2014 International Conference on Indoor Positioning and Indoor Navigation, 27th-30th October 2014

Beyond HTML5 Geolocation

A Flexible Concept to Enable and Easily Use Advanced Positioning Technologies for Mobile Indoor Location Based Service Web Applications

Martin Krammer, Thomas Bernoulli, Ulrich Walder
Building Informatics Group
Graz University of Technology
Graz, Austria
martin.krammer@tugraz.at

***Abstract*— This paper introduces a possibility for websites running in a standard web browser on smartphones to also use context adjusted technologies not supported by web technology, although strongly needed especially for location based service (LBS) application for. We outline surrounding conditions of the technologies used for the implementation of LBS. Inferred from identified limiting factors and the inflexibility of available location strategies, we show how e.g. an established IMU-sensor based positioning system library could be used by web applications with a standard HTML hyperlink. This extends the applications of available, specialized indoor positioning modules greatly to the domain of existing and upcoming location based online-services.**

***Keywords*— hybrid web applications; native browser interaction; sensor positioning addon; location provisioning; context provisioning**

I. Introduction

Today using information concerning the domain of 'location' is a popular approach of applications to ease the daily life of consumers and professionals with modern technology. Such applications fall into the category of *location based services* (LBS).

An important part within LBS implementations is an online component (i.e. a server) providing up-to-date information to realize the intended service around a location. Therefor it is common to use website technologies around the HTML core also for LBS application development. Further using HTML is convenient because of the big advantage that they are built with the intention to provide the same service to whatever computer platform accesses it. The needed abstraction allows simplified, platform independent development and provides a common user experience for both web technology developers and users.

But the most useful HTML based applications of LBS - LBS with augmented current location information - only became possible with HTML evolving to version 5. HTML5 introduced features to better support developers creating interactive, multimedia and foremost *mobile* web applications. With HTML5 long missing possibilities to access native, platform specific services were specified e.g. the *Geolocation API*. Such a geolocation API call initiates the activation of just a device's standard positioning means i.e. GPS or location inferred from network signals and requests the current location estimate from within websites. [1]

So a further implementation of more advanced location strategies e.g. for desirable indoor-context web applications are still impossible with pure HTML based mobile solutions.

This paper introduces a possibility for websites running in a standard web browser on smartphones to also use context-adjusted technologies not supported by web technology, although strongly needed by developers. We show how an established IMU-sensor based positioning system library was connected to a web application running on a smartphone to support on-construction-site inspections with just standard HTML hyperlink markups.

This novel and easy to use approach extends the applications of available, specialized indoor positioning modules greatly to the domain of existing and upcoming location based online-services.

Further the approach allows the generalization to combine the simplicity and power of web application development with largely any platform native features, which until now were out of reach for web developers. The technology allows for an aspired decoupled development of the web/content part and native/expert feature part of any web application linking them together via simple URI based calls and responses. It pushes web technology feature-wise another step closer to the richness of native mobile software development.

II. LBS - Location Based Services

Starting with still ongoing research topics around raw location sensing algorithms and technologies years ago, other fields of research saw favorable possibilities to augment their applications with location context. This now broad field of expertise is often subsumed as part of the term *pervasive computing.* [2]

Applications focusing on using location data as the sole service-enabling factor are called *location-based services (LBS)*. Today LBS are considered to be one of the major trends in (mobile) information technology. Combining both LBS and pervasive computing to *mobile LBS* leads to numerous possibilities, which partly can be seen and used in everyone's daily life already.

Common smartphones are *the* ideal LBS device everyone carries in his pockets [3]. To state the simplest LBS example, even the cheapest smartphones in the market are outfitted with built-in global navigation satellite system (GNSS) receivers. This enables the development of location aware software. Outdoors such a GNSS provides the device's location to a running smartphone mapping software (*app*), which uses it to at least indicate the location on a map. More complex: With live-location available, the software could provide real-time navigation using a routing graph of the surrounding area or present information about the surrounding points of interest etc. All such LBS apps are not only imaginable but also reality for consumers nowadays, provided by smartphone software developers and researchers specialized in the areas of location sensing and LBS in general.

