are said to have not been as popular as expected. Probably mashups in their ecosystem do not fare better. [172] state the point and anticipate a new wave of services: linked services, mostly coded in RDF. As you may think, the more recent RDF data are still less established in the mashup ecosystem. By today (02-08-2012) only 65 APIs on ProgrammableWeb are of RDF format, the oldest ones from 2006. All DBpedia mashups fit on one page.

[5] http://www.w3.org/TR/sawsdl/.

[6] http://clarkparsia.com/pellet/.

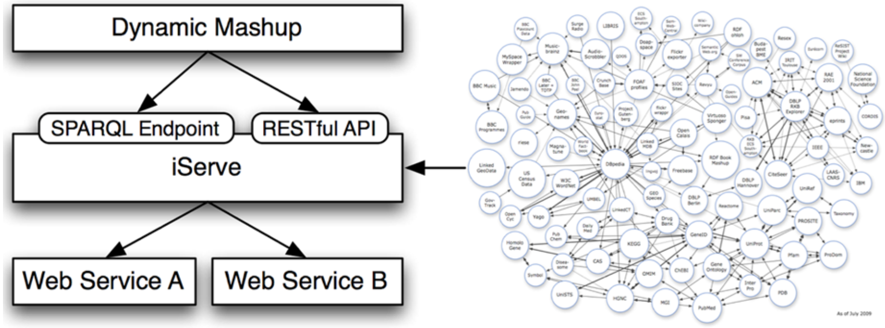


Fig. 1.11 The dynamic mashup using iServe support (source: [143])

Work in the linked data mashups ecosystem is illustrated by:

- a dynamic mashups technology with iServe repository [143]
- the classic RDF book mashup [24]
- Semantic Web Pipes (SWP—[137, 138])
- the Black Swan mashup[7] for interpretation of rare events [145]
- the FlyBase Insitus mashup [159] for the Drosophila genome

1.3.1 Dynamic Mashups with iServe Support

While traditional mashups tend to be static, dynamic mashups supported by a linked data server (called iServe [143, 174]) can select suitable resources during runtime. The mashups use linked data, but also other REST-based web services. iServe disposes of a repertory filled with annotated services. When a mashup requires a service, iServe looks it up in its directory, and in case of problems replaces it with a better equivalent. Thus the mashup delegates resource invocation and gains flexibility (see Fig. 1.11).

1.3.2 RDF Book Mashup

In their classical RDF Book Mashup, the authors of [24] demonstrate how a mashup works in an RDF environment. Books and authors have URIs with an RDF description. A SPARQL query engine handles the search inside the RDF triple pool. The RDF descriptions contain outbound web links, in the present case to Amazon and Google APIs (see Fig. 1.12). They fill the local query result with additional

---

[7] http://blackswanevents.org/.

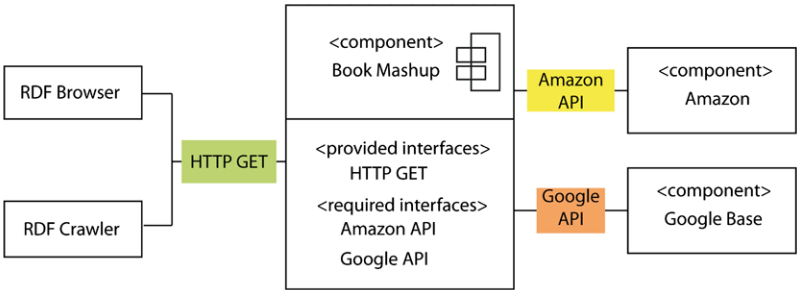


Fig. 1.12 RDF book mashup structure, remake (source: [24])

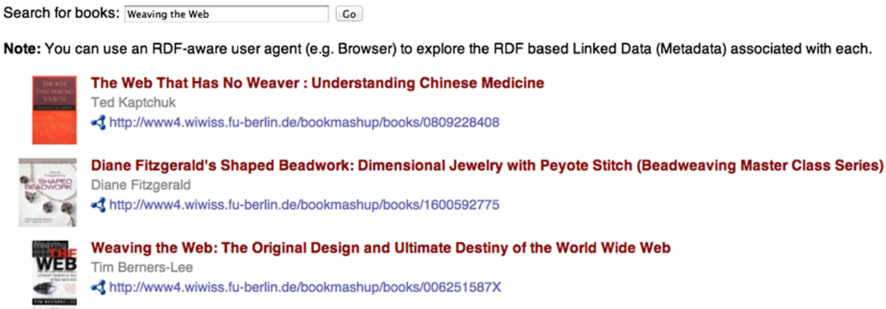


Fig. 1.13 RDF book search results

data. A short PHP script manages the HTTP communication. An output snippet (Fig. 1.13) shows data imported from Amazon.

