The Road to Smart Spaces: Convergency of Consumer Electronics with a Change of Content Management and Application Design Principles

Alexey Koren

State University of Aerospace Instrumentation
Bolshaya Morskaya 67, Saint-Petersburg, Russia
Alexey.Koren@extendedmobile.net

Sergey Balandin

Nokia Research Center
Itamerenkatu 11-13, 00180 Helsinki, Finland
Sergey.Balandin@nokia.com

Abstract

Methods and solutions for efficient processing reuse and reasoning over user's data has gained a lot of attention in the last two years and became the key aspect of application design. It is especially critical nowadays when the volume of information increases as rapidly as its value. In this paper we analyze how the basic design principles of dealing with information and methods of it representation in the user services has changed over the time for the areas of web-based applications, mobile applications and PC software. It is notable that all of them are moving in one direction and faces similar problems. Based on this analysis we conclude that likely the content management and application design principles for the above listed areas soon will converge under the same common principles. Then, we argue very painful question for every convergent sphere - how to avoid platform-dependency and provide maximum functionality and information for every participant without introducing significant redundancy. Solution should also be sensitive to user's context and personal requirements. Finally we observe an architecture which is intended to solve most of the raised question regarding information exchange and its representation on different platforms. It is based on concept of self-organizing Smart Spaces, where information is stored in dedicated semantic information structure. In this architecture widgets play role of agents which connect to Smart Spaces, perform reasoning over Space's information and use it in providing services for end-user. We conclude that the latest trends for the addressed industries lead them to adaption of the Smart Space principles, as the overall framework of the converged industry.

Index Terms: device convergency, application design principles, smart spaces, content management.

I. INTRODUCTION

Application design and information management models have been changing continuously in the computer systems for the whole time of existence. It happens because of a significant progress in the device computation power and amount of information produced and consumed by them, development of network infrastructure and many other factors. As total amount of information is growing rapidly, the task of search and reasoning over it becomes more and more difficult. However in most cases user does not need to have search and reasoning performed over all of the information available in the Web. Even more, for most tasks this approach is not user-friendly, over-complicated, redundant and requires extra computational power, which affect systems performance and even ecology through energy consumption increase. As a result, the ability of a system to perform computations in the given user context becomes more valuable as well as ideas of ubiquitous computing becomes more and more popular [1].

Moreover nowadays not only amount of information is increasing but also many associations, dependencies and duplication of the portions of information appear. So we can notice that the information management systems in a very close future should not only be able to handle increasing volumes of information, but have to deal with completely new structure of the content. This requires from applications to adopt the new principles of information management, have a strong toolset for data aggregation, reasoning, linking, representation, re-use, etc. For example, for aggregation of information or services there are many mash-ups in the Web, which integrate a few web-services into one larger and value adding tool. But despite of good design, performance and usability, all these solutions are more like fixes applied to the already made architectures, which are not intended for solving them naturally, e.g. in [2]. This makes us believe that revolution in knowledge management technologies will start soon.

While trying to foresee appearance of the new paradigm we have analyzed three main service provision domains. In section 2 we discuss PC solutions that represent still the largest domain for software services. In sections 3 and 4 we discuss two rapidly developing and evolving domains such as mobile and web-based services.

The first intention was to analyze these domains separately, which would lead us to determining three independent results. However, after deeper study on possible correlations we found that these areas have quite similar problems and follow the same trends which we believe lead them towards the same direction. This is not so much surprising if one remember predictions about the convergency of different classes of software and hardware into a super-class. Borders between different classes of solutions are already almost disappeared, e.g. it is impossible sometimes to distinguish 3G-enabled ultra small netbook (PC domain) from the high-end smartphones with qwerty-keyboard (mobile domain). Similar can be seen with the web-based, PC and mobile software and services. For example, could we clearly distinguish whether Gmail Offline is a web-based service or desktop application? Should we classify Web Runtime widgets [3] as a solution of mobile or web world? So we can see that these three industries are heading to complete convergency and we believe that the future application design and content management will follow this trend. More detailed analysis on the subject is given in section 5.