A. Extending LBS to different usage domains

Shrinath et al. state in [4], that people often misinterpret LBS as just *"maps and check-ins. Location based services (and allied geo-spatial services) have a potential and scope much wider in fields ranging from urban governance, fleet management, data analytics and local exploration"*. Although being a trending topic for years, only recently many of the needed technological parts for satisfying LBS became generally available and began to influence consumer applications a lot or even enabled completely new possibilities. With the proof of the applicability and usefulness of LBS now it starts to attract industrial application developers. LBS can greatly improve or even replace conventional workflows and increase the efficiency, or in other words lower the cost of products, increase security, speed up processes etc.

Introducing LBS within the corporate space also forces software developers to integrate novel LBS services into existing corporate software. This task creates a challenging situation, because two components of potentially very different expert domain software have to work together. In addition the knowledge of two very different types of developers has to be synchronized to create a combined service to get professional work done more efficiently.

B. Decoupling of expert domains: A software engineering challenge

The history of software engineering contains a prominent track of attempts to manage the complexity of software by the age-old warfare strategy of *divide and conquer*. Modularization is the key method to implement this paradigm in software systems. A lot of researchers searched for principles software engineers can follow to define meaningful modules within a specific project.

The concept of information hiding [5] and the principle of low coupling and high cohesion [6] have been published a long time ago but are still the main guidelines to structure software systems. Other work applied these principles using different perspectives on software systems ([7],[8],[9]) or analyzed how software in real organizations is actually split into modules [10].

But not only explicitly stated principles have been published to guide software engineers through their daily struggle. Another source of wisdom are *design* patterns, which became popular in the mid 90ies when first published as a sound and everyday usable collection [11]. By describing problems by their pattern and providing generalized solutions they offer multiple guides how software components can be structured.

Software architectures provide similar guidelines as design patterns but on a bigger scale. There does not exist one architecture which fits every use case but many of them. Some are already well-known for a long time, such as the *layered architecture* for system with a stack of abstractions [12], the *event-driven architecture* [13] which is a good match in case the system consists of loosely coupled components, and the *client-server architecture* [14] which is favorable in case a central component (the server) shall keep track of ongoing activities. Other architectures are not only well-known but have become very popular in the last years such as the *peer-to-peer* (P2P) [15] architecture, a distributed application architecture where all participating systems are equally privileged (peers), or the *service oriented architecture* (SOA) [13] following the idea of providing functionality as unassociated, loosely coupled services.

The above listed architectures are not orthogonal to each other, in contrary, some of them can be combined in a favorable way, *event-driven service oriented* systems may be the most well-known example of such a combination.

The title of this section already named the primary goal of the architecture: Decouple the different domains as good as possible. This is a perfect fit for a SOA.

Once the architecture is fixed, the technology to implement it respecting project specific functional and non-functional requirements has to be identified. Most programming languages provide different possibilities: Libraries with service-providing public interfaces, independent services which communicate using operating system inter-process communication, and network interfaces to communicate with local or remote applications. Every alternative has its pros and cons but each provides the necessary mechanisms to decouple different components of a software system following the idea of a service-oriented architecture.

III. Common Strategies to Integrate Location

So the common software engineering principle to decouple the concerns with independent software components further makes it possible to deploy the development effort of different components to individual (likely specialized) software developers.

This decoupling approach seems to solve the stated challenges coming up with the extension of professional/industrial software applications to LBS. On the one hand, to create useful LBS within professional workflows firstly one has to have a deep understanding of the specific business problem domain. Secondly the same person has to have a principle understanding plus the vision for possibilities to combine them with available location properties. That demands for people with interest and knowledge in different application domains and technologies. Further one needs the will to innovate and allow for off-common-application-context or out-of-the-box thinking.

On the other hand, to create novel LBS within existing professional applications to form professional LBS applications, it is not that important to know exactly how to acquire the needed location. So the possibility to split concerns within the software should be used to also split up the developers of location usage (i.e. the service part) and acquisition (i.e. the location sensing part).

In the following we analyze some obvious approaches to provision the needed location data for LBS extensions.

A. Develop own/custom location sensing component

If one needs location data, it is possible to develop a custom location-sensing component. The assigned people have to research current location technologies and decide which properties are best suited for the specific professional LBS application.

After the evaluation process the chosen technology has to be accessed programmatically and it has to be integrated into the main application. With then available location data, the main application can be extended with the service part of LBS.

It has to be noted that just assigning people to the expert task of location sensing is not enough. The developer has to have or acquire a great understanding around location technologies including probably higher knowledge of physics principles and mathematics depending on which technology is used. Further accessing the location sensor often involves software development skills for low level data acquisition and processing which is probably not common or even possible within the main application development environment.