1.3.3 Semantic Web Pipes (SWP)

Yahoo Pipes[8] are the most popular tool for end-user mashup development. With aggregated pipes (processes accepting inputs and delivering an output) users define their mashups on a graphical interface. [137, 138, 160] reconstruct the pipes approach for a semantic-web environment. The editor/graphical user interface is maintained in the well-known pipes style, but the operators change. Consider the fetches in Fig. 1.14: instead of the Yahoo-own 'Fetch CSV' or 'Fetch Feed', Semantic Web Pipes offers 'RDF Fetch', 'HTML Fetch', 'HTTP GET', 'Sparql Result Fetch', and so on. RDF, XML, Microformats, JSON, and binary streams are accepted. Pipes can be entered into other pipes. As with Yahoo Pipes, users can store and publish their pipes.

[8] http://pipes.yahoo.com/pipes/.

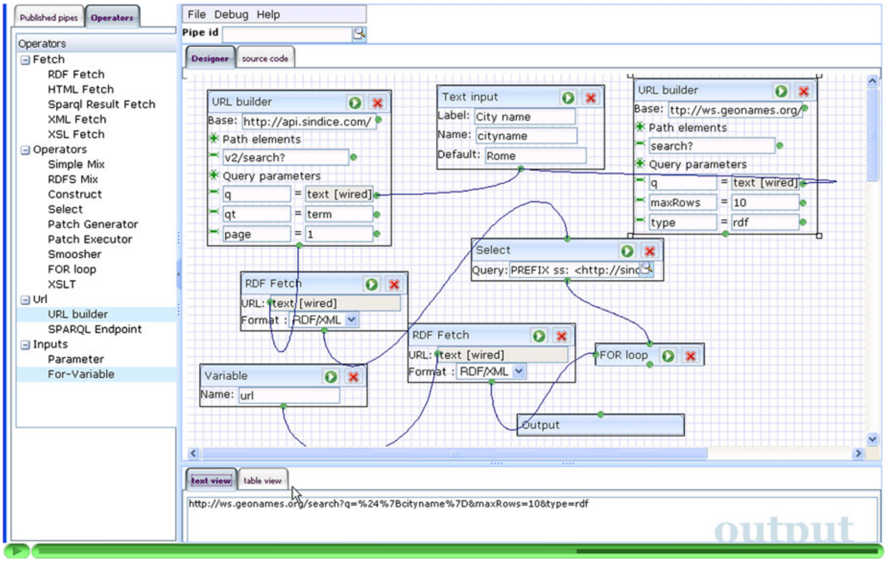


Fig. 1.14 SWS mashup development (from http://pipes.deri.org/cityfacts.html)

1.3.4 Black Swan—Discovering Events That Matter

Black swan events are as rare as black swans, but they exist.

The Black Swan mashup [145] enables users to explore timelines of statistical data. The timelines are decorated with factual event knowledge that may help to interpret the data. One may for instance ask whether a war outbreak influences the income per capita. Black Swan uses event data from DBpedia, Freebase, NOAA, Correlates of War, EM-DAT and BBC Timeline. Locations are imported

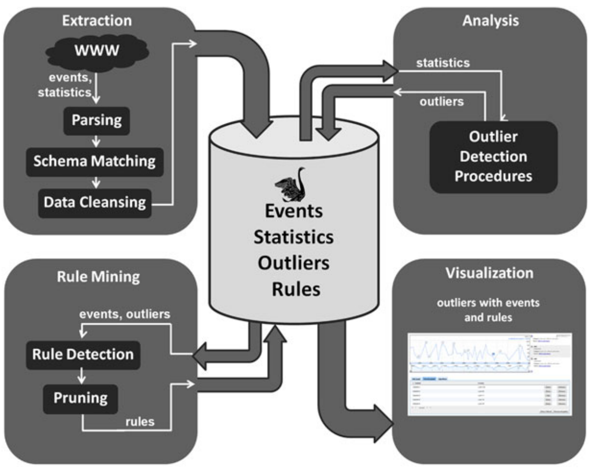


Fig. 1.15 Black Swan. The architecture overview (from http://blackswanevents.org/?page_id=179)