Finally in section 6 we discuss the most promising ways to organize content management in the new world. We believe that the answer is Smart Spaces concept, which is currently an emerging topic discussed in a number of research papers [4, 5]. In this section we analyze the current trends, achievements and propose ideas for further development, taking into account needs of existing domains and their abilities to further evolution. The paper is concluded by the summary of the main finding and discussion points followed by the list of references.

II. PC SOFTWARE

Desktop software is the oldest and most conservative domain among those under consideration. Historically there it provides two basic patterns of information management: files and databases.

File-based approach was the easiest way to organize and share information - users worked with disks and FTP for information exchange. Databases had import/export information and use socket connection for communicating with clients that form SQL queries to access particular part of information. Information was gathered on ftp-servers or in big databases, however, practically there wasn't any possibility for efficient work with it. Search over information possible only in the case when user was sure that file with particular name (or part of the name) exists somewhere in the system.

At the early stage desktop applications mostly had a form of compiled code and worked with proprietary data formats. Common formats was used mostly not for sharing or information reuse but as method of saving developers' and architects' resources.

With significant growth of the network infrastructure size and amount of data generated by users, the need for the efficient information management mechanisms that allow information reuse and sharing is rising continuously. The Advanced Programming Interfaces (APIs) started to appear for many applications, code started to be distributed not only as application but also as shared libraries. In software development "application constructors" become popular which allow user to gather application from modular pieces and pre-defined components (Java Beans, Swing, .NET) and use modules once created by user in future applications.

Nowadays many users see standard desktop software as old-fashioned inflexible solutions, they don't want to do installs anymore, preferring small and light-weight applications solving particular user problem called Widgets [6] (Windows/KDE/GNOME Desktop Widgets, etc) or just use advanced web-services instead. The web-based software domain is covered in more details in chapter 4.

Because of totally different platforms for widget, so far there were practically no attempts to create common API or at least set up some kind of data exchange between single widgets. However, ideas that this is necessary were proposed very a lot of times [7, 8, 9].

Personal information storages such as hard-drives or small network disks became inexpensive for users and it poses new problem: how to organize huge amount of personal information and manage it properly? It is well known problem that sometimes it is easier to search Web for some particular document and download it once again than find it on local disk. There were certain attempts to solve this problem, from simple text desktop search to Semantic Desktop engines [10]. Most recent works are trying to apply Web search principles to a desktop search [11].

III. MOBILE APPLICATIONS

The mobile software at its very beginning was organized pretty much like desktop software: simple, inflexible applications with necessary install procedure. Most of them were tools and casual games. In many proprietary OS, users had access only to pre-installed applications without any possibility of getting or creating new software. Due to OS constraints there was no information exchange amongst applications at all, even at the file system level. No common usage or sharing was possible.

Changes started with rapid development of mobile access to the Web. It gave possibility for software developers to use the main advantage of mobile devices - their mobility. The main problems they faced were that the regular web sites are too heavy and slow for resource-restricted mobile devices. Also they were designed for use with the large screens so standard browsing on the small devices wasn't handy. PDA and WAP versions didn't solve the problem efficiently because of limited functionality compared to the full web versions. On the other hand creating mobile software in native code or Java with rich functionality was complicated. Huge step to popularizing mobile web was wave of technologies, which make development process simpler, integrate and adapt web services and information for mobile applications. Support for some of them is already discontinued (e.g. WidSets), but some are maturing and improving actively (e.g. Web Runtime). The key idea of these platforms is to grab only information from the original Web and replacing its representation with the user interfaces created directly for mobile screen and input methods. The main basic technologies for such

platforms are open Web standards for information management. That was expected because Web standards were created taking into account need of cross-platform information exchange. WidSets service is a good example of such architecture. WidSets service connects mobile devices to rich web applications and extracts from it as it is illustrated by Figure 1.

