Embedding MindMap As a Service for User-Driven Composition of Web Applications

Adnene Guabtni
National ICT Australia
NSW 1435 Australia
aguabtni@nicta.com.au

Stuart Clarke
The University of New South Wales, NSW 2052 Australia
stuni@gho.st

Boualem Benatallah
The University of New South Wales, NSW 2052 Australia
boualem@cse.unsw.edu.au

ABSTRACT

The World Wide Web is evolving towards a very large distributed platform allowing ubiquitous access to a wide range of Web applications with minimal delay and no installation required. Such Web applications range from having users undertake simple tasks, such as filling a form, to more complex tasks including collaborative work, project management, and more generally, creating, consulting, annotating, and sharing Web content. However, users are lacking a simple but yet powerful mechanism to compose Web applications, similarly to what desktop environments allowed for decades using the file explorer paradigm and the desktop metaphor. Attempts have been made to adapt the desktop metaphor to the Web environment giving birth to Webtops (Web desktops). It essentially consisted of embedding a desktop environment in a Web browser and provide access to various Web applications within the same User Interface. However, those attempts did not take into consideration to the radical differences between Web and desktop environments and applications.

In this work, we introduce a new approach for Web application composition based on the mindmap metaphor. It allows browsing artifacts (Web resources) and enabling user-driven composition of their associated Web applications. Essentially, a mindmap is a graph of widgets representing artifacts created or used by Web applications and allow to list and launch all possible Web applications associated to each artifact. A tool has been developed to experiment the new metaphor and is provided as a service to be embedded in Web applications via a Web browser's plug-in. We demonstrate in this paper three case studies regarding the DBLP Web site, Wikipedia and Google Picasa Web applications.

Categories and Subject Descriptors

D.2 [**Software**]: Software Engineering
; H.m [**Information Systems**]: Miscellaneous

General Terms

Experimentation

Keywords

MindMap, Web Application, User-Driven Composition

WWW 2011, March 28–April 1, 2011, Hyderabad, India.
ACM 978-1-4503-0637-9/11/03.

1. INTRODUCTION

There is a significant shift underway in how software is delivered to end users. Web-based software development, coupled with the scalability offered by Cloud Computing, have changed the way people use computers and the Internet. Every day more software is being delivered over the Web in the form of full applications running in a Web browser and accessible from any computer.

Web applications offer various processes allowing to manipulate artifacts. For example, in a social networking Web application (e.g. Facebook), artifacts are people, profiles, pictures, videos or messages and the offered processes allow to create, modify, publish, share or access those artifacts. We refer to the Open Provenance Model (OPM) [3], illustrated in figure 1, to define the three types of entities involved in a Web application environment: artifact[1], process[2] and agent[3]. In the Web environment, an agent corresponds to the end user of a Web application.

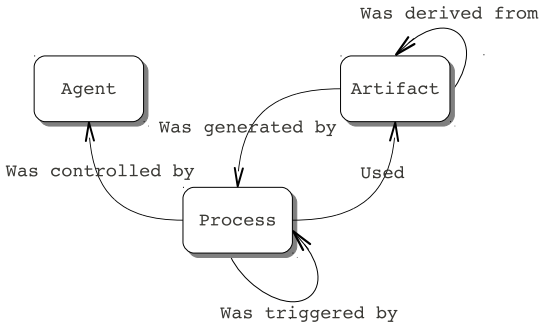

Figure 1: The Open Provenance Model

The World Wide Web is evolving towards a very large distributed platform allowing ubiquitous access to a wide range of Web applications producing various types of artifacts. Such Web applications range from having users undertake simple tasks, such as filling a form, to more complex tasks including collaborative work, project management, and more generally, creating, consulting, annotating, and sharing arti-

[1]An artifact is "*an immutable piece of state, which may have a physical embodiment in a physical object, or a digital representation in a computer system*" [3]
[2]A process is "*an action or series of actions performed on or caused by artifacts, and resulting in new artifacts*" [3]
[3]An agent is "*a contextual entity acting as a catalyst of a process, enabling, facilitating, controlling, or affecting its execution*" [3]

facts. However, users are lacking a simple but yet powerful mechanism to compose Web applications, similarly to what desktop environments allowed for decades using the file explorer paradigm and the desktop metaphor. They are used to browse, create and share artifacts (files) by composing the available applications (installed on a single machine). Contextual menus and Graphical User Interfaces, but also command line tools, are extensively used to enable such composition in a desktop environment.

