Differentiating and Defining Portlets and Widgets: A Survey Approach

Effie L-C Law[a], Daniel Müller[b], Anh Vu Nguyen-Ngoc[a]

[a]Department of Compuer Science, University of Leicester, LE1 7RH Leicester, UK
[b]IMC information multimedia communication AG, 69115 Saarbruecken, Germany

[a]{elaw|anhvu}@mcs.le.ac.uk, [b]Daniel.Mueller@im-c.de

Abstract. Widget is not a new computing concept, but it has recently attracted much research interest because of its potential uses in a range of technological products/services. In assuming the challenge to develop an evaluation scheme for widgets/widget-based applications, some conceptual ambiguities about widget, especially its distinction from portlet, need to be clarified first. A survey aiming to collect opinions in this regard was conducted. Qualitative data were analyzed with *Cmap* (a content map tool) and *Nvivo8* (a content analysis tool). Widgets invite more diverse interpretations than portlets, for which specifications have existed for some time. Relationships between these two entities are not consistently understood by respondents. A consensual definition of widget entails more scientific discourses in the future and will have significant implications for developing a robust evaluation framework for widget-based platforms.

Keywords: Portlet, Widget, Concept map, Content analysis, Open and responsive learning environment

1 Introduction

Widget is not a new concept in the computing world. Some researchers claim that the emergence of widgets (or their precursors) can be traced back more than 25 years ago when GUI was first designed for home use [1]. Since 2003 when desktop widgets were brought to Mac OS X users, widget has become a buzzword and prevailed in various contexts such as websites (especially social network pages), mobile devices, and desktops. In the same year, the first Java Portlets Specification JSR 168 (http://jcp.org/en/jsr/detail?id=168) was published; it addresses the issue of interoperability of portlets across different portal platforms. In contrast, it was not until very recently in July 2009 when W3C released Widgets 1.0 Specification[1]. Despite their different paths of evolution, the similar characteristics and functions of

[1] Accordingly, widget is defined as: "A widget is an interactive single purpose application for displaying and/or updating local data or data on the Web, packaged in a way to allow a single download and installation on a user's machine or mobile device. A widget may run as a stand-alone application (meaning it can run outside of a Web browser), and it is envisioned that the kind of widgets being standardized by this effort will one day be embedded into Web documents.

portlets and widgets have tempted people to use the two terms synonymously. Interestingly, each of them is associated with a bunch of phrases or names that may (or not) imply their key properties, such as “reusable Web module”, “pluggable UI components” for portlets and “gadgets”, “snippets”, “flakes” for widgets. Out there in the scientific and grey literature, there exist a number of formal as well as quasi definitions for portlets and widgets. Due to their crossroad developments, researchers and practitioners are somewhat confused how widgets should be distinct from portlets or not at all. Whilst several attempts to develop and validate an evaluation scheme for portlets and portals have been undertaken (e.g. [2]), to our best knowledge, it is yet to be done for widgets and widget-based applications. We intend to assume this challenge. The first task we need to tackle is differentiating portlets from widgets, i.e., *to what extent they are similar or different*? Answers to this key question will lay the cornerstones for our further work, because they will have the significant implication whether the existing evaluation frameworks for portlets can somehow be adopted and adapted for widgets. Specifically, widgets, given their versatility and flexibility, will presumably play an increasingly important role in technology-enhanced, open and responsive learning environments (cf. http://www.role-project.eu).

As an effective and efficient means to collect views from experts working on the related topics, we have developed a survey (Section 2). The data enable us to clarify certain conceptual ambiguities and provide us a solid foundation to proceed with our plan to develop a valid evaluation framework for widgets/widget-based applications.

2 Methodology

Survey Design and Administration: To maximise the response rate, our survey is designed to be succinct and precise. It consists of four main open-ended questions, which are further divided into sub-questions (Table 1). The survey has been administered via personalized emails to a group of European experts on portlet- and widget-related topics. It has also been posted onto a personal blog (http://www.pontydysgu.org/2009/06/portlets-and-widgets/) and LinkedIn ROLE community (http://www.linkedin.com/groups?gid=1590487). We have gathered 24 completed responses with various levels of details. They are designated as R1, R2, and so forth. Most of the respondents are developers with strong technical background whereas the other respondents are more mediators (who collect technical requirements and communicate them to the development team) or active users of portlets/widgets. Four of the respondents are female.

