# Distributed Sensor-Driven Web Applications through Multi-device Usage Patterns

Heiko Desruelle and Frank Gielen

Ghent University – iMinds,
Dept. of Information Technology – IBCN, Ghent, Belgium
{heiko.desruelle,frank.gielen}@intec.ugent.be

**Abstract.** To access their computer applications and services, people tend to use an increasing variety of consumer electronic devices. Devices range from laptops and netbooks, to smartphones and tablets, and even interactive television sets. In the context of mobile applications, this ubiquitous revolution allows for various multi-device use cases and scenarios that are based on a user's dynamic usage patterns. In this paper we discuss how people can access an application using multiple devices, both in sequence as well as in parallel. Moreover, we elaborate on the technological opportunities and challenges for such multi-device enabled applications.

**Keywords:** Multi-device applications, dynamic usage patterns, ubiquitous web, HTML5.

## 1 Introduction

The increasing popularity of internet-enabled devices and technology is allowing people to access online content virtually anywhere, at anytime, and on any device. The available devices range from smartphones and tablets, to laptops and interactive television sets, etc. With the help of web technology, mobile applications can be built that are accessible by most of these device types (e.g., using web applications and widgets, or PhoneGap). Nevertheless, existing application solutions only partly succeed in providing end-users a convincing user experience. This issue is mainly due to the fact that most mobile applications are still tightly bound to the physical device on which they are being executed [1]. Existing application platforms barely take advantage of the diversity of devices owned by its users. The intended immersive and blended interaction aspect of such ubiquitous applications is thus mainly lost.

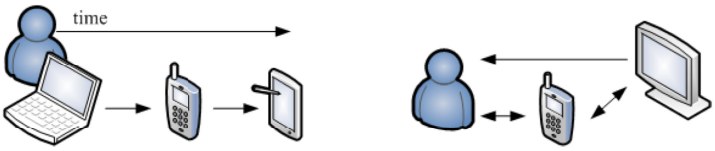
In this paper we introduce a web-based platform that aims to be a generic enabler for such multi-device applications. The proposed platform does so by relying on standardized technology in order to maximize its value and impact, both towards application developers as well as consumers. The remainder of the paper is structured as follows. In Section 2, we describe opportunities and related work for dynamic usage patterns that arise in environments with ubiquitous consumer electronic devices. Section 3 presents the developed application platform and discusses the main technological challenges for multi-device applications, which it aims to resolve. Section 4

H.-Y. Jeong et al. (eds.), *Advances in Computer Science and Its Applications*,
Lecture Notes in Electrical Engineering 279,
DOI: 10.1007/978-3-642-41674-3_3, 

presents the prototype implementation of the proposed platform for realizing an e-learning application use case. Finally, the conclusion and future work are presented in Section 5.

## 2 Background and Related Work

Online content can take various forms. It can range from documents, to presentations, webpages, videos, etc. Depending on the user's contextual setting, the typical usage patterns for accessing these resources may vary considerably. The user's context is dynamic over time. Its characterizing parameters include user preferences, as well as the available devices, the user's current location, etc. From this perspective, we identify two generic multi-device usage patterns for personal ubiquitous applications: sequential device usage, and parallel device usage (see Fig. 1).




(a) Sequential device usage (b) Parallel device usage

**Fig. 1.** Multi-device usage patterns in a ubiquitous computing environment

- *Sequential device usage.* Sequential usage patterns aim to smoothen the state transfer from one device to another and make it as seamless as possible. Based on this usage pattern, a user should, e.g., be able to start a session on a desktop PC and yet still be able to seamlessly pick up and continue this session with his mobile device when being away from home.
- *Parallel device usage.* This usage pattern aims to combine the interaction and presentation capabilities of multiple devices simultaneously. A real immersive experience should include the ability to distribute an application's user interface over multiple devices. Accessing content such as a video stream can, e.g., result in a rendering component being displayed on a television screen, whilst the stream's playback controls are shown on the user's mobile device.

Despite the prevalence of mobile application frameworks and operating systems, aiming to simultaneously combine features of multiple devices, solutions are often tight to a very limited set of devices, or a specific set of usage contexts. Existing work generally focuses on proprietary protocols and only supports specifically targeted platforms and vendors (e.g., the emerging interactive and connected TV platforms, which enable second screen applications via smartphone devices). As a counter, the Munin toolkit and Gibraltar framework aim to broaden this scope with a more flexible peer-to-peer design for distributed mobile applications over the Internet [2] [3].

