WINDMash : A Visual Mashup Environment Dedicated to the Design of Web Interactive Applications

The Nhan Luong[1], Patrick Etcheverry[1], Thierry Nodenot[1],
Christophe Marquesuzaà[1] and Philippe Lopistéguy[1]

[1] IUT de Bayonne - Pays Basque, LIUPPA-T2I
2 Allée du Parc Montaury
64600 Anglet, FRANCE
{thenhan.luong, patrick.etcheverry, thierry.nodenot, christophe.marquesuzaa,
philippe.lopisteguy}@iutbayonne.univ-pau.fr

Abstract. This paper reports on WINDMash, a visual mashup environment to design interactive applications. WINDMash provides for teachers (*end-user designer*) a design environment so that they can design interactive applications based on geographical textual documents by themselves. The design process is divided into three successive facets: data, interface and interaction. The data facet enables the designer to create a processing chain in which the input of textual document may be processed by dedicated services (to automatically retrieve the geographical entities from a textual document and to tag them). The tagged information may then be presented to the designer thanks to visual components (e.g. text, map, and calendar). The designer may configure how the visual components display such tagged information thanks to the interface facet. Last, the interaction facet enables the designer to create interactions between visual components.

Keywords: Interaction Design, Applications of TEL in Geography, Visual Instructional Design Languages, Technologies for Personalisation and Adaptation.

1 Introduction

Local cultural heritage document repositories are characterized by contents which are strongly attached to a territory (i.e. geographical references). Many corpora of these documents are available and it is difficult to use them automatically to retrieve and to make explicit the spatial information that they contain [1]. Different works and experiments have shown that current Web services and frameworks are partially unfitted for the easy design of Web geographical applications. Geographical information is often composed of three complementary features: the spatial feature, the temporal feature and the phenomenon feature [2]. We have focused on the spatial feature in this paper. There are currently many approaches that can be used to visually

create Web applications. However, most of them are designed for *power users*, i.e. users who do not have significant programming skills, but understand the technological difficulties of the problem being treated.

Our research challenge is to provide *end-users*, i.e. users who only acknowledge the problem at the design level and do not have any technical skills, with an online visual environment that they can use to design and to assess by themselves applications based on geographical textual documents, without the intervention of a programmer. This leads us to present WINDMash, a visual mashup environment to design interactive applications. It addresses Active Reading Learning Scenarios. This specific pedagogical activity "*refers to set of high level reading, searching, problem solving and meta-cognitive skills used as readers pro-actively construct new knowledge*" [3], making use of localized documents (travel stories, travel guides) that embed a lot of geographical information about the movements of an actor within a territory. Considering a text as input, WINDMash provides designers with visual artefacts to implement a processing chain that can tag the content of such a text. Interactions can then be defined on the tagged text before code generation based on the WIND[1] API [4]. It is a JavaScript API that may describe interactive applications integrating visual components (e.g. text, map, calendar).

Several key features of WINDMash are identified:
- easy composition thanks to a Web 2.0 drag and drop functionality;
- prevention of incorrect user operations;
- no installation, configuration or maintenance.

In the second section, we present the related works analysing visual design environment proposals. The third section describes a design scenario created with WINDMash. The fourth section presents three design facets using the WINDMash environment. The conclusion synthesizes the paper and discusses the capabilities of WINDMash.

2   Related Works

End-user mashup programming environments are a new generation of online visual tools enabling users to quickly create, for example, Web-based applications [5]. They rely on metaphors that are easy to grasp by non professional coders. They may bind together spreadsheets, the flow of linked processing blocks and the visual selection of GUI actions. [6] provided a good synthesis of available mashup environments, e.g. Yahoo! Pipes[2], Microsoft Popfly, Google Mashup Editor, MashMaker [7], Marmite [8]. Yahoo! Pipes provide tools to combine RSS feeds and JSON data. Microsoft Popfly is a very powerful tool which can be used to create different kinds of mashups. It provides functional base components of various natures. Google Mashup Editor is a developer-oriented Web 2.0 development environment that excels in integrating some Google Services such as Google Maps and Google Calendar. However, Microsoft Popfly was taken down from 2009, Google Mashup Editor was also stopped and