Data Analysis Tools: To enhance our understanding of survey data, we employ several data analysis tools, viz. *CmapTools* and *Nvivo*, to consolidate our empirical findings. Here we briefly describe each of them:

CmapTools is a free software application enabling users to construct, navigate, share and criticize knowledge models represented graphically as concept maps. Concepts are enclosed in circles or boxes. Related concepts are linked hierarchically by connecting lines annotated with words/phrases to specify their relationships. A concept is defined as a perceived regularity in (records of) objects/events designated

by a label [3]. With CmapTools we have built concept maps on the responses to Q.1 and Q.2 (Table 1).

Table 1: Survey questions

Q1. Questions about **Portlets**
1a) Please give your **definition** of portlets
1b) Please list specific **characteristics** (=attributes, properties) of portlets
1c) Please list specific **features** (=functionalities) of portlets
Q2. Questions about **Widgets**
2a) Please give your definition of widgets
2b) Please list specific **characteristics** (=attributes, properties) of widgets
2c) Please list specific **features** (=functionalities) of widgets
Q3. Please tell us, what do YOU consider as the **major differentiator(s)** between:
3a) Portlets and Widgets? (cf. Wikipedia on Web widget)
3b) Widgets and Java Applets? (cf. W3C Widget requirements)
Q4. Please share with us YOUR ideas how to evaluate:
4a) Portlets? 4b) Widgets?

Nvivo 8 is a proprietary software package supporting users to analyse qualitative data of different sizes, ranging from simple text to multimedia data. It facilitates importing, classifying, sorting, and some other manual qualitative data analysis tasks. It also provides the possibility to query data and visualise them with charts and models. We have deployed Nvivo 8 to code the survey responses. The coding results enable us to further explore the data in a greater detail, both manually and with the use of Nvivo 8, to identify the concepts commonly elicited from the respondents and to highlight their possible misunderstandings about and contradictory interpretations of portlets and widgets.

3 Concepts Associated with Portlet Definitions and Features

In response to Q.1 of the survey, a variety of concepts are elicited. Whilst most respondents tended to provide a succinct statement for defining porlet (1a), a few of them provided an elaborated answer. Besides, quite a number of respondents remark that it was hard to differentiate between characteristics and features and thus collapsed their answers to 1b and 1c into one. Hence, we compiled the answers to the three parts of Q.1 and constructed concept maps accordingly.

Fig. 1 shows that the main concepts associated with portlets are: portal, web-based application, and widget. Each of them is connected to some sub-concepts. Presumably the diagram is self-explanatory Note that what depicted are respondents' interpretations of portlets, which may or may **not** be valid.