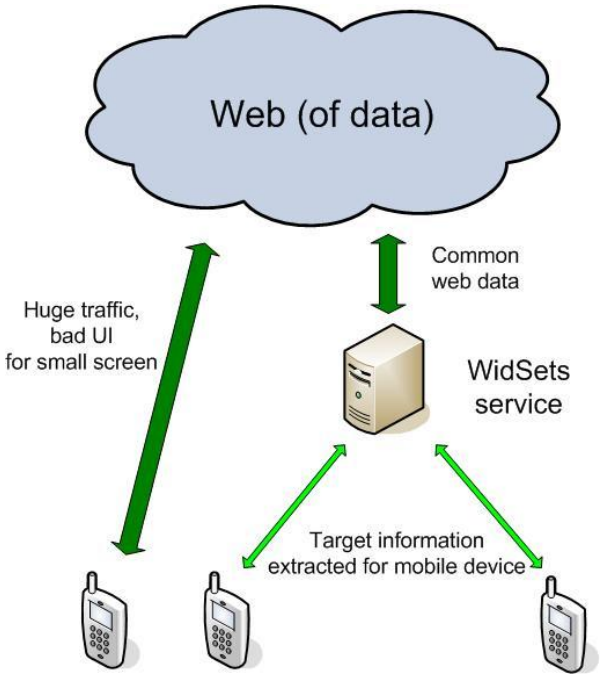

Figure 1: The historical WidSets architecture

The main disadvantages of WidSets implementation were that it was too centralized and relied on parsing web-pages with regular expressions web-pages instead of using semantic approach to the information distribution. It is possible to say that WidSets was a hack that shows problems in information management in convergent web and mobile domains. And as any other temporary fix it soon had to be replaced by the more generic and optimized solution that takes information management on to the new level.

Further study of the mobile software domain reveals another trend – context-dependency. The main ideas here are location-based services (LBS) and near-field communications (NFC). This kind of services provide user only with information valuable for a given geographical location, i.e. in the area around user (or any other selected area), or with information about active objects inside the given range. Huge popularity of these services and technologies show that this simplification is very suitable for users as it plays role of filter, which extracts from huge database of generic info only parts related to user in geographical sense.

IV. WEB SERVICES

Originally Web domain was not counted as a part of software domain. It consisted only of static pages sometimes filled with scripts which were mostly used for simple animation of page's content. But even on this stage there were attempts to create universal cross-platform data formats, link information objects between each other and separate data from its representation. Most popular language for this is RDF [12].

Logical transition from "Web 1.0" to "Web 2.0" emphasized and improved these trends and also introduced new patterns of information interaction. A few of the most dominating trends:

1) Need for information aggregation and sharing. It concerns generic user information as well as specific, for example, authentication data - OpenId which implements Single Sign-On concept in Web domain. Aggregation is actively developing both in data domain (RSS, Atom protocols, APIs) and functionality (different mash-up services).

2) Appearance of web-services with extremely high functionality, competing with desktop applications. For example, Gmail and Google Docs are pretty good replacement for desktop office packs. Main disadvantage of such services – a constant need for Internet connection, but there is progress in this direction and probably it will be fixed in a near future.

So we can see that borders between desktop and web-based software are disappearing. One can remember that 15 years ago there were no borders at all between these domains as they were considering completely different with no interactions or dependencies. And now after opening a shortcut on desktop user sometimes cannot distinguish desktop application from bookmark created in Google Chrome. Widgets are also very much in use in web domain and there are a lot of initiatives to introduce common standards for them [13, 14].

There is the same indeterminacy with Web Runtime widgets which can be seen as half mobile applications and half web resources. For more traditional web-developers, the manufacturers provide API for using the device platform features in web-site functionality. Usually it comes as JavaScript libraries that give access to platform UI elements or platform capabilities like GPS or camera. Examples are iPhone and Nokia Platform Services.