---

[1]   Web INteraction Design

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

migrated to the Google App Engine[3]. MashMaker [7] is a simple tool that is tailored towards end-users which specializes in the creation of Web pages that combine information from diverse sources. Marmite [8] is a mashup solution inspired by Apple Automator, a visual scripting tool for automating repetitive tasks under Mac OS. Users can choose from a number of activities for extracting data from websites, and local and remote databases. Other environments are Vegemite [9], Exhibit [10] and Bill Organiser Portal [11]. Afrous[4] is also a well-known example of mashup platform allowing users to create and to run their application on the web browser.

Mashups may be classified by three dimensions [12]. The first dimension is the nature of the mashups, distinguishing between *data* mashups, *logic* mashups, and *presentation* mashups. The second dimension is the type of users capable of using the framework to create a mashup application, distinguishing between *developers*, *power users*, and *end-users*. The third dimension regards "how" and "where" a mashup is run, distinguishing between *client-side* mashups and *server-side* mashups. According to these three dimensions, our approach tackles both *logic* and *presentation* mashups; targets *end-users* who do not have any programming skills; takes advantage of *client-side* mashups relying on lightweight Web technology such as JavaScript, XML and JSON.

Following P. Rabardel's results on Activity Theory [13], our ongoing research works about WINDMash clearly try to provide designers with anthropocentric instruments, putting the emphasis on creativity, design ideas, and not on technology (that should remain hidden to end-users).

3   An Educational Example

Let us consider a WIND application[5] whose learning objective is to help the learners to discover prefecture around the cities that the user may highlight in the text area (*cf.* Figure 1).

Within the text "*En partant de Lourdes, je suis passé par Juillan qui ne ressemble pas à Asson alors que ces villes ne sont qu'à 25 kilomètres de distance. Je suis arrivé enfin après-midi à Tarbes où j'ai dormi le soir. Le lendemain je suis parti pour Pau: j'ai traversé Ibos, puis Soumoulou et je suis arrivé enfin à Pau où j'ai visité le château.*"[6], there are many words referring places, some of them are cities, others are not (e.g. the Adour river or the Ossau valley). If the user highlights a city (e.g. the city of Lourdes), the map next zooms on the prefecture around such a city (e.g. the prefecture of Lourdes is Tarbes).

---

3   http://appengine.google.com
4   http://www.afrous.com
5   http://erozate.iutbayonne.univ-pau.fr/Nhan/ectel2010/example.html
6   Starting from Lourdes, I went through Juillan that is not like Asson while these cities are only 25 kilometers away. I arrived late in the afternoon in Tarbes, where I stayed the night. Next day I went to Pau : I crossed Ibos, then Soumoulou and finally I arrived at Pau where I visited the castle.
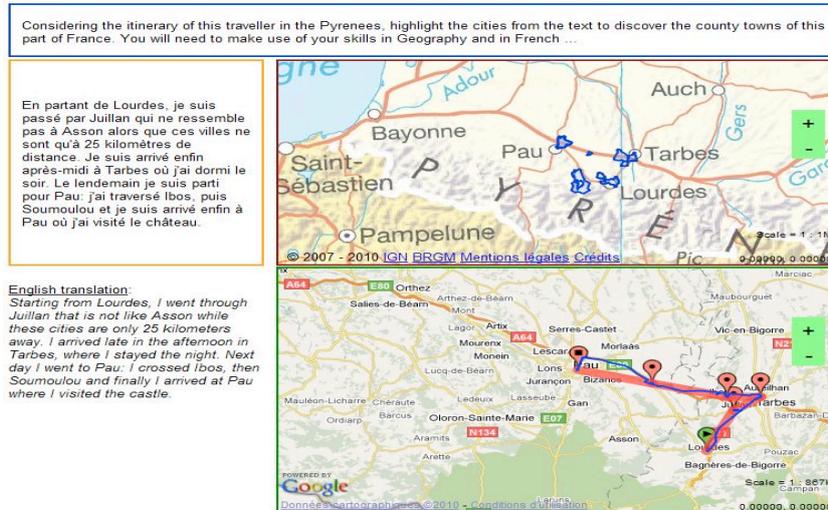
Figure 1. An example for educational purpose.

