Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 56 (2015) 34 – 40

Procedia Computer Science

The 12th International Conference on Mobile Systems and Pervasive Computing (MobiSPC-2015)

From Apps to Liquid Multi-Device Software

Antero Taivalsaari[a]*, Tommi Mikkonen[b]

[a]*Nokia Technologies, FI-33720 Tampere, Finland*

[b]*Tampere University of Technology, FI-33720 Tampere, Finland*

Abstract

The recent history of software development has witnessed a battle between web-based software and native apps. At this point, native apps seem to be winning especially in mobile computing. In fact, the trend towards mobile apps seems to be strengthening with the increased popularity of “branded” apps. Such apps are rapidly replacing the use of traditional system applications in mobile devices. We argue that the transition to branded apps by no means predicts the demise of the Web as a software platform. However, there is still work ahead in turning the Web into a platform that can compete with dominant native platforms. At the same time, the focus in the industry is moving from PC and smartphone markets towards new device categories. In our view the industry needs to shift its focus from devices and device-specific apps to liquid software, i.e., multi-device experiences that allow people to use all of their devices seamlessly.

Keywords: Disruptive trends; multiple device ownership; branded apps; liquid software.

1. Introduction

In recent years, desktop and mobile software have evolved in different directions. On desktop computers, the most popular application for accessing content and applications on the Internet is the web browser. In mobile devices, in contrast, the majority of web content is consumed via custom-built native apps. This topic was discussed most visibly in a 2010 Wired magazine article[1] in which Chris Anderson and Michael Wolff proclaimed that “the (World Wide) Web is Dead”, based on two main arguments. The first argument was that the proportion of Internet traffic generated by web page downloads has decreased dramatically over the years compared to the traffic generated by video and music downloads and streaming. The second argument was that users soon will no longer surf

* Corresponding author. Tel.: +358 50 480 4620.
E-mail address: antero.taivalsaari@nokia.com

doi:10.1016/j.procs.2015.07.179

webpages with a web browser because, for the majority of web services such as e-mail, Facebook, Twitter, and Skype, the users prefer custom-built native applications over open, unfettered web browser access.

Anderson and Wolff's article was intentionally written in a rather provocative fashion. For instance, increased video and music traffic on the Internet surely by no means proves the "death" of the World Wide Web; in reality, the usage of the web browser and the number of web pages have also continued to increase dramatically. However, otherwise the predictions in the paper have turned out to be quite accurate. A recent Developer Economics report[2] confirmed that the trend towards app dominance continues to accelerate. Global worldwide app economy revenue is estimated to exceed $140 billion by 2016. HTML5 and web-based application development are gaining traction as well, but their progress has been overshadowed by the rapid increase in popularity of "branded" apps such as Dropbox, Instagram, Spotify, Twitter and WhatsApp. Such apps are rapidly replacing corresponding built-in applications in mobile devices.

In this paper we will look at the ongoing battle between apps vs. the Open Web. First, we will revisit some predictions that we made in our earlier papers[3, 4, 5], and re-evaluate the opportunities in turning the Web into a compelling software platform in the mobile space. Then, we turn our attention to the recent emergence of entirely new device categories and the challenges associated with multiple device ownership. We argue that the industry needs to shift its focus from individual devices and device-specific applications to liquid software, i.e., multi-device experiences that allow people to use all of their computing devices seamlessly and effortlessly.

2. The Battle of the Decade

The evolution of the Web has transformed the web browser from a document distribution tool to an increasingly encompassing environment that can serve as a general-purpose software platform as well[5]. Our prediction was that in the 2010's we would witness a major battle between two types of technologies: (1) native web apps and (2) Open Web applications that run in a web browser or other standards-compliant web runtime environment. This "battle of the decade" would determine the future of the software industry, as well as the future of software engineering research, for years to come[4].

It is still too early to declare the victory of the apps and proclaim the end of the battle of the decade or the death of the Web more broadly. On the desktop computing side, as opposed to mobile computing, the popularity of installed applications seems to be waning compared to the use of web-based services. An example of this is the ongoing transition of the Microsoft Office from a desktop-bound set of applications to a web-based service suite (http://office.microsoft.com/). In general, the most popular application in desktop computers today is the browser, and the majority of activities on desktop computers today are performed using a browser instead of conventional, specialized binary applications. This is in striking contrast with mobile devices in which the majority of content is consumed via custom-built apps.