Many attempts have been made to adapt the desktop metaphor to the Web environment giving birth to Webtops (Web desktops) [2, 5]. It consisted of embedding a desktop environment in a Web browser and provide access to various Web applications within the same User Interface. Solutions have been proposed to even port automatically desktop application to the Web without requiring any changes in the source code of that application [4]. However, those attempts consisted of simply porting the exact same desktop metaphor to the Web with no or little consideration to the radical differences between Web and desktop environments and applications.

In a desktop environment, artifacts are produced by desktop applications and stored and managed as files. A hierarchical file explorer allows to browse those files and perform various actions on them by enabling a user-driven composition of all available desktop applications. For example, after creating a document using a word processor, the user can browse to created files and use a file compression application to obtain a compressed file. In that scenario, two applications have been used to achieve two processes: creating a document and compressing it. The file explorer allows to browse the hierarchy of files and folders, provide for every file, usually using a context menu on right click, a list of all available applications that could manipulate such file, and allow users to launch those applications.

Contrary to desktop environments, Web applications do not necessarily store artifacts into files but rather as resources accessible at Unique Resource Locations (URL). Moreover, artifacts' relationships are not necessarily hierarchical (e.g. social networks). Moreover, Web applications are disparately distributed on the Web contrary to desktop applications installed on the same machine. All of those characteristics add to the challenge of a user-driven composition of Web applications.

In this work, we introduce a new approach for Web application composition based on the mindmap metaphor [1]. It allows browsing artifacts (Web resources) and enables user-driven composition of their associated Web applications. Essentially, a mindmap is a graph of widgets which has a root widget from which all other widgets branch out with possible interconnections between the branches. Each widget is representing an artifact created or used by Web applications and allow to list and launch all possible Web applications associated to each artifact. A tool has been developed to experiment the new metaphor and is provided as a service to be embedded in Web applications via a Web browser's plug-in. We implemented the proposed approach as a service similarly to how Google Maps are used as a service in various Web sites. The resulting service, called MMAAS (MindMap As A Service) is demonstrated in this paper using three scenarios where the MMAAS is embedded to various popular Web applications including Wikipedia, Google Picasa and the DBLP Web site.

In the rest of the paper, we will first describe in details the different components of the proposed service, their implementation and deployment and finally, three demonstration scenarios on (1) the database community's citation database DBLP, (2) Wikipedia and (3) the Google Picasa Web site.

2. SYSTEM ARCHITECTURE

The MMAAS system's architecture is illustrated in figure 2. It is composed of three main layered components described as follows:

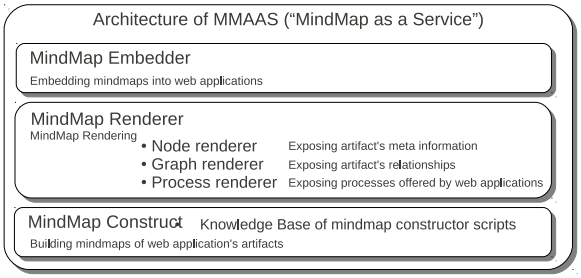

Figure 2: Architecture of the MMAAS system

2.1 MindMap Constructor

This component is in charge of constructing a mindmap representation of the artifacts in a given Web application for a given context[4]. A knowledge base of mindmap constructor scripts, written in javascript has been developed. A mindmap constructor script generates an XML representation of the artifacts, their relationship, and the possible processes available on the Web for such artifacts. The generated mindmap contains at least one artifact, and necessarily one of them is selected as the "focus" and will be considered as the center of the mindmap. The output of a mindmap constructor script is saved in XML format following a defined XML schema for mindmaps.