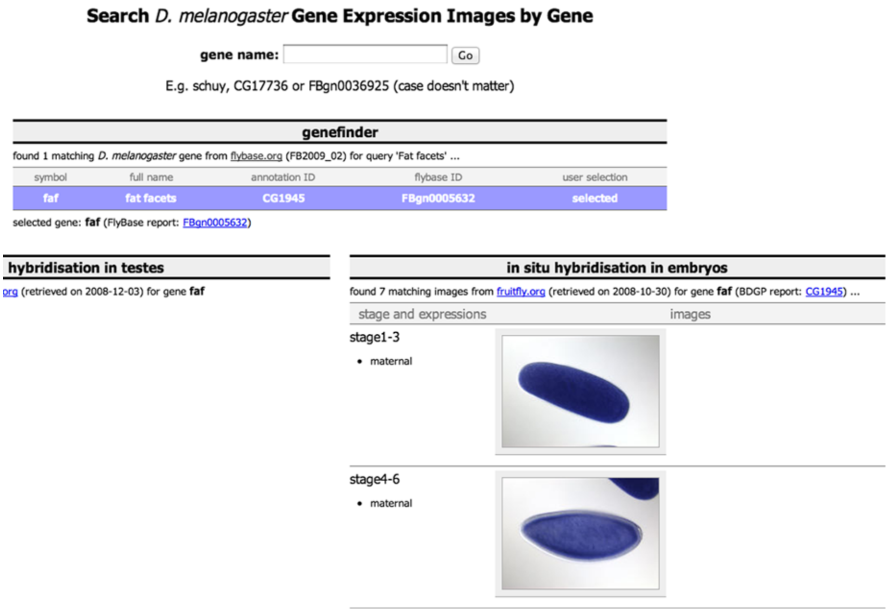


Fig. 1.16 Insitus mashup from FlyBase—abridged, many images cut off

from GeoNames, statistics from GapMinder.[9] Black Swan exemplifies an intelligent semantic mashup because it heavily reworks resources before visualizing them for a user. Methods include information extraction, geo-localizing of data, rule mining, and several regression techniques (cf. Fig. 1.15).

1.3.5 FlyBase Insitus Mashup

The FlyBase[10] contains genetic, genomic and functional data of the fruit fly (Drosophila Melanogaster). Drosophila Melanogaster is common, has a very fast generation alternation and a well-explored genome. The FlyBase collects the genome data from the research literature and other external sources/databases.

Insitus[11] [159] is a mashup integrating data from the Berkeley Drosophila Genome Project (BDGP) and from the Drosophila Testis Gene Expression Database (Fly-TED). It references the FlyBase for disambiguation of gene names. As shown for the fat facets gene in Fig. 1.16, it depicts expressions of genes in different states in embryos and testes. The implementation uses AJAX and SPARQL end-points.

[9] http://www.gapminder.org/.

[10] http://flybase.org/.

[11] http://openflydata.org/search/insitus.

1.4 Media Mashups

No mashup type outperforms maps/geodata mashups in popularity. Countless mashups visualize Google maps and locate some data on it. Together with GPS and location-based services (LBS) they help to find points of interest (POIs), for instance a restaurant nearby. Google, Yahoo, MapQuest and some others provide their maps for worldwide use. No wonder that in this book, three chapters are using geographical maps. Interested readers switch to Travel Mashups, Urban Mashups and/or Mashups for Emergency Management.

Compared to the popularity of map mashups, other species of the media mashup ecosphere remind one of the above-cited black swans: mashups integrating speech, video or augmented/virtual reality (AR/VR) are uncommon, but they exist.

The following presentation concentrates on the less known:

- Given the presence of map mashups in three book chapters, only crowd-sourced collaborative map mashups are discussed in this section.
- Speech mashups deserve to be mentioned. As a whole chapter in this book treats them, they are only briefly presented.
- An augmented painting and a weather webcam with integrated data from web services exemplify AR mashups. Virtual reality mashups come from Second Life.
- The Virtual Director and a mock-up emotional video mashup illustrate video mashups.

1.4.1 Map Mashups

On Maps: [16, 17, 29, 33, 85, 97, 110, 132, 134, 140, 141, 182, 218]

More often than not, map mashups appear as the prime mashup type [33]. In common practice, users or developers apply a geo-map API and locate some of their own data on it. The general state of map development is reported in [182]. [134] explains how to enrich geoinformatics systems with semantics.

Inside GISs (Geographic Information Systems) map mashups incorporate neo-geography[12] tendencies aimed at placing cartography into the reach of non-professional users and developers. Crowd-sourced map mashups are a fact: the community creates free editable maps in a wiki style [17, 140]. A notable example is OpenStreetMap.[13] Google Map Maker[14] follows the approach on the commercial side.