## 3 Enabling Multi-device Usage Patterns

The platform described in this section aims to generically enable multi-device usage patterns. As depicted in Fig. 2, the proposed platform consists of a web application framework. Web technology has been selected as primary delivery channel based on our platform's goal to cover a broad range of devices (i.e., PC, mobile, tablet, TV, etc.). The application platform does so by leveraging standardized and widely adopted web technology such as HTML, CSS, and JavaScript. In result, applications can be accessed from virtually any web-enabled device's browser via their uniform resource identifier (URI).

In comparison to traditional web application frameworks, the proposed platform is able to automatically adapt its served applications' user interfaces (UI) based on the number and types of devices operated by the end-user. For this purpose, the application platform must be capable of dynamically enrolling requesting devices to a particular user session. The platform does so by generating QR codes (Quick Response code). This two-dimensional barcode is encoded with the active application's URI and a session identifier. Any device with a camera can in turn scan the code to automate its enrollment.

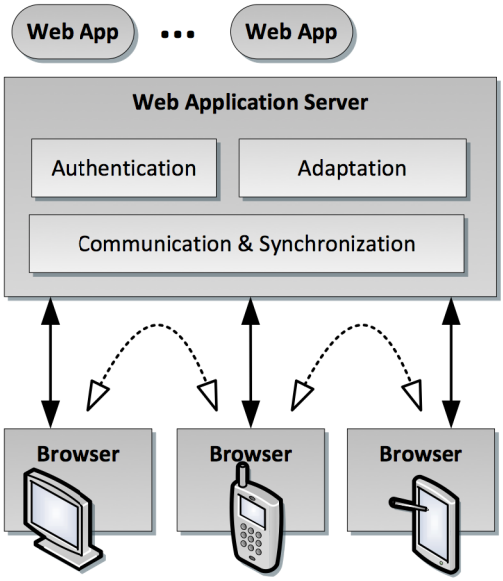



**Fig. 2.** High-level architecture for a web application framework, enabling multi-device usage patterns

The subsequent adaptation of an application's user interface is supported via two mechanisms, i.e., server-side and client-side adaptation. Server-side adaptation allows for the optimization of an application's user interface before it is sent to the requesting device. This type of adaptation enables developers to perform server-side

UI adaptations based on the user's contextual setting. To do so, the proposed platform provides access to detailed device feature and capability information, which are detected via user agent matching. Moreover, this step aims to minimize resource usage on the client's device (network, CPU, memory, etc.).

However, the usage patterns described in Section 2 are primarily characterized by dynamic session handovers. As devices are allowed to randomly join and leave active sessions, support is needed for on-the-fly UI adaptation as well. Hereto, the client-side adaptation mechanism aims to enable the adaptation of a UI that is already being rendered by one or more particular devices. Within the proposed platform, client-side adaptation relies on the at runtime manipulation of the application's DOM (Document Object Model) via JavaScript instructions.

By default, client devices communicate with the application server over standard HTTP (Hypertext Transfer Protocol). Additionally, a WebSocket communication channel is set up for efficient bi-directional communication once the initial HTTP request is closed. This way, application state changes can easily be propagated to all devices within the same user session. Moreover, server-initiated adaptation instructions can be pushed to a client after another device has joined or left the session.

## 4 Proof of Concept Implementation

A prototype of the proposed platform has been implemented as part of the webinos open source project [4]. The project consortium involves over 30 partner companies, including device manufacturers, service providers, universities, and research organizations. The prototype's server-side components are implemented on top of Node.js, a flexible and event-driven runtime for Google's V8 JavaScript engine [5]. In order for the prototype to cover a broad range of devices, the client-side requirements have been kept to a minimum and encompass all devices with at least an Internet connection and a browser supporting HTML5 WebSockets.

Moreover, a proof of concept e-learning application was implemented to showcase the multi-device capabilities of the proposed platform. The implemented application focuses on providing students with a blended learning experience when accessing educational content. The application's intended end-user experience is based on the two multi-device usage patterns presented in Section 2. The application provides traditional e-learning functionality by enabling users to navigate through various types of learning content. This content ranges from static text, to presentations, videos, etc. The added value of the proposed platform, however, is the built-in support for enrolling additional devices. When a secondary device joins the user's active session, the presentation and interaction components of the application are automatically distributed between the active devices.