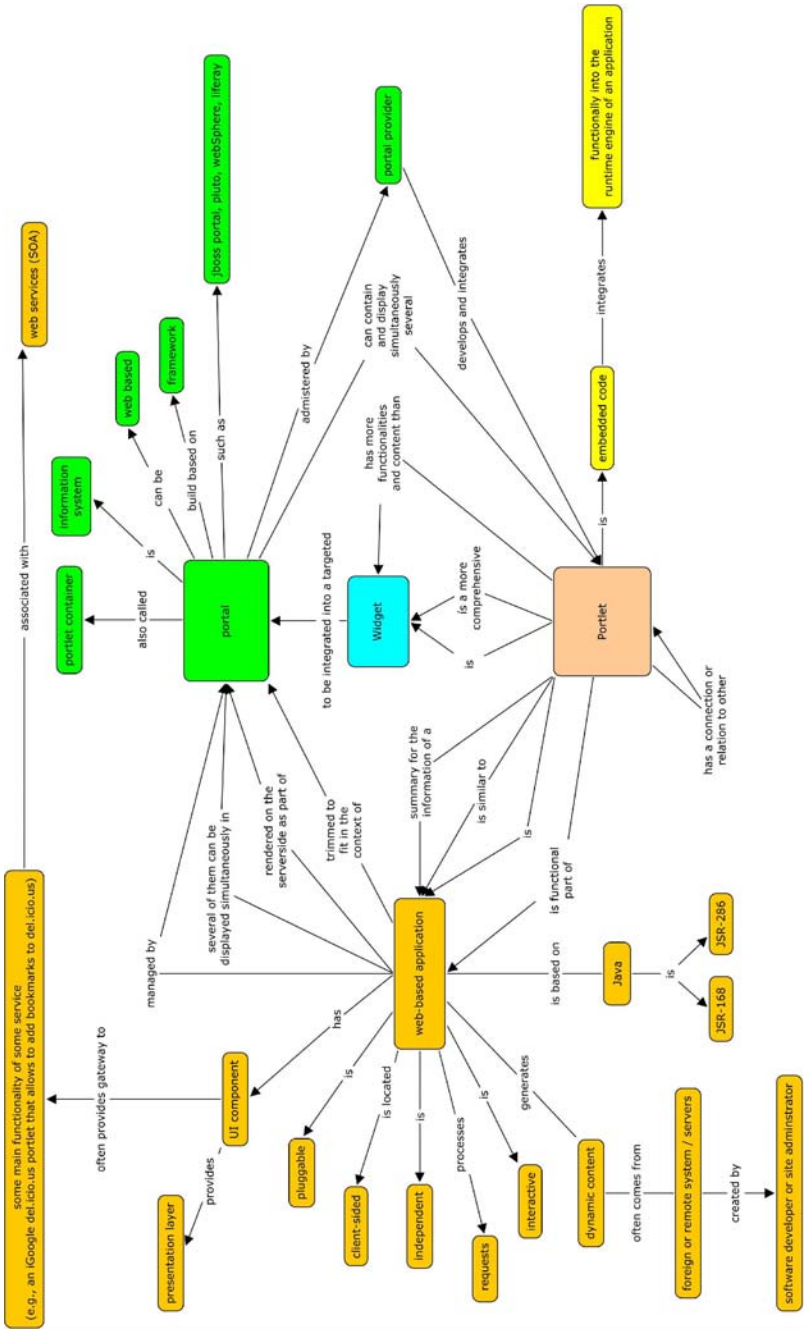

Fig. 1. Concept map of a comprehensive portlet definition

To complement as well as supplement the above findings, Nvivo results allow us to explore individual ideas more in detail, how often they are claimed and by whom, and how they are connected.

As portlets are standardised with JSR 168/286 developed under the Java Community Process, it is logical that eight of the respondents claim that portlets are Jave-based and another four mention that portlets are independent, small applications. Eleven out of 24 respondents define a portlet as a user interface component. 62.5% of the respondents state that portlets must be associated with or run in a web portal. Besides, four respondents remark that porlets should serve as a functional part of an application. Interestingly, several respondents explain portlets in terms of widgets: a portlet is a "*widget to be integrated in a targeted Web portal*", "*a more comprehensive widget*", or "*not far from widget*".

In describing characteristics and features of portlets, there are communal as well as contradictory responses. Seven respondents claim that portlets are embeddable or pluggable. Another four remark that portlets "*produce document fragments*" that can be combined in Web pages. In addition, the functionality of portlets depends on the portal in which they are running, as clearly defined in the JSR: "*The content generated by a portlet is also called a fragment … a piece of markup (e.g. HTML, XHTML, WML)…. The content of a portlet is normally aggregated with the content of other portlets to form the portal page*". Some respondents assert that portlets can be configured and customisable, though the configuration can be restricted.

Interoperability of portlets is another key feature mentioned by ten respondents. Some of them write that a portlet can "*interchange with or possibly talk to other portlets*", and portelts can "*communicate to a certain extent (difficult problem, though)*". Some refer to standards and specifications that support the portlet interoperability, such as "*event-based inter-portlet communication (IPC)*" and "*services-API to other portlets*".

Despite the relatively established status of portlets, respondents hold contradictory views about their characteristics and features. One respondent claims that a portlet is a client-sided component whereas two reply that it is server-side technology. Some mention that a portlet "*highly relies on the application logics on the server*". Others say that a portlet has "*interaction with backend services"* or follows "*client/server communication*" and "*MVC paradigm*", in which the application logics are carried out mostly at the server side and the client side is used mostly for displaying the data process results. Other conflicting views include static vs. dynamic content generation and lightweight vs. heavyweight.

4 Concepts Associated with Widget Definitions and Features

As explained in Section 3, we merged the responses to the sub-questions when performing data analysis. Fig 2 illustrates the Concept maps on the concepts elicited from the respondents when interpreting widgets. The diagram is intuitive as well as informative.