[12] http://en.wikipedia.org/wiki/Neogeography.

[13] http://wiki.openstreetmap.org/wiki/Main_Page.

[14] http://www.google.com/mapmaker.

In MapTube[15] of [17] a Java program called GMapCreator accepts an ESRI[16] (Environmental Systems Research Institute) shape file from Google maps. It generates an overlay of user-own data and puts this layer onto the Google map. GMapCreator displays the result as a Google Map in a web page format.

3D maps are available from Google Earth, Bing Maps, and others. In Google Maps, the 3D view is an integrated function. User additions are possible.

1.4.2 Speech Mashups

On Speech: [68]

Speech mashups use web voice services. Multilingual voice-as-a-service (VaaS) options are available worldwide for all networked devices. Most popular are smartphone and tablet applications. Speech mashups are a core tool for assisting disabled web users.[17]

Web voice services perform speech recognition and text-to-speech processing on the provider's web server. A web client uses the services, so that it listens and talks without storing the (huge) databases that encode the voices. All voices for all languages in the provider's repertory are available.

The speech mashups chapter of the book explains the details as seen in the pioneering AT&T environment. If you want, have a look at a speech mashup iPad app[18] of the author. It supports writers during document revision with an Acapela[19] voice service.

1.4.3 Augmented Reality

On Augmented Reality/Virtual Reality: [17, 72, 73, 82, 84, 133, 237, 238]

In augmented reality (AR) the image before the eyes of a user is enhanced with a computer-generated virtual part. A good example is disentangling the messed cables in a machine. It helps when an AR image presents the user with the target bundling of the cables and the step-by-step procedure of how to reach it.

1.4.3.1 CAMAR—An Augmented Real-World Object

Consider an AR mashup that adds virtual information to a real-world painting in the user's environment. [238] calls this combination of a real object and related virtual

[15]http://www.maptube.org/.

[16]http://www.esri.com.

[17]http://www.research.att.com/projects/AssistiveTechnology/.

[18]https://sites.google.com/site/nospeech3/.

[19]http://www.acapela-vaas.com.

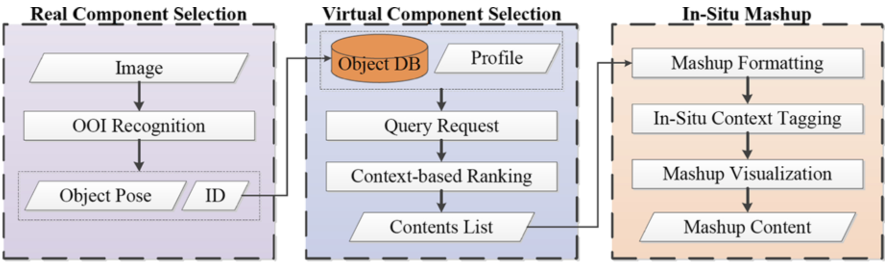


Fig. 1.17 Scheme of mashup with real and virtual component (from [238])

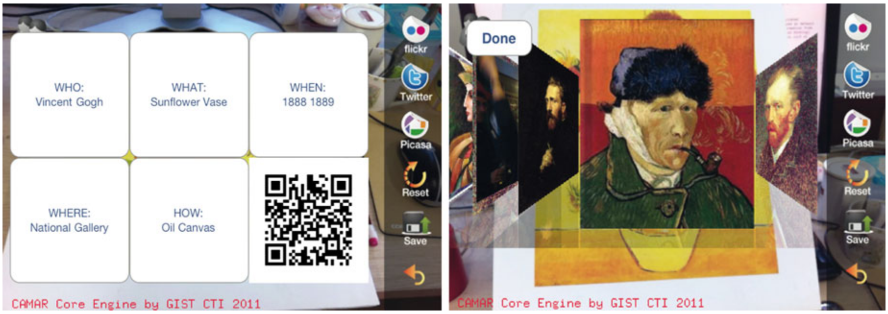


Fig. 1.18 Bar code identification of the object of interest and augmented reality output (from [238])

content in-situ mashup:

> In-situ AR mashup is seamlessly combining additional contextual information to a real-world object to enrich content in one or more senses, where mashup process and its outcome are enhanced with context awareness and visualized with augmented reality for intuitive UI/UX.[20]

In the architecture sketch in Fig. 1.17 the mashup receives a bar code from the user's smartphone camera, identifies the current object of interest (OOI) in the real world (e.g. the painting) and superimposes annotations from a database, so that the mashup components come in part from the real world, in part from an object database. All information is visualized (see Fig. 1.18).