2.2 MindMap Renderer

This component is in charge of rendering the mindmap generated by mindmap constructor scripts. An entity-relationship model for artifacts, processes and their relationships is illustrated in figure 3. Each entity is managed by sub-components of the MindMap Renderer described as follows.

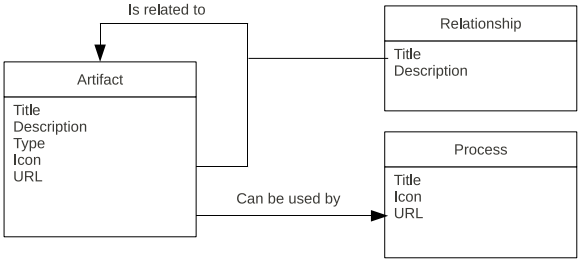

Figure 3: Entity-Relationship model for artifacts, processes and their relationships

[4]The context in a Web application corresponds to all available information about a user's session.

2.2.1 Artifact's Widget Renderer

The Artifact's Widget Renderer allows to render a widget, as illustrated in figure 4, exposing the artifact's meta information as well as the list of processes available on the Web for such an artifact. Artifact's meta information rendered in the widgets consists of a title, a type, a description, and an icon. The list of processes available on the Web for an artifact is rendered using a dynamic drop down menu. A click on any process in the drop down menu allows to instruct the execution of the process and render its result using the Process Controller.

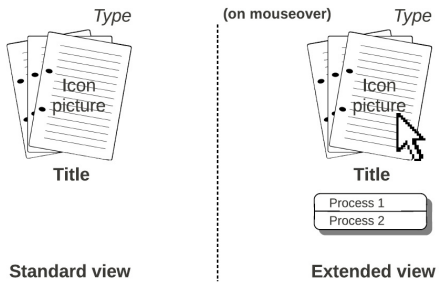

Figure 4: Artifact's widget rendering

2.2.2 Graph Renderer

The Graph Renderer allows to display and position all Artifact's Widgets and their relationship using an organic layout. The "focus" artifact is displayed in the center of the layout and the positioning of all other artifacts will depend on their relationship to the central one. An organic layout algorithm, optionally using attraction-repulsion forces, is used to place each artifact at the best possible position. Relationships between artifacts are displayed using non oriented edges displaying their titles.

2.2.3 Process Controller

The Process Controller allows to launch Web applications' processes when requested. It is used by the the Artifact's Widget Renderer when the end user selects one of the proposed processes. When a Web application is launched, its result is rendered as an embedded Web page within the mindmap layout.

The Google Web Toolkit (GWT) has been used to build the MindMap Renderer including its three core sub-components. GWT is a widget based comprehensive toolkit for Web front-end development and has been coupled with advanced rendering libraries to address the specific needs of this work as follows:

- The SmartGWT library expands on GWT to provide a great deal of aesthetically pleasing features, namely User Interface Objects (UIObjects) which include images, advanced menu systems and layouts. It provides rich programmable features for each, including customized handlers for many mouse operation. It is possible through the use of layouts to stack UIObjects of different types, i.e. images, labels menus etc to create a customized object which can interact as a unit with the end user. This library has been used for implementing the Artifact's Widget Renderer as well as the Process Controller.
- The GWT-Graphics library have been used to draw artifact's widgets connections in the implementation of the Graph Renderer. Connecting lines can be drawn to connect any two UIObjects.