Context-dependency that was noticed in mobile domain also presents in Web. It is all kind of LAN resources. Trend for inner corporate portals or sets of resources called Intranet became stronger with appearance of Enterprise 2.0 - this concept attempts to apply Web 2.0 basic ideas for corporate domain: private social networks, service aggregators, semantic search engines, etc [15].

V. ANALYSIS OF THE COMMON TRENDS

By taking into account the above listed thoughts, we can notice that there are a few common trends and problems faced by the domains of study. Historically, desktop, mobile and web software belonged to different domains clearly separated from each other. But now it is clear that they are moving towards the same direction. As well as problems are common, architectural and technical solutions for information management have to converge: cross-platform and cross-domain. Below we provide an aggregated list of trends for domains under consideration:

1) Separation of data and its representation, in connection to ability of exchanging, managing and aggregating information without platform dependencies;

2) Reuse of information and/or application functionality;

3) Context-based services;

4) Cross-platform open formats and solutions.

Based on these trends lets draw an idealistic picture of architecture that could potentially solve most of the problems described above for the given software domains. First consider information representation. It is essential that information should be stored and being distributed in open formats. Family of formats most suitable for this seems to be Semantic Web formats like OWL, RDF, etc. Using of them solves problem of information exchange as well as gives potential compatibility with Semantic Web linking and reasoning mechanisms. Next question is where the information should be stored? There are two orthogonal concepts

here. "Information Cloud" concept says that all information should be placed on huge data arrays somewhere in big player's cloud (e.g. Amazon, Google, etc). Another approach encourages users to store and share their information under personal control, when almost every user acts like small database where his/her information stored and other users can access it by common protocol. Despite to the fact that drivers of each approach are in deep opposition because of philosophy and business interests, technically there is no issues to have architecture that can serve both information storage models. The same protocol and interfaces could make huge Cloud with millions of users hosted inside and terabytes of information indistinguishable from single user who shares few megabytes of personal information in terms of information management. This way a few sources of information (Clouds, Spaces, etc) form a personal context of every user. It could be some private (like Smart Home agent) or shared sources aggregated in one place to satisfy individual needs of information and/or services. It is very important that information in these sources is platform-independent, preferably have semantic format, e.g. in OWL [16], and any device could connect to it and get or post some information. Example is shown Figure 2.

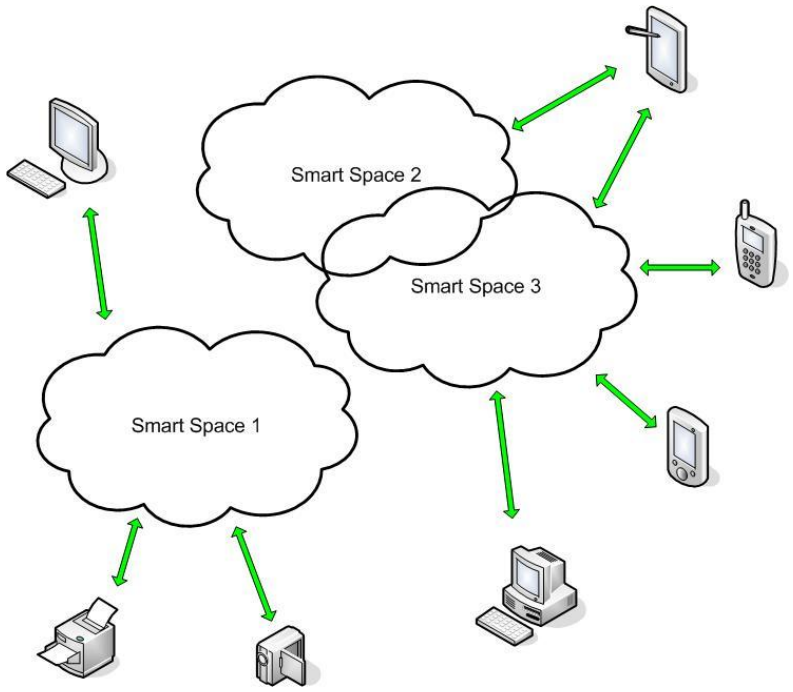

Figure 2: Local information sources accessible from any device

The concept that puts together all the points discussed above was already presented under name Smart Spaces. The Smart Spaces stay for context-aware networks with joint source of information in platform-independent format. It utilizes open web standards, Semantic Web reasoning mechanisms and provides possibility for every person or robot to use shared information and add new information by merging it into the semantic graph and subscribe to all updates of information in given space as illustrated by Figure 3 [17].