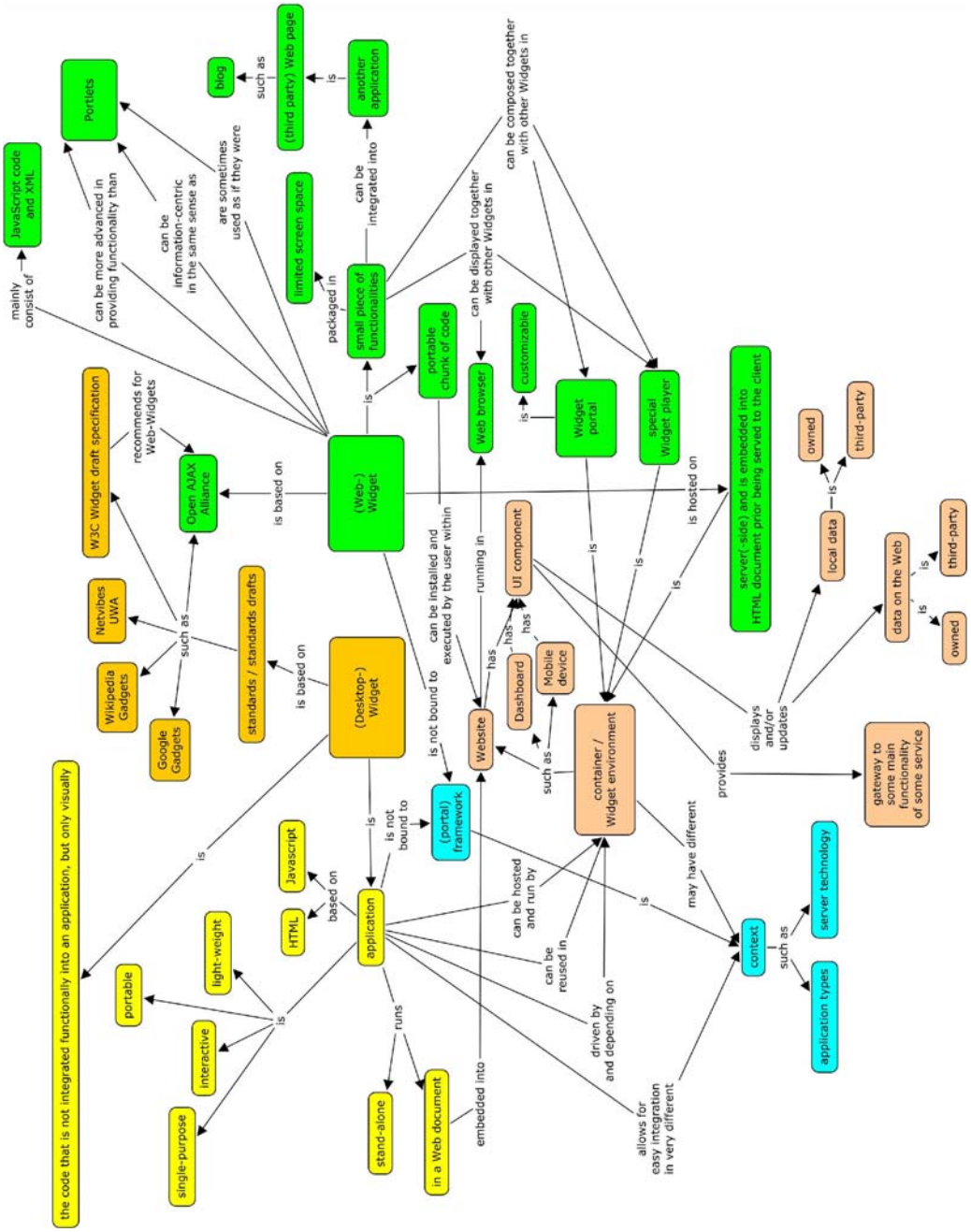

Fig. 2. Concepts of a comprehensive widget definition

Apparently, there is no clear definition about widgets. The respondents provided a diversity of concepts in connection to widgets. One respondent admits that "*widgets are less well and less explicitly defined than portlets*". Twelve (or 50%) respondents remark that widgets are simple and small applications. Four respondents mention that widgets can be defined user interface components (cf. 11 mention so for portlets). According to three respondents, 'desktop widgets' should be explicitly distinguished from 'web widgets'. As pointed out by six respondents, developers tend to use a lot of HTML, Java and Javascript to develop widgets; however, they took only web widgets into account. There are some possibly controversial and contradictory concepts in the widget definitions. Here are some of the interesting findings:

- Platform dependencies: Eight respondents only refer to widgets as web applications or as applications that run only on web platforms. Most other respondents tend to think that widgets can run on different platforms, including web, desktop and mobile devices;
- Implementation: According to one respondent, widgets are "*simple HTML and Javascript applications*". Similarly, another asserts that "*widgets are small applications built using Javascript, CSS and HTML"*. There are also other responses that tend to support those statements, including "*widgets tend to use web standards (HTML, CSS, Javascript)*" or "*html snippets plus javascript code*", "*mainly consisting of Javascript and XML*", "*using a lot of Javascript*".
- Functionalities: One respondent mentions that widgets are "*not integrated functionally into an application, but only visually*". In fact, a widget can also perform rather complicated functionalities. Many respondents agree that a widget can be seen as an entity providing specific and useful functionality. They remark that a widget can "*have any functionality, except heavyweight application functions"*, can provide *"quite complex application logics at the client (in the browser) in combination (or not) with functionality from a server*", can be "*more advanced in providing functionality*" or can "*process data as well as display data and result of processed data*". Besides, one respondent argues that widgets "*are about the web-based visualisation a functional part of a web application*". Similarly, another respondent states that widgets are "*developed for being included and displayed in a software application*". However, many respondents refer to a widget as an application itself.