2.3 MindMap Embedder

This component is in charge of embedding rendered mindmaps into existing Web applications as a service. Such service becomes available to Web application for exploring artifacts, their relationships and the processes available over the Web for such artifacts. The objective of the Embedder is to allow end-users to embed mindmaps within any of the supported Web applications similarly to how Google Maps are used as a service in various Web sites.

Greasemonkey[5], an Add-on for the Mozilla Firefox Web browser, has been used to implement the MindMap Embedder. It allows to customize the way a Web page displays using user defined scripts in JavaScript. A repository of all available mindmap constructor scripts for various Web applications has been created and allows end-users to quickly install those scripts on their own Web browser. Once Greasemonkey and the required scripts are installed, browsing the supported Web applications will automatically display a special icon on the top right corner of the screen. A click on that icon will display the mindmap corresponding to the Web application being visited as illustrated in the following sections.

3. DEMONSTRATION SCENARIOS

Three case studies have been conducted regarding the DBLP Web site, Wikipedia and Google Picasa. For each case study, the mindmap tool has been embedded to the Web application and a mindmap constructor script has been developed. A demonstration video is available at

`http://soc.cse.unsw.edu.au/demo/mindmapdemo/`

3.1 Exploring the DBLP artifacts

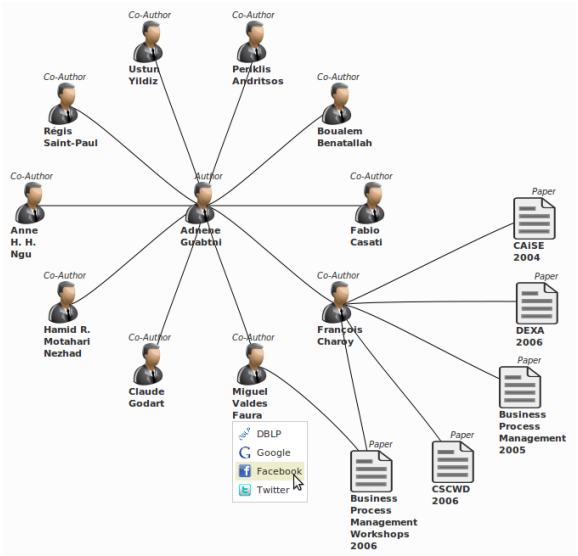

Figure 5: Exploring the DBLP artifacts

[5] `http://www.greasespot.net/`

DBLP is a Web application providing bibliographic information on major computer science journals and proceedings. Enabling mindmaps on DBLP when an author's DBLP page is visited is of great interest as bibliographical information is naturally a graph (of authors, papers, conferences, journals, etc.). The mindmap has been embedded as a service using a mindmap constructor script specifically implemented for DBLP. Once the mindmap is constructed, an icon entitled "MMasS" appears on the top right side of the visited DBLP page. A click on such an icon shows the constructed mindmap, as illustrated in figure 5, having the author in its center as well as co-authors and papers they've contributed. Such mindmap is automatically generated by the mindmap constructor based on the DBLP page's content. A further click on the MMAAS icon makes the mindmap hidden again to return to the classical DBLP view.

Using mindmaps within DBLP allows to offer the possibility for the end-user to compose various processes offered by DBLP and other third party Web applications. For example, when focusing on a co-author, the following processes are offered in the demonstration: DBLP to display the publication track record of the author, Google to search for the author using the Google search engine, Facebook to get in touch with the author using Facebook, Twitter to display the author's tweets, etc.