The Smart Spaces work efficiently with different OS and devices (types of devices) because of their focus on information distribution or service distribution through information-based interfaces. A user with any device including PC, mobile phone or even toaster could connect to the space by common shared interface and acquire information in open format, interpret this data on his own, make personal decisions based on parts of information relevant to him and share new information to all space members, thereby evolve with all main trends principles listed above.

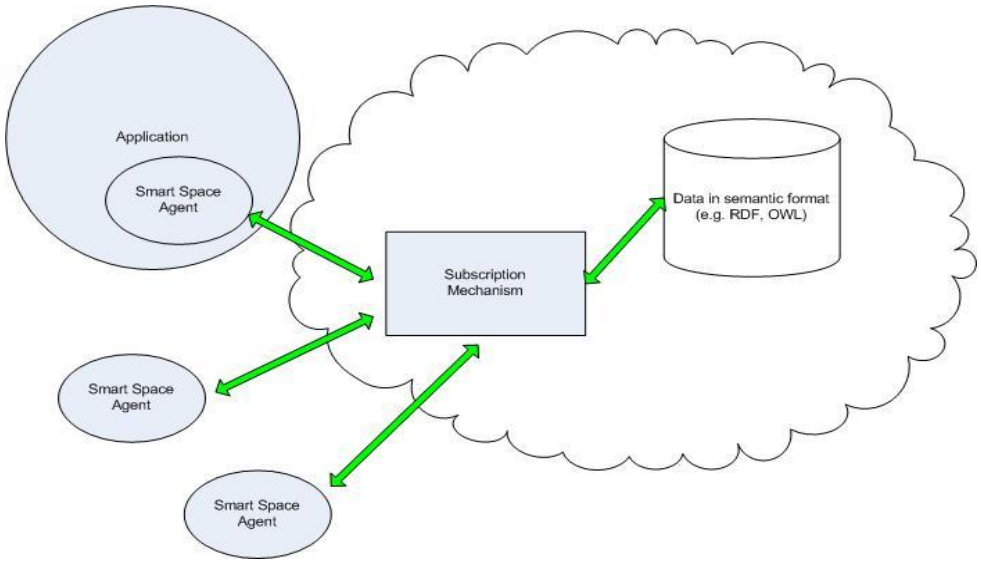

Figure 3: Inner architecture of a Smart Space

VI. CONCLUSION

In this paper we discuss the Application design and content management principles in web-based, PC and mobile software solutions and it has been shown that these three dominant software domains are moving in the same direction and their convergency is a reality of today. As a consequence we believe that very soon there will be no strict boarders between solutions for desktop PCs, netbooks, smartphones, web-based services and even applications for the embedded software in the various consumer electronics devices (e.g., TV or music station). The consumer electronics and embedded software is very young but rapidly emerging area. Still there is not much trends and patterns of information management to discuss in scope of this paper, but potentially consumer electronics is one of the most promising domains for implementing next phase of Smart Spaces solutions. We see solutions of the future to be easily portable, device and platform independent, highly granular and based on the open standards. These Agents will generate and consume from the localized context-sensitive platform-independent sources of semantically aggregated information, which also known under term Smart Spaces. We have discovered that the trend to organize communication via Smart Space model is not a phenomenon of the mobile industry as it was suggested before by some authors, but similar could be denoted also in the web and PC worlds. It is still remains to be an open discussion of how to better organize information sources, i.e. as independent small pieces which communicate with neighbors for establishing rich information context in given area or as Giant Global Graph which aggregates data from whole Web and is stored on the Cloud, but our position is that the first approach is more scalable and better reflects user interest, e.g. preserves privacy and content ownership feeling.