It is also important to note that today's native apps are not like the native applications of the earlier desktop computing era. Nearly all commercially significant mobile apps today utilize cloud-based resources available online. In that sense, native apps and cloud-based software technologies and economies are inseparable and inextricably bound to each other.

3. From Apps to Brands

In a 2005 book[6], "The World is Flat", Thomas Friedman analyzed globalization in the early 21st century. The title of the book is a metaphor for viewing the world as a level playing field where all competitors have an equal opportunity from the commercial viewpoint. The title also alludes to the perceptual shift required for countries, companies, and individuals to remain competitive in a global market where historical and geographical boundaries are becoming increasingly irrelevant.

The world is flat metaphor is especially applicable to those industries in which distribution costs are low or virtually non-existent, such as the software industry. For instance, in online software game development there is virtually infinite upside potential. Even individual game developers (and not just large game studios) can make tens or hundreds of millions of dollars if they manage to create a game that becomes a hit and generates tens or hundreds of millions of downloads or in-app purchases. In a flat application economy, success is all about attention and publicity. While there are plenty of “long tail” (niche) opportunities for large masses of application developers, usually only a handful of popular applications and developers bask in the limelight, become dominant and take most of the profits as well.

The consequences of the flattening application economy are very visible in the mobile app ecosystem today, where the dominant trend today seems to be the “brandification" of apps. By brandification, we refer to the concept where traditional system apps such messaging, notepad, camera, photo gallery or the music player (provided by the operating system vendor) are effectively being replaced with custom-built apps that are tied to commercially popular third-party “brand leader” services such as WhatsApp, Twitter, OneNote, Instagram, Flickr or Spotify. As a side effect, the consumers’ data is increasingly stored in service-specific “silos” that are separate from each other. Many of these apps and services started out as fairly simple garage projects. However, once apps reach a large enough user base, they become serious business, as witnessed by Facebook’s recent acquisition of WhatsApp.

Most of these brand leader apps are built on top of native mobile operating systems, especially iOS and Android. There are purely web-based versions of many of these apps as well, but so far their popularity has been overshadowed by the native versions. The history could have taken a different path, as we will examine next.

4. The Open Web Promise

Following the Open Web principles laid out in the Mozilla Manifesto[7], web applications should be built on technologies that are open, accessible and as interoperable as possible, and should run in a standards compatible web browser without plugins, extensions or additional runtimes. In December 2010, Tim Berners-Lee – the inventor and founder of the World Wide Web – published an article in which he called the trend towards custom-built native web apps “disturbing”, because that trend divides information into separate content silos that are isolated from each other[8]. Such content is off the Open Web, and usually under the control of an individual company. Typically, one cannot bookmark, copy or e-mail a link to such a page using a web browser. Rather, one must explicitly download, install and use (and later upgrade) a vendor-specific app from a vendor-specific app store for each device platform to access such content.

More recently, Tim Berners-Lee renewed his plea for the continued support for Open Web principles in his video speech celebrating the 25th birthday of the World Wide Web (http://www.webat25.org/). In addition to highlighting the important of keeping the Web universal, royalty-free, open and decentralized, he was especially eager to promote the creation of a high-performance open architecture that would run on any device, without falling back into proprietary technologies or content silos.

In principle, Open Web applications have significant benefits. For instance, they require no installation or manual upgrades, and they can be deployed instantly worldwide. The Open Web principles allow application development and instant worldwide deployment without middlemen, distributors, or platform-specific app stores. In principle, conventional binary applications are at a major disadvantage in comparison to their web-based counterparts.

In practice, a number of obstacles have hindered the development of full-fledged, truly interactive web applications. While web technologies such as HTML5 and CSS3 promise plugin-free animation, video and typography, many key standards are incomplete and inconsistently implemented in browsers. These obstacles have been especially apparent in the mobile context.

5. The Apps vs. the Web Paradox

There is a paradox in the software industry that continues to puzzle us. In spite of all the hype on apps in mobile computing, desktop computing seems to be headed in an entirely different direction. On PCs, people rarely install new binary applications nowadays; the average number of applications that the users install on their new PCs these days is dramatically lower than the average number of applications installed 10-15 years ago. This is largely because of the shift from traditional installed applications to browser-based web services; the majority of activities on desktop computers today are performed using a web browser instead of conventional binary applications.

The ultimate manifestation of this "webification" trend is the Google Chromebook (http://www.chromebook.com/) – a purely browser-based thin client laptop in which all the data is stored in the cloud accessed via an internet connection. From the user's perspective, a Chromebook does not have any conventional binary applications at all. Rather, all the applications are web applications that are downloaded dynamically from the Web. Furthermore, no data backups or explicit application upgrades are required, since all the functionality stays in sync with the cloud. In the mobile space, the push towards similar web-based platforms such as Mozilla's Firefox OS and Nokia's Cloudberry platform[9] failed to receive any significant traction.