By using the WINDMash environment, it is not difficult for the users (e.g. teachers) to design such an application. They drag and drop the selected modules in the data facet (*cf.* Figure 3), organize the interface of their generated application in the interface facet, and define the interactions between the visual components in the interaction facet.

4   WINDMash: A Visual Mashup Environment

This section describes the WINDMash environment in which designers, by themselves, may design and automatically generate interactive applications handling geographical information. We highlight the necessary use of an "agile" approach to reduce as much as possible the delay between the design and the evaluation step of authoring tools [14]. Our environment enables designers to quickly produce their interactive applications without any programming skills. The design process is currently divided into three successive facets: data, interface, and interaction.
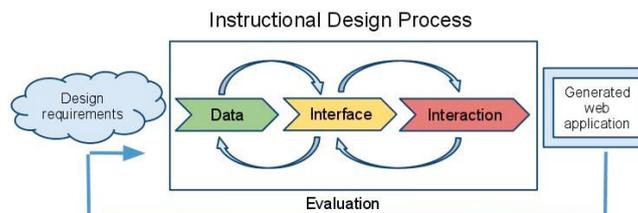

Figure 2. Three facets of the instructional design process.

4.1 The Data Facet

The data facet focuses on the information that will be provided to the learner at runtime. Starting from one or from several plain texts, the designer (teacher) may easily create a processing chain by selecting dedicated modules. This processing chain can automatically transform such input into results that can be either processed again or can be visualised with dedicated viewers: map, calendar and text viewers (*cf.* Figure 3). Available modules can be parameterized by the designer to reach a specific goal, enabling the designer:

- to normalize plain texts into the WIND format (`Normalization`);
- to extract places, itineraries (`PlaceExtraction`, `ItineraryExtraction`);
- to intersect or to join previous results (`Union`, `Intersection`);
- to later visualize results with dedicated viewers to check the design process (`TextViewer`, `MapViewer`, `CalendarViewer`).
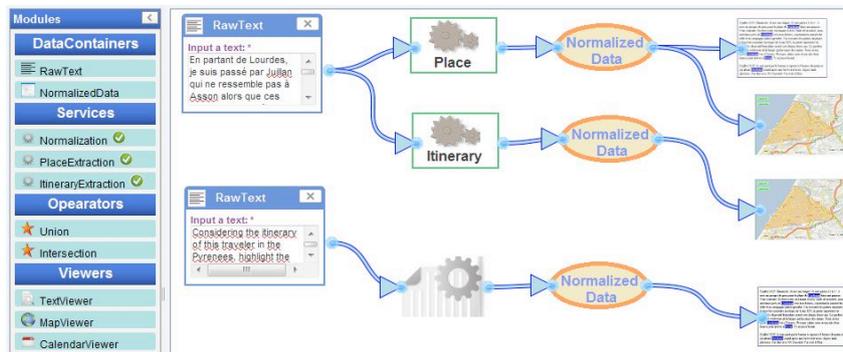
Figure 3. Screenshot example for the data facet of the WINDMash environment.