3.2 Exploring Wikipedia artifacts

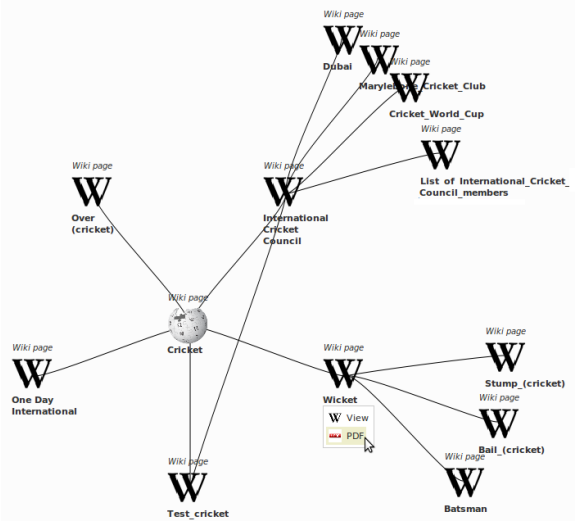

Figure 6: Exploring Wikipedia artifacts

Wikipedia is organized in inter-related Wiki pages. Using mindmaps allows to browse efficiently those pages but also to extend the capabilities offered by Wikipedia. For example, we propose to associate a "PDF export" process to each Wiki page using a third party Web application. Therefore, the end-user can export Wiki pages in PDF format.

3.3 Exploring albums on Google Picasa

Browsing albums and pictures using the mindmap metaphor is of particular interest from visualization point of view. This is due to the fact that all pictures can be visualized within the graph offering a quick overview. Once an album or a picture is selected, various options are offered such that "view slide show" which allows to start a slide show, or

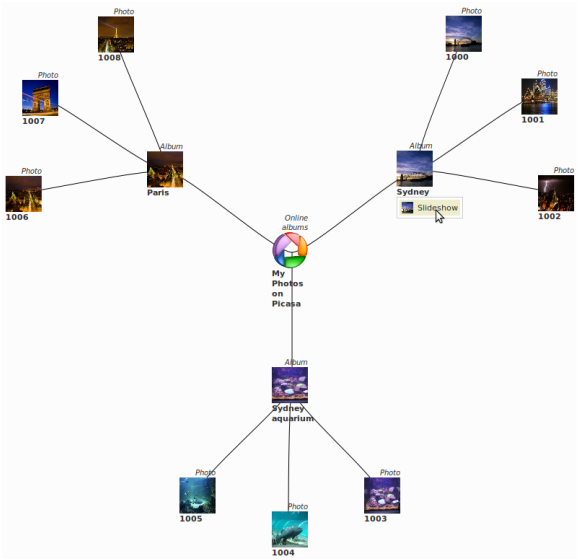

Figure 7: Exploring albums on Google Picasa

"comment" which allows to add comments on the albums or pictures.

4. CONCLUSION AND FUTURE WORK

In this paper, we demonstrated how the mindmap metaphor can be used to enable user-driven composition of Web applications. The proposed approach is the equivalent of what the file explorer paradigm achieves in desktop environments for composition of applications. However, there are many differences between desktop and Web environments and these are exposed and addressed in the proposed approach. A tool has been developed to experiment the new approach and this paper demonstrated three case studies regarding the DBLP Web site, Wikipedia and Google Picasa.

Further development would extend the capabilities of the proposed tool by introducing additional mindmap constructor scripts to support a larger list of Web applications. Furthermore, mindmap constructor scripts implemented in the above demonstration scenarios used rendered content only as a source of information. However, extracting additional information, using alternative APIs, can drastically improve the quality of the constructed mindmaps as well as the performance of the constructor scripts.

5. REFERENCES

[1] Tony Buzan and Barry Buzan. *The Mind Map Book*. BBC Books, London, 2 edition, 1995.

[2] Cheukyin Chan. Wood - web-based object-oriented desktop. In *WWW Posters*, 2001.

[3] Luc Moreau et al. The open provenance model core specification (v1.1). *Future Generation Computer Systems*, In Press, Corrected Proof:–, 2010.

[4] Arno Puder. Extending desktop applications to the web. In *Proceedings of ISICT'04*, pages 8–13. Trinity College Dublin, 2004.

[5] Haifeng Shen, Zhonghua Yang, and Chengzheng Sun. Collaborative web computing: From desktops to webtops. *IEEE Distributed Systems Online*, 8, 2007.