Our expectation is that the apps vs. Web divergence between mobile and desktop computing will not continue indefinitely. However, in the next five to ten years, native and web-based application economies are likely going to co-exist and complement each other. For instance, native application usage will likely spread rapidly to new embedded domains such as automotive and home control systems as the focus in the software industry shifts to new types of devices. Correspondingly, as summarized in a Q3/2013 Developer Economics survey[10], web-based applications will increasingly serve as "common denominator" technology for situations in which the developers do not want to write several versions of their applications for different types of devices or operating systems. In general, specific application ecosystems as well as the battle between apps and the Web may become much less relevant since the broader industry trends will increasingly require interoperability between a wide range of devices and platforms. These trends reflect changes in the industry towards multiple device ownership and the end of the dominant era of PCs and smartphones.

6. The Next Paradigm Shift

Today, the average consumer in developed markets has two primary computing devices: a personal computer (usually a laptop) and a smartphone. Additionally, after Apple's successful launch of the iPad and the subsequent proliferation of inexpensive Android tablets, many people carry a third device: a web tablet. While it may be tempting to think that today's PC and smartphone centric world will simply be extended with yet another device, in reality the number of network-connected devices that people use in their daily lives is expanding much more dramatically. We will quickly move to a world in which people will use dozens of devices in their daily lives: laptops, phones, tablets and "phablets" of various sizes, game consoles, TVs, car displays, digital photo frames, home appliances, etc. – all connected to the Internet.

The trend towards multiple device ownership was reflected clearly in a survey in which over six thousand developers worldwide were surveyed to study their platform preferences, developer attitudes, revenue models and tools[10]. According to that survey, developers are actively looking into alternative target devices (e-readers, TVs, set-top boxes and game consoles) above and beyond desktops, smartphones and tablets in their future efforts. In summary, we are at a tipping point with connected devices, entering a new era of multiple device ownership. This new era will dramatically raise the expectations for device interoperability, implying significant changes for software development as well.

7. Towards Liquid Software

As pointed out by Dearman and Pierce back in 2008, the transition to a world with multiple device ownership is rife with problems[11]. For instance, the need to synchronize and back up data between multiple devices is a major hassle. Likewise, the requirement to manually specify e-mail accounts, web bookmarks, RSS feeds, and other personal preferences and settings for each individual device can be painful. The need to explicitly install and then later frequently upgrade the applications for all the devices separately can also be time-consuming. Incompatibilities between devices abound; applications intended for one platform do not run on other platforms; applications bought from one app store cannot be installed in other devices. These problems get worse rapidly as the number of devices in a person's daily life grows.

Systems such as Apple's iCloud (http://www.icloud.com) and Google Sync (http://www.google.com/sync) are paving the way for automatically synchronized devices. However, these systems are limited to devices supporting the same native ecosystem; in other words, they lock the users in a single platform "silo". Furthermore, these systems do not yet provide seamless experiences and transitions across devices. Ideally, when the user moves from one device or screen to another, the users should be able to continue doing exactly what they were doing previously, e.g., continue playing the same game, watching the same movie or listening to the same song on the other device. This type of "liquid" usage of software is not generally supported yet, but such user interface concepts have been presented before[12, 13].

We believe that multiple device ownership should be as casual, fluid and hassle-free as possible. A central aspect of a truly casual computing experience is the ability to move fluidly from one device to another. By liquid software, we refer to an approach in which applications and data can flow from one device or screen to another seamlessly (see Figure 1). In such setting, the user can roam effortlessly from one device to another; device management chores such as backups, application installation, application upgrades, restarting the recently used applications, account migration, copying settings across devices, or other similar activities that burden the daily lives of users today should be things of the past.