According to our conceptual model of the data facet :

- A `DataContainer` may be either a `RawText` or a `NormalizedData`. A `RawText` is a textual document (e.g. travel story) used by designers and is linked to the `Service` module. A `NormalizedData` is outputted from a `DataTransformer` module (i.e. `Service` or `Operator`) and contains the data structured by this `DataTransformer` module. The `NormalizedData` may be visualized in the `Viewer` modules (i.e. `TextViewer` or `MapViewer` or `CalendarViewer`).
- The `Service` modules (i.e. `Normalization` or `PlaceExtraction` or `ItineraryExtraction`) module is the gap between the `RawText` and the `NormalizedData`. The `Normalization` service may transform the plain text into the XML format of the `NormalizedData`. The `PlaceExtraction` module implements the GeoStream web service [15] to tag geographical information within (French) textual documents and the

`ItineraryExtraction` module implements the PIIR web service [16] to tag movement verbs and itineraries within (French) travel stories.

- The `Union` and the `Intersection` are `Operator` modules whose inputs are two `NormalizedDatas` and whose output is a `NormalizedData`.
- A `Viewer` module is a visual component. The `TextViewer` may display normalized textual documents with the defined CSS styles on tagged words. The `MapViewer` may display geographical information on a web-based map with multi-layer mapping services (mainly Google Maps layers, Microsoft Bing Maps layers, and France's IGN layers). The `CalendarViewer` may display temporal information on an interactive calendar like Google Calendar. Each `Viewer` displays the corresponding tagged information within its `NormalizedData` input.

4.2 The Interface Facet

The interface facet enables the designers to organize the interface of the generated application (size, position, map provider, zoom level ...). The `Viewers` from the previous facet are concerned here. An interface containing all the `Viewers` is automatically generated and displayed to the designer, enabling him/her to easily and rapidly define the look and feel of each `Viewer`. Each `Viewer` displays the information from the data facet and the designer may then decide where each `Viewer` should be presented on the screen.

Each `Viewer` supports its own configuration settings:

- For a `TextViewer`, the geographical words (`TextPart`) are automatically tagged by the `Service` modules of the data facet.

- For a `MapViewer`, the `MapParts` are automatically marked as geometries on the map layer. A point represents a location, a place; a line represents a route, a river, an itinerary; a polygon represents a region, a city, etc. The designer can choose his/her preferred map layer for his/her application. We highlight the power of multi-layer support of the `MapViewer`, which can support many layers from different providers.

- For a `CalendarViewer`, the concerned time (`CalendarPart`) may be tagged and displayed.

4.3 The Interaction Facet

This facet allows the designer (teacher) to design the interactions between the visual components (`Viewers`) displayed in the previous facet. Currently, by default we automatically offer some interactions between the `Viewers`:

- when clicking on the `TextPart`, the corresponding `MapPart` is focused and the corresponding `CalendarPart` is highlighted;

- when clicking on the `MapPart`, the corresponding `TextPart` is boldfaced and the corresponding `CalendarPart` is highlighted;

- when clicking on the `CalendarPart`, the corresponding `TextPart` is boldfaced and the corresponding `MapPart` is focused.

The XML file describing the `Viewers`, the `SensibleParts`, the `Reactions`, and the `Interactions` is parsed by JavaScriptCodeGenerator[7] to automatically generate executable code based on the WIND API. So, the designers may use the application that they are designing. If the application satisfies them, they commit the design process and save their application. If not, they may come back to the previous steps.

5 Conclusion

In this paper, we have presented WINDMash which is a complete environment dedicated to a web design of interactive applications by end-users. This framework is currently available at this URL[8]. Our environment is both used for the design and for the evaluation/use of the application thanks to automatic code generation abilities. We have defined a design process composed of three main steps: the data facet, the interface facet and lastly the interaction facet. The data facet may start from a geographical textual document that the designer will equip with services designed for example to place and itinerary extraction. The interface facet allows the designer to define the display of the generated application. The interaction facet finally enables the designer to define the behaviour of the interactive application generated from the initial textual document.

Our future work will continue to enrich WINDMash through the integration of many `Service` and `Viewer` modules, particularly those leading to design of educational web-based applications. We are also interested in studying the portability of our approach to mobile devices. The designer will be able to begin his/her design process on the computer, and to continue it with his/her mobile device ; the users may manipulate the application of the designer either on the computer or on the mobile device.