1.4.3.2 Augmented Reality Weather Cam

[82] augments the image of a weather webcam on a building of the University of Münster. As Fig. 1.19 shows, the authors stack data layers from web services onto the basic layer of the image. For the bottom layer image, the camera is turned into

[20] UI: user interface, UX: user experience, more in [102].

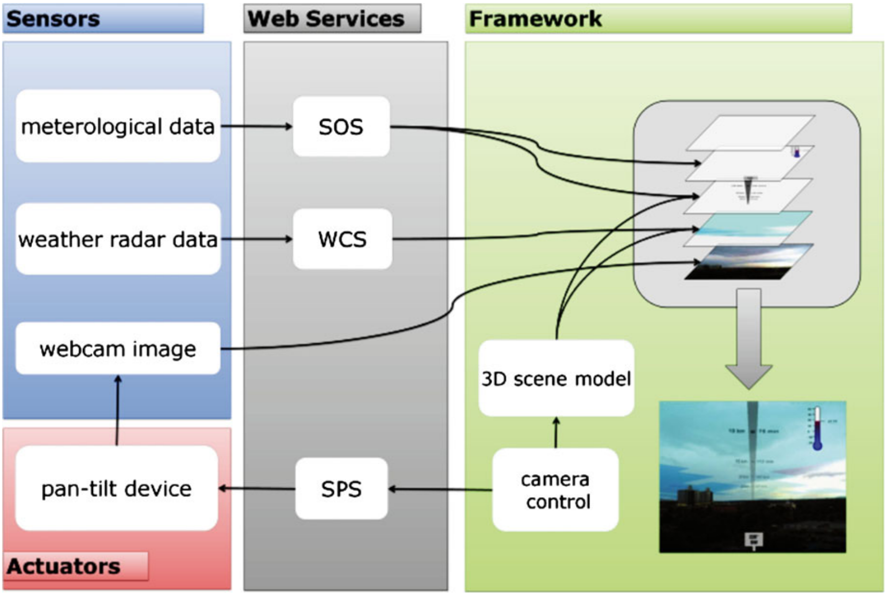


Fig. 1.19 Weather webcam: Standardized sensor data added layer-wise onto the webcam image (from [82])

the wind direction with data from the Sensor Observation Service (SOS) and Sensor Planning Service (SPS). On the second layer the weather radar data are presented with the help of the Web Coverage Service (WCS). The third layer shows a combined scale with physical and temporal distance for weather/rain clouds to come in. On layer four, current temperature and wind speed are displayed. Textual data are added on layer five.

1.4.3.3 Virtual Reality: Mashups in Second Life

In virtual reality (VR) the whole interaction takes place in a computer-generated virtual world. [73] discusses mashups and their semantics in the virtual reality of Second Life.[21] Who wants to develop a Second Life mashup is invited to inspect the API.[22] ProgrammableWeb lists a few Second Life mashups.[23] [17] illustrates their maps in Second Life. Other mashups enter specific APIs like delicious or Flickr into Second Life. The Planespotting[24] mashup of Google Earth and Second Life tracks

[21] http://secondlife.com/.

[22] http://wiki.secondlife.com/wiki/Linden_Lab_Official:Map_API.

[23] http://www.programmableweb.com/api/secondlife/mashups.

[24] http://nwn.blogs.com/nwn/2007/09/planespotting-g.html.

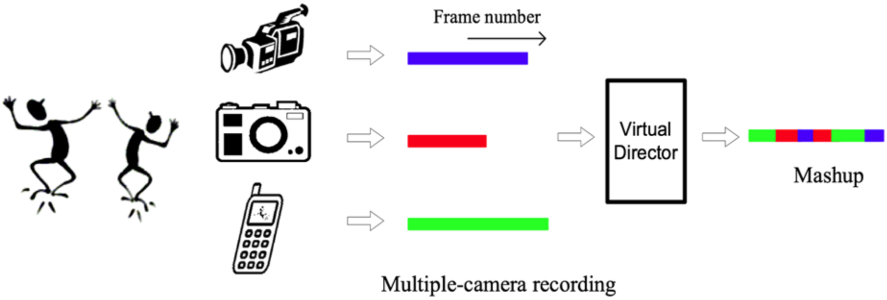


Fig. 1.20 The Virtual Director integrates multiple camera recordings into a video mashup (from [203])

planes at LAX (Los Angeles) airport. An up-to-date mashup makes the iKnow app[25] available inside Second Life, so that speakers of Japanese can improve their English vocabulary in virtual reality.