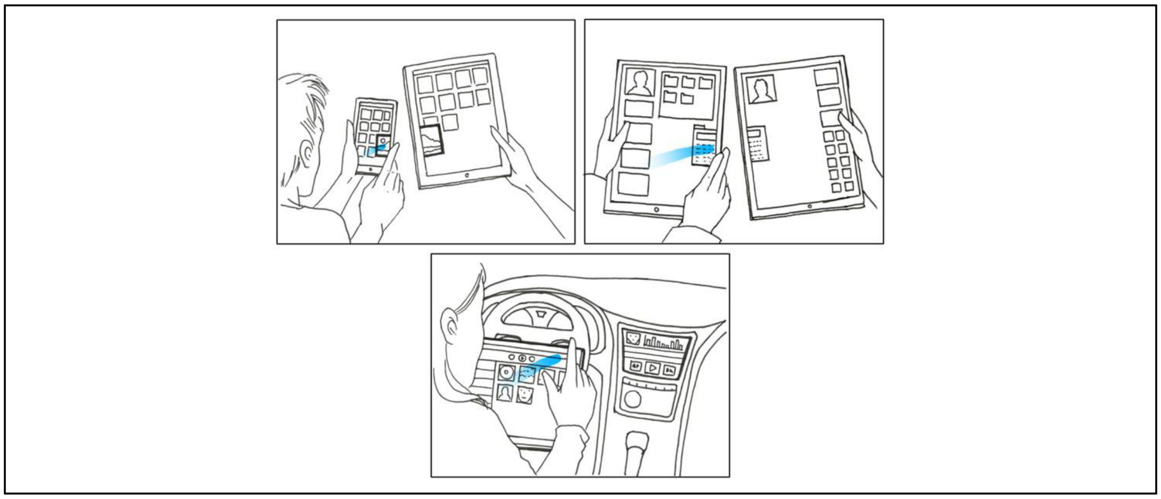

Fig. 1. Liquid Software Illustrated.

The recently published Liquid Software Manifesto[14] laid out the general principles and requirements for liquid software based on seminal work in the 1990's by Hartman, Manber, Peterson, and Proebsting[15, 16]. The core requirement of the manifesto is the ability to roam effortlessly between the different devices that a user has, with minimum maintenance or device management hassles. While liquid software may still seem like science fiction, the technical building blocks are already in place, with a number of architectural choices and dimensions[14].

Although liquid software is likely going to emerge first in the form of individual applications spanning multiple devices, the ultimate manifestation of the vision would be a "Liquid OS" – a multi-device operating system in which all the user sessions, including the state of each open application, can be transferred dynamically from one computer or device to another, or simultaneously used from multiple computing devices. Similarly, all the user's data would be accessible on all devices. The OS would provide system apps with built-in state synchronization, as well as developer APIs for writing applications whose state can be synchronized across devices with minimum development hassles.

Granted, there are still challenges with liquid software. Many of the limitations associated with multi-device usage arise from the broad variety of devices not only in terms of processing power and memory capacity, but especially different screen sizes, input mechanisms and usage patterns. With screens ranging from tiny wristband displays to wall-sized screens, and input mechanisms ranging from T9 keypads and remote controls to touch displays and QWERTY keyboards, one user interface solution simply does not fit all. This makes it difficult to migrate live applications from one device to another[17]. Although design approaches such as responsive web design[18] make it easier to create software that automatically adapts to different types of target devices, the fact is that applications intended for a specific type of a device do not necessarily make sense at all on other, different types of devices.

In our earlier work, we assumed that HTML5 would emerge as the dominant unifying platform for multi-device usage, serving as a common denominator and "ecosystem of ecosystems" that would bridge the gaps between different native desktop and mobile operating systems. The recent popularity of native apps has mostly invalidated this assumption, at least for now. Presently, it seems that liquid software capabilities will first emerge in native operating systems (Android, iOS, Windows) as an extension to their current cloud-based synchronization capabilities (Google Sync, iCloud). For instance, the "handoff/continuity" features introduced in Apple's recent Yosemite OS are a clear step in this direction.

8. Conclusions

In this paper we have revisited the recent battle between web-based software and native applications or "apps". At this point native apps seem to be winning especially in mobile computing. Furthermore, the trend towards mobile apps seems only to be strengthening with the increased popularity of "branded" apps that are rapidly replacing the traditional system applications in mobile devices. Nevertheless, it is still too early to declare the victory of the apps and proclaim the death of the web-based software development. Rather, in the next five to ten years, native and web-based application economies are likely going to co-exist and complement each other.

In the larger scheme of things, the battle between apps vs. the Web is becoming somewhat irrelevant as we are at a cusp of yet another paradigm shift – the dominant era of PCs and smartphones is coming to an end. So far, standalone devices have been the norm, and software has been attached primarily to a single device. We believe that in the computing environment of the future, the users will have a considerably larger number of connected devices in their daily lives than today. Unlike today, no single device will dominate the user's digital life. We will enter the era of multiple device ownership.