Fig. 3 depicts the use case of a mobile device joining a session started on a television set. Scanning the displayed QR code starts the enrollment procedure. With the obtained application URI and session id, the mobile device opens a browser window and requests the application platform access. This request initiates the first adaptation phase, i.e., server-side adaptation. As elaborated in Section 3, the application platform

aims to optimize the returned user interface based on the a-priori knowledge of the client device's characteristics and capabilities. For the proof of concept implementation, this data is gathered by matching the browser's user agent string with the WURFL device description repository [6].

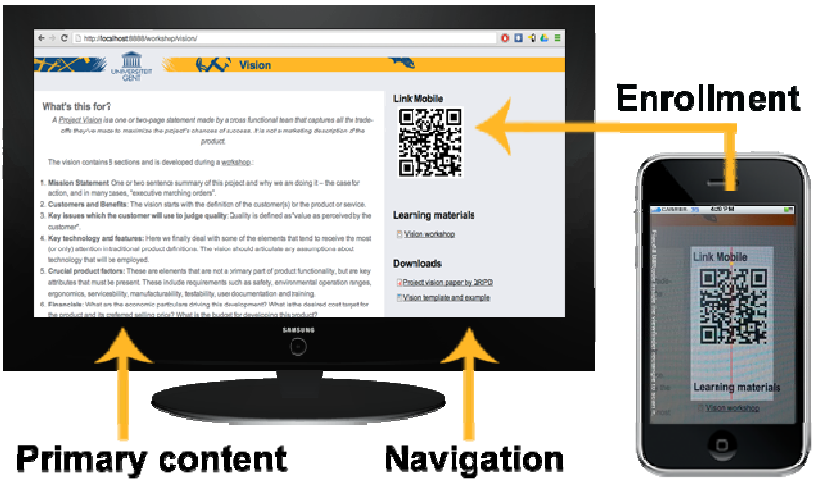



**Fig. 3.** Device enrollment and session synchronization via QR code scanning


**Fig. 4.** Distribution of the application's presentation and navigation modules between enrolled devices

Right after the enrollment, the television set is notified about the newly connected device. The application platform pushes client-side adaptation instructions to that device via their shared WebSocket communication channel. For the user's convenience, the television set is instructed to focus on primary content rendering and to remove the navigation bar (see Fig. 4). The client-side adaptation mechanism is implemented via jQuery's DOM manipulation API (Application Programming Interface). The API allows for the insertion and removal of specific DOM elements, as well as the modification of their contents and styling properties.

## 5 Conclusion and Future Work

In this paper, we've elaborated on the evolution towards multi-device usage patterns for accessing mobile and ubiquitous applications. We've presented the design for a web-based application platform, capable of automatically coping with the enrollment and synchronization of multiple devices owned by a particular user. Moreover, the platform supports the on-the-fly adaptation of its served application user interfaces. A platform prototype is implemented, as well as a proof of concept application for blended e-learning using multiple devices.

Future work includes a thorough quantitative study on the proposed platform's performance and scalability, as well as its contextual adaptability. Moreover, a qualitative user study is planned, which will be based on the prototype application presented in this paper. This evaluation data will serve to further validate and refine the assumptions made with regards to multi-device usage patterns.

**Acknowledgments.** The research leading to these results has received funding from the European Union's Seventh Framework Programme under grant agreement number 257103 (webinos project).

## References

1. Desruelle, H., Blomme, D., Gielen, F.: Adaptive mobile web applications: a quantitative evaluation approach. In: Auer, S., Díaz, O., Papadopoulos, G.A. (eds.) ICWE 2011. LNCS, vol. 6757, pp. 375–378. Springer, Heidelberg (2011)
2. Elmqvist, N.: Munin: a peer-to-peer middleware for ubiquitous visualization spaces. In: Proc. of the 1st Workshop on Distributed User Interfaces (DUI 2011), pp. 17–20. University of Castilla-La-Mancha (2011)
3. Lin, K., Chu, D., Mickens, J., Zhuang, L., Zhao, F., Qui, J.: Gibraltar: exposing hardware devices to web pages using AJAX. In: Proc. of the 3rd USENIX Conference on Web Application Development (WebApps 2012). USENIX Association, Berkeley (2012)
4. Desruelle, H., Isenberg, S., Lyle, J., Gielen, F.: Multi-device application middleware: leveraging the ubiquity of the Web with webinos. Journal of Supercomputing (2013)
5. Node.js, `http://www.nodejs.org`
6. WURFL – Mobile Device Database, `http://wurfl.sourceforge.net`