1.4.4 Video Mashups

On Video: [5, 34, 42, 162, 202, 203, 249]

A video mashup is the result of combining multiple audiovisual sources. It is a product with its own identity, so that its meaning/semantics can widely deviate from the content of the source videos.

1.4.4.1 Virtual Director: Mashup of Multiple Event Recordings

[203] integrates multi-cam recordings of events into a video mashup. Figure 1.20 illustrates the situation. Input movies may come from all sorts of devices and resources (camera, webcam, web resource, smartphone, etc.) with different technologies, so that they differ in angle, duration, frame rate, part of the event covered, sound quality, image quality (e.g. shaking, dizziness, lighting, etc.). A mashup choosing the best frames of all movies reaches a better quality and avoids the boring effect of one persistent camera and view angle. With a suitable tool, an end-user can generate such a video mashup. A professional creator can do the same with a higher artistic endeavor. The resulting mashup is a single video stream as known from the early music and video mashups. The Virtual Director follows user requirements collected in an explorative study with 18 video camera users. The users asked for: synchronization, image quality, diversity, tuning to user preferences, suitable point cuts, semantics, suitable segment duration, and completeness.

[25] http://iknow.jp/.

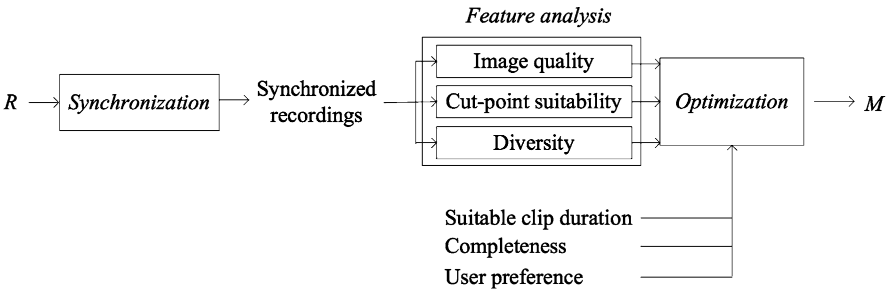


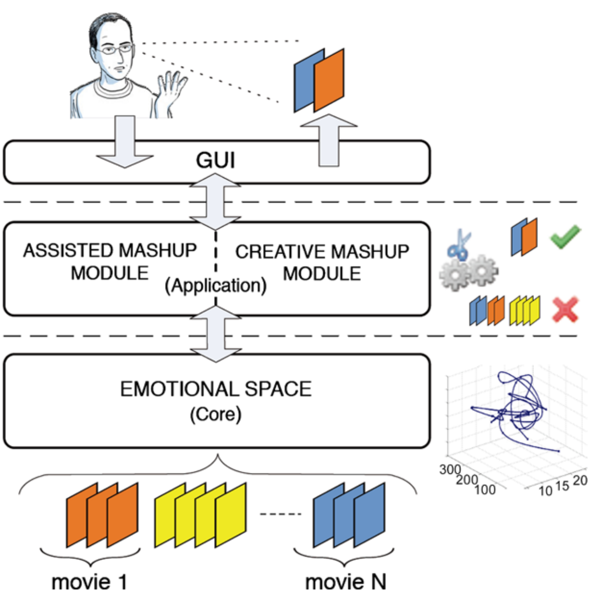
Fig. 1.21 Virtual Director overview of the mashup process. R: recording, M: mashup (from [203])

The overall mashup quality is represented by an objective function that combines the requirements. The segments for a mashup are selected so that the function is maximized. Figure 1.21 shows how the system proceeds.

1.4.4.2 Emotional Video Mashup

In a movie directors interleave the emotional stimulation(s) for the moviegoers. An emotional video mashup [34] (see Fig. 1.22) follows this practice. The authors build their system around an emotional space with the dichotomies warm/cold, dynamic/slow and energetic/minimal on its axes. The emotional development during the movie trajectory is recorded. Mashups are expected to keep the mood while being composed from different sources. Users composing their movie can choose between an assisted mode and a creative mode of mashup building.

Fig. 1.22 Emotional video mashup creation (from [34])



When a user proposes to mix a movie in, the best fit shots are calculated by Euclidean distance in the emotional space. A ranked list of emotionally good shots is offered to the user.