In this paper we reiterated the need for a truly liquid multi-device software environment. By liquid software, we refer to a multi-device software experience that can seamlessly and effortlessly "flow" from one device to another. Liquid software entails a virtualized but personal computing experience that is independent of any particular device or OS platform, allowing the users to seamlessly roam and continue their activities on any available device or computer. Although liquid software may still seem like science fiction, the technical ingredients and enablers for realizing the vision are already largely in place. The vision itself can be implemented either using native or web-based software. The seeds for the paradigm shift towards multiple device ownership have already been planted, and it will be very interesting to see which route the future will take. In summary, paraphrasing Mark Weiser[19], we believe that by the end of this decade multi-device usage will become so seamless and ubiquitous that "it will weave itself into the fabric of everyday life until it is indistinguishable from it". This is simply how computing devices shall work from now on.

References

1. C. Anderson, and M. Wolff, The Web is Dead: Long Live the Internet, Wired, Sept. 2010, pp.118-127 & 164-166.
2. VisionMobile, Ltd., Developer Economics Q1 2014, State of the Developer Nation, February 2014. URL: http://www.visionmobile.com/product/developer-economics-q1-2014-state-developer-nation/.
3. T. Mikkonen and A. Taivalsaari, Reports of the Web's Death are Greatly Exaggerated. Computer, May 2011, pp.30-36.

4. T. Mikkonen and A. Taivalsaari, Apps vs. Open Web: The Battle of the Decade. Proc. MSE'2011 (Santa Monica, California, USA, October 27, 2011, pp.22-26.
5. A. Taivalsaari and T. Mikkonen, The Web as an Application Platform: the Saga Continues. Proc. SEAA'2011 (Oulu, Finland, August 30 – September 2), 2011, IEEE Computer Society, pp. 170-174.
6. T. L. Friedman, The World is Flat: A Brief History of the Twenty-first Century. Farrar, Straus and Giroux, 2005.
7. Mozilla, Inc., The Mozilla Manifesto. URL: http://www.mozilla.org/en-US/about/manifesto/.
8. T. Berners-Lee, Long Live the Web: a Call for Continued Open Standards and Neutrality, Scientific American, Vol. 303, Nr 4 (December), 2010, pp.56-61.
9. A. Taivalsaari and K. Systä, Cloudberry: HTML5 Cloud Phone Platform for Mobile Devices, IEEE Software, July/August 2012, pp. 30-35.
10. VisionMobile, Ltd., Developer Economics Q3 2013, State of the Developer Nation, July 2013, URL: http://www.developereconomics.com/reports/q3-2013/.
11. D. Dearman and J.S. Pierce ,"It's on My Other Computer!": Computing with Multiple Devices. Proc. CHI'2008 (Florence, Italy, April 5-10), 2008, pp. 767-776.
12. M.A. Nacenta, D. Aliakseyeu, S. Subramanian, and C. Gutwin, A Comparison of Techniques for Multi-Display Reaching. Proc. CHI'2005 (Portland, Oregon, USA, April 2-7), 2005, pp. 371-380.
13. S. K. Kane, A. K. Karlson, B. R. Meyers, P. Johns, A. Jacobs, and G. Smith. Exploring cross-device web use on PCs and mobile devices. In Proc. Human-Computer Interaction–INTERACT 2009 (pp. 722-735). Springer Berlin Heidelberg.
14. A. Taivalsaari, T. Mikkonen, and K. Systä, Liquid Software Manifesto: The Era of Multiple Device Ownership and Its Implications for Software Architecture. Proc. COMPSAC'2014 (Västerås, Sweden, July 21-25, 2014).
15. J. J. Hartman, U. Manber, L. L. Peterson, and T. A. Proebsting, Liquid Software: a New Paradigm for Networked Systems. Univ. of Arizona Tech Report TR 96-11, 1996.
16. J. J. Hartman, P. A. Bigot, P. Bridges, B. Montz, R. Piltz, O. Spatscheck, T. A. Proebsting, L. L. Peterson, and A. Bavier, Joust: a Platform for Liquid Software. Computer, April 1999, pp. 50-56.
17. D. Thevenin and J. Coutaz, Plasticity of User Interfaces: Framework and Research Agenda. In M.A. Sasse, C. Johnson (eds), Human-Computer Interaction – Interact'99, IOS Press, 1999, pp. 110-117.
18. E. Marcotte, Responsive Web Design. A Book Apart, 2011.
19. M. Weiser, The Computer for the 21st Century. Scientific American, September 1991, pp. 94-104.