Concerning the characteristics and features of widgets, we identify both consensual and contradictory concepts in the survey responses. Many respondents agree that widgets are possibly configurable and/or customisable. However, one respondent argue that a widget is "*customisable for developers, for users it is just an 'interaction element' or 'graphical unit'…*" and another remarks that only "*widgets layout format is configurable by users*". Various properties of widgets are mentioned by respondents: light-weight, single-purpose, independent from the framework or infrastructure in which they run, interactive, reusable, portable or embedded. Concerning the technologies used for widgets development, beside the discussions on the definitions, five respondents explicitly mention AJAX (i.e. asynchronous JavaScript and XML).

The issue of widget interoperability is highly relevant as it can help develop useful and powerful mashup applications such as personal learning environments (PLEs). Some respondents tend to support that widgets are interoperable, but some

others are rather against it. The latter argue that technically it is very difficult to support widget interoperability because standards for inter-widget communication is still lacking.

5 Comparisons between Portlets and Widgets

In discussing the difference between a portlet and a widget (Q.3 in Table 1), the respondents again expressed diverse opinions. In fact, many of them reiterated the concepts that they had already provided for Q.1 and Q.2. An answer with a slightly high level of agreement is on the platform where portlets and widgets can run: Portlets need to run in a portal while the widgets can run in different platforms. The context in which widgets can be used is broader (i.e., widgets are platform independent). Another point is that widgets seem to use more client-side technologies and more lightweight. In contrast, portlets are more complex and heavyweight. Apparently, respondents hold different views how portlets are distinct from widgets in terms of ease of implementation and integration (widgets easier) and pace of evolution (widgets faster). An interesting argument is the subsumptive relationship between the two objects with some respondents seeing portlets as a subset of widgets.

To sum up, we tend to agree with the expressed view that "*widgets are less well and less explicitly defined than portlets*". As portlets are established technologies standardised in the Java community, developing portlet-based applications seem to be clear and straightforward though it may be complicated in the sense that developers should follow some rules defined in the specifications. However, the portlet usage is quite limited in the web portal. We also agree with some respondents that portlets are subsumed by widgets. Widgets can be used as standalone applications or combined with other widgets to form bigger applications, especially when the future widget-widget communication APIs are standardised. Widgets do have a great potential.

6 Concluding Remarks

While the concept of widgets is not new, how they can be exploited by today's technologies is perceived to an innovative challenge. Resolutions to this challenge will presumably have strong impacts on the future development of a range of products/services, given (potential) versatile functionalities and platform independence of widgets. The hype surrounding widgets has recently been turned into something that takes some time to ripe - the latest W3C working draft on Widgets 1.0 API and Events which is directly related to the widget development. Meanwhile, in the market many companies offer widget development using different technologies and programming languages. It is deemed important to enable widgets to be deployed in various contexts and platforms.

Another challenge related to widget development is widget evaluation (i.e. design and evaluation of two faces of the same coin). As mentioned earlier, what has driven us to undertake the task of distinguishing portlets from widgets conceptually is a more ambitious goal to develop an evaluation scheme for widgets/widget-based

applications. This is where Q.4 of our survey comes into play. We will tackle this challenge as our next contribution to the wider research community.

Some reflections on the usability and usefulness of the three data analysis tools deployed for this study are presented here. While CmapTools is powerful to visualize results in an intuitive manner, it is not easy to deploy as it entails quite a lot of manual work. Similarly, Nvivo 8 allows users to abstract relationships among concepts (i.e. nodes) at different levels and support other qualitative data management functions. However, it is a heavy application in the sense it necessitates some form of training to use it effectively and non-trivial manual operations.

References

1. Kennedy, N. (n.d.). A brief history of widgets. Online: http://www.niallkennedy.com/blog/2007/09/widget-timeline.html
2. Moraga, M.A., Calero, C., Piattini, M., & Diaz, O. (2007). Improving a portal usability model. *Software Quality Journal, 15*, 155-177.
3. Novak, J. D. & Cañas, A. J. (2008). The theory underlying concept maps and how to construct and use them. Online: http://cmap.ihmc.us/conceptmap.html