1.5 Ubiquitous Mashups

The ubiquitous web covers the whole web and reaches out into the physical world. It can be seen as a synthesis of the web and ubiquitous (pervasive, ambient intelligence) computing. Ubiquitous mashups may reside on all sorts of networked units—for instance sensors, mobiles, navigation systems, intelligent appliances. The Manhattan Story Mashup of [217] made pervasive devices from phones to large public displays support a big public cooperative story telling event.

Ubiquitous applications may dip into very different contexts. A simple example is an app being projected on a phone, a tablet or a huge TV screen. Or change the user: the mashup happens to be managed in Russian, Italian or Japanese instead of English. Or tune the depth of context penetration: a sensor of a mashup may sit on a lamp post, but think of a heart rhythm monitor under the skin of a patient. Context awareness is a first-order issue with the multitude of situations where ubiquitous applications/mashups may need to accommodate and to perform.

After dealing with context awareness, the report focuses on mobile mashups in different surroundings, sensor mashups, and (embedded) physical mashups.

1.5.1 Context Awareness

On Context Awareness: [21, 31, 43, 44, 60, 70, 77, 106, 123, 124, 128, 144, 178, 180, 181, 188, 194, 214, 215, 217, 220, 225, 237, 253, 254]

In principle, a context-aware system executes its main job while respecting some context features. Figure 1.23 displays this basic view. There the system adapts automatically, for instance by raising the voice in noisy surroundings. Alternatively it might ask the user to adapt the sound level. The adaptation to the user and the use situation/task is often called personalization.

For a mashup, context-driven accommodation may begin during service selection and content composition. Two examples:

- If the web communication is weak, a service from nearby is a better choice than an overseas mirror.
- For presentation on a smartphone, mashup content must be restricted to a minimum, while on a tablet, one can afford to spread more information.

[181] lists more reasons for the context-aware mashup modification. [180] proposes a context-aware mashup model including an event-based submodel for adaptivity.

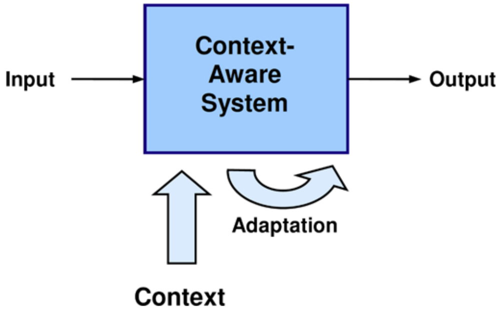


Fig. 1.23 Context-aware system with active adaptation (source: [254])

1.5.2 *Mobile Mashups*

On Mobile Mashups: [7, 15, 21, 26, 30, 31, 41, 44, 55, 56, 68, 77, 81, 106, 112, 118, 120, 128, 144, 161, 175, 183, 195, 204, 213, 223, 224, 226, 236, 248, 249]

Mobile mashups are at home on mobile devices such as smartphones or tablet computers.

As they are made for changing their location all the time, smart devices need ubiquitous computational logistics, e.g. a wireless internet connection. Smartphones feature sensors. The most popular ones pick up GPS location data. There are a camera and a microphone so that the device perceives local context data. Monitoring the surroundings is possible and useful, so that context-aware, adaptive, and personalized applications prosper. A smart mobile web device has to adapt to the user and to the local context, thus improving the user's grasp on the surroundings.

The cast for the most instructive mobile mashups yielded four players with different context-awareness behavior:

- TELAR introducing POIs (Points of Interest) of [30]
- Cooperating mobile mashups [183]
- Personal health mashups [21, 213] digging deep into the everyday life of their users
- Telco mashups integrating telephone and web services

1.5.2.1 TELAR: Mobile Mashup with Context Awareness

The TELAR mobile mashup platform [30] is designed for a Nokia N810 Internet Tablet. It is implemented as a client–server system. End-users can configure mashups running on the mobile browser. Web services and context information of local sensors are integrated, so that the mashup adapts to the current location of the phone (see Fig. 1.24). The Google Map is centered to this position via GPS data. Three web services for Points of Interest (POIs) are mapped in: Fon, Panoramio, and Wikipedia. Local sensor data are handed over in a DCCI specification.[26]

[26] http://telardcci.garage.maemo.org.