ACKNOWLEDGMENT

The authors would like to thank FRUCT (Finnish-Russian University Collaboration in Telecommunications) program [18] for support and assistance provided and also would like to acknowledge importance of the financial help provided by Nokia and Nokia Siemens Networks.

REFERENCES

[1] Some computer science issues in ubiquitous computing, Mark Weiser, Communications of the ACM, Vol. 36, No. 7 (July 1993), Pages 75-84.

[2] Rapid Prototyping of Semantic Mash-Ups through Semantic Web Pipes. Danh Le-Phuoc, Axel Polleres, Manfred Hauswirth, Giovanni Tummarello, Christian Morbidoni. WWW 2009, April 20–24, 2009, Madrid, Spain. ACM 978-1-60558-487-4/09/04.

[3] Nokia Inc. Nokia. s60 goes beyond web browsing with web run-time and widgets.http://sw.nokia.com/id/e67c057a-7549-4edb-b9f1-193ab27cce71/S60_Web_Run_Time_and_Widgets.pdf {Retrieved 03-08-2009}.

[4] Dynamic, Localised Space Based Semantic Webs. Ian Oliver, Jukka Honkola, Jurgen Ziegler. Nokia Research Centre.

[5] Information spaces as a basis for personalizing the Semantic Web, Ian Oliver. Nokia Research Centre.

[6] An introduction to Widgets with particular emphasis on Mobile Widgets, Christian Kaar, University of Applied Sciences, Hagenberg, Technical Report Number 06/1/0455/009/02, October 2007.

[7] A Messaging API for Inter-Widgets Communication. Stephane Sire, Micael Paquier, Alain Vagner, Jerome Bogaerts. WWW 2009, April 20–24, 2009, Madrid, Spain. ACM 978-1-60558-487-4/09/04.

[8] Awareness Support in a Groupware Widget Toolkit. Jason Hill, Carl Gutwin, GROUP'03, November 9-12, 2003, Sanibel Island, Florida, USA.

[9] Standardising Widgets. Marcos S. Caceres. PhD Thesis at Queensland University of Technology, Brisbane Australia. 17 September 2009 Final draft.

[10] A Layered Framework Supporting Personal Information Integration and Application Design for the Semantic Desktop. Isabel F. Cruz Huiyong Xiao. Department of Computer Science

University of Illinois at Chicago.

[11] Connections: Using Context to Enhance File Search. Craig A. N. Soules, Gregory R. Ganger. Carnegie Mellon University.

[12] Resource Description Framework (RDF): Concepts and Abstract Syntax, W3C Recommendation 10 February 2004, http://www.w3.org/TR/rdf-concepts/, {Retrieved 03-08-2009}.

[13] Widgets 1.0: Packaging and Configuration, Candidate Recommendation, http://www.w3.org/TR/2006/WD-widgets-20061109/, 23 July 2009. {Retrieved 03-08-2009}.

[14] Universal Widget API (UWA) 1.2, Working Draft - 22 August 2008, http://netvibes.org/specs/uwa/current-work/ {Retrieved 03-08-2009}.

[15] Web 2.0 as Approach for Enterprise Information Technology - How Companies Can Benefit from the Internet of the Second Generation. Claus Menzel, VDM Verlag Saarbrucken, Germany, 2007.

[16] OWL Web Ontology Language Overview, W3C Recommendation 10 February 2004, http://www.w3.org/TR/owl-features/, {Retrieved 03-08-2009}

[17] The Processes of Split and Merge in Smart Spaces. Sergey Boldyrev, Ian Oliver, Sergey Balandin. Nokia Research Centre.

[18] Finnish-Russian University Collaboration in Telecommunications (FRUCT) program official webpage. www.fruct.org, {Retrieved 28-08-2009}.