Currently, WINDMash focuses on geographical data that can be extracted from texts and GIS providers. But we plan to empower end-users by enabling them to annotate and then exploit thematic information included in localized documents (travel stories, travel guides) : fauna, flora, activities, etc. Thus, our agenda for the next releases of WINDMash includes enabling the end-user to describe and assess educational driven interactions mixing spatial, temporal and thematic information.

---

[7] It is PHP script to parse an XML file to automatically generate HTML and JavaScript codes.
[8] http://erozate.iutbayonne.univ-pau.fr/Nhan/windmash/

6 Acknowledgements

References

1. Casenave, J., Marquesuzaà, C., Dagorret, P., Gaio, M.: La revitalisation numérique du patrimoine littéraire territorialisé. EBSI-ENSSIBJ. (2004)
2. Gaio, M., Sallaberry, C., Etcheverry, P., Marquesuzaà, C., Lesbegueries, J.: A Global Process to Access Documents' Contents from a Geographical Point of View. Journal of Visual Languages and Computing (JVLC), pp. 3–23. (2007)
3. Murray, T.: Hyperbook Features Supporting Active Reading Skills. Chapter 8 in Web-based Intelligent e-Learning Systems: Technologies and Applications, by Zomgmin Ma (Ed.). Idea Group Publishing: Hershey, PA, pp. 156-174. (2005)
4. Luong, T.N., Etcheverry, P., Nodenot, T., Marquesuzaà, C.: WIND: An Interaction Lightweight Programming Model for Geographical Web Applications. In First International Opensource Geospatial Research Symposium. (2009)
5. Altinel, M. et al.: Damia: A Data Mashup Fabric for Intranet Applications. In Proceedings of the 33rd International Conference on Very Large Data Bases, pp. 1370-1373. (2007)
6. Beletski, O.: End-User Mashup Programming Environments. Technical Document. (2008)
7. Ennals, R., Garofalakis, M.: MashMaker: Mashups for the Masses. In Proceedings of the 27th ACM SIGMOD International Conference on Management of Data, pp. 1116-1118. (2007)
8. Wong, J., Hong, J.: Making Mashups with Marmite: Towards End-User Programming for the Web. In Proceedings of the SIGCHI Conference on Human factors in computing systems, pp. 1435-1444. (2007)
9. Lin, J., Wong, J., Nichols, J., Cypher, A., Lau, T.A.: End-User Programming of Mashups with Vegemite. In: Proceedings of International Conference on Intelligent User Interfaces, pp. 97-106 (2009)
10. Huynh, D.F., Karger, D.R., Miller, R.C.: Exhibit: Lightweight Structured Data Publishing. In: Proceedings of 16th International Conference on World Wide Web, pp. 737-746 (2007)
11. Ro, A., Xia, L.S., Paik, H., Chon, C.H.: Bill Organiser Portal: A Case Study on End-User Composition. In: 9th International Conference on Web Information Systems Engineering, LNCS 5176, pp. 152-161 (2008)
12. Albinola, M., Baresi, L., Carcano, M., Guinea, S.: Mashlight: A Lightweight Mashup Framework for Everyone. In 2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web. (2009)
13. Rabardel, P.: Les hommes et les technologies: Approche cognitive des instruments contemporains. Armand Colin (1995)
14. Vickoff, J-P.: Systèmes d'Information et Processus Agiles. Hermes Science Publication. (2003)
15. Sallaberry, C., Royer, A., Gaio, M., Loustau, P., Joliveau, T.: Geostream: A Spatial Information Indexing Web Service. In First International Opensource Geospatial Research Symposium. (2009)

16.   Loustau, P., Nodenot, T., Gaio, M.: Design principles and first educational experiments of PIIR, a platform to infer geo-referenced itineraries from travel stories. International Journal of Interactive Technology and Smart Education, pp. 23-29. (2009)