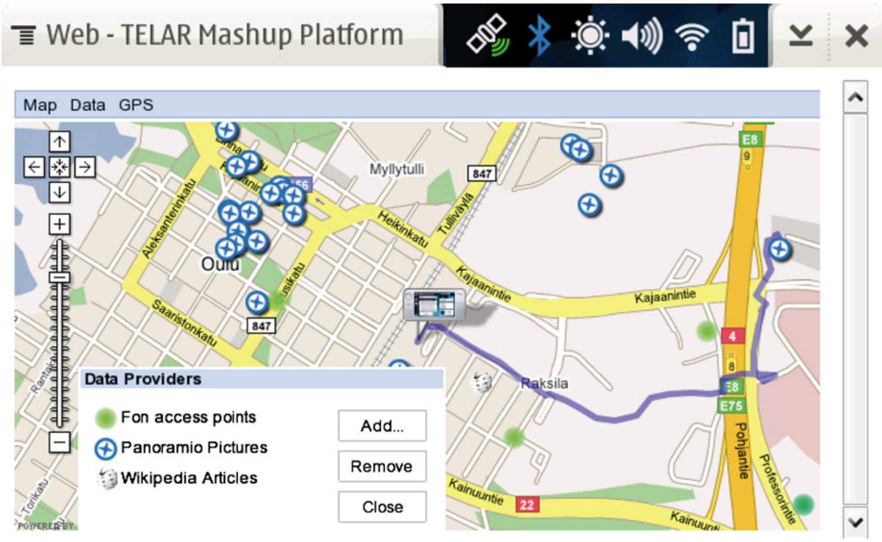


Fig. 1.24 Mobile TELAR mashup sample (source: [30])

1.5.2.2 Cooperating Mobile Mashups

Imagine a group of cooperating mashups: one host and two guests. The host is on an Android phone, the guests/clients are on iPhones. The group (proposed by [183]) compares prices while their users are shopping in a department store. On behalf of their users the guest mashups scan bar codes of interesting goods. The host/server collects them and asks the Google Search API for Shopping[27] for reference prices. It communicates the results to all cooperating mashups via normal SMS or email channels.

Custom agents implemented on the guests and a custom communication center on the host execute the communication processes. Figure 1.25 shows the user-side configuration. Users/mashup composers specify the intended mashup configuration in an XML-based description language called C-MAIDL (Cooperation—Mobile Application Interface Description Language). A mashup generation engine realized in Java reads the description and produces the cooperating mashup applications. Figure 1.26 explains the generation process.

1.5.2.3 Personal Health Mashup

Change of scene—please consider the scenario for the personal health mashup with computerized and connected consumer appliances. Among other things, there are

[27] https://developers.google.com/shopping-search/?hl=de.

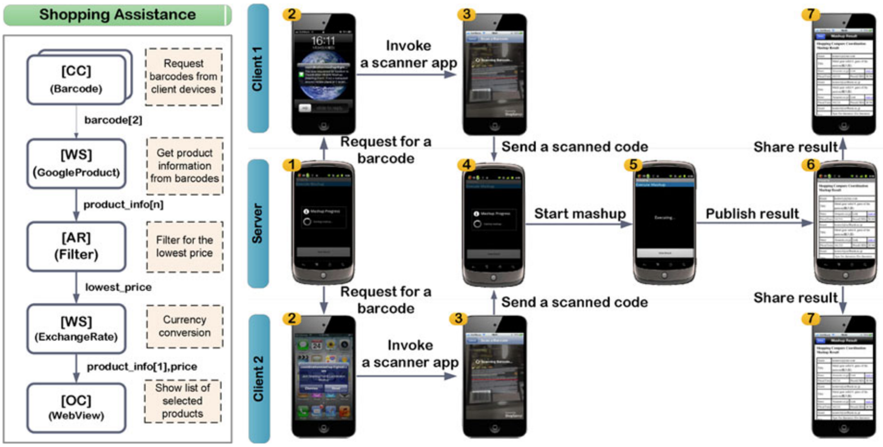


Fig. 1.25 Shopping assistance by cooperating mashups (source: [183])

Fig. 1.26 Mashup generation process for cooperating mashups (source: [183])

bathroom scales[28] that record and upload their users' weight. WiFi activity trackers[29] count the steps and the stairs taken on a day, the hours of sleep and so on. From the smartphone calendar the user's appointment load can be retraced. The users may provide some other data, for instance about their meals. Public sources can add context information, for instance about the weather or about traffic jams. All user-related data can be interpreted statistically with public statistics data as background.

The mashup can present results on common health factors depending on the own behavior so that users understand relationships that they normally do not know. The risks of fast food might be spelled out. Some practical advice can be given, such as: "More steps are better for your health!". Monitoring and recommendations can improve individual well-being, because individuals know more about their own behavior and possibly improve it.

[28]http://www.withings.com.

[29]http://www.fitbit.com/product/features.