So it could be concluded that the in-house development of a custom location sensor component clearly involves a serious amount of non-core business application development. Depending on available resources, it may be not efficient to go that route. Further it must be acknowledged that location-sensing technologies change and improve quite fast. So choosing the right technology today may not be sufficient for tomorrow's needs or possibilities. The thread of sunk costs of an unsuccessful or non-future proof implementation is evident.

B. Coupling with context framework libraries

As professional software development has shown to be very successful by using external libraries for specialized, non-core business tasks, research was first to think about bundling and providing context sensing expert knowledge as context provisioning frameworks (e.g. [16],[17]). Obviously within these context-provisioning frameworks 'location' is the hardest but most important type of context for LBS.

An evaluation of existing frameworks shows quite creative approaches how location-sensing technologies can be used. Because of compatibility with computer platforms and location sensing technologies, wide spread programming languages like C or JAVA are used. Also there seems to be a heavy use of web services to communicate between components. Additionally it can be seen that the frameworks often concentrate on the specific research domain of the inventors. This sometimes creates some lack of generality with the individual framework designs [18].

In principle for an extension of applications to professional LBS, the idea of using context frameworks is to be preferred over a custom location component development. It reduces risk, time and very likely cost to develop LBS. But often it means also an integration of a lower level code library with little influence on stability, flexibility and future development, although it is highly important for the main application service. Sometimes it may also be impossible or cumbersome to integrate the framework because of technological barriers.

C. Using HTML5 Geolocation API

The *hypertext markup languag*e (HTML) is used to structure the content of electronic documents. With its ability to link to other HTML documents it builds the foundation of the World Wide Web (WWW) since around 1990 and the current version number is 5 (HTML5). HTML browser software, commonly named as *web browser*, is able to interpret these HTML document structures and display them in a uniform way. By providing HTML documents via servers connected through the Internet, web browsers (clients) can access that entire potentially cross-linked HTML content. HTML allows different *element types* with defined attributes to structure documents e.g. *header* or *paragraph* elements. These elements are delimited within HTML documents by so called markup *tags* for these elements written like `<h1>` and `<p>`. Linking to other content is done by the hyperlink tag `<a href="URL">`. URL is the *uniform resource locator* and defines where to find the content and what protocol is to be used. [20]

Besides structure and linking there is a way to define some kind of interactivity within HTML documents. By including *JavaScript* programming code within the *script* HTML element, HTML documents can be more than only static multimedia content to be viewed. In addition to mainly server side manipulation of HTML content, JavaScript enables a client-side processing of user input, calculations etc. So a thoughtful collection of HTML tags creates the web-based content and interactivity. This in combination with the server infrastructure forms the WWW as everybody knows and uses it today. [20]

Additionally to the current markup definition of HTML5 some scripting *application programming interfaces* (APIs) accessible from JavaScript code have been specified (for an overview see Fig. 1). They enable web developers to create *rich* browser web applications with features similar to desktop applications. But individual web browsers would have to support these JavaScript APIs to enable HTML documents to access the features. So it depends upon the web browser's developers if some feature is supported or not which somehow counteracts the intended compatibility paradigm behind web technology. Further the computer platform the web browser is running on, defines how a specific API is implemented or if it could be supported at all, so also the quality of service could be different on specific platforms.

Within the scope and interest of this paper the *Geolocation API* is important. It has to be noted that this API is not part of the HTML5 specification but is recommended for a *World Wide Web Consortium (W3C)*[1] standard API *"that provides scripted access to geographical location information associated with the hosting device"*. The API is agnostic of the underlying location information and it depends on the specific device, which source of location information is used or if location information is available at all. Typically used in mobile device use cases, "*location could be inferred from network signals such as IP address, RFID, WiFi, and Bluetooth MAC addresses, and GSM/CDMA cell IDs, as well as user input"* or for high accuracy GPS. The API specifies JavaScript function definitions to request a one-time location or a steady update of location information by using different call parameters. It is possible to only request high accuracy locations (i.e. GPS). Following an API request, location data as well as possible errors are delivered by the system to defined JavaScript callback functions to be processed further by the website programmer in JavaScript. [1]

So it is possible and practical to request location information from within HTML5. At least for outdoor LBS use cases on current smartphones it is to be expected that a website can request the device's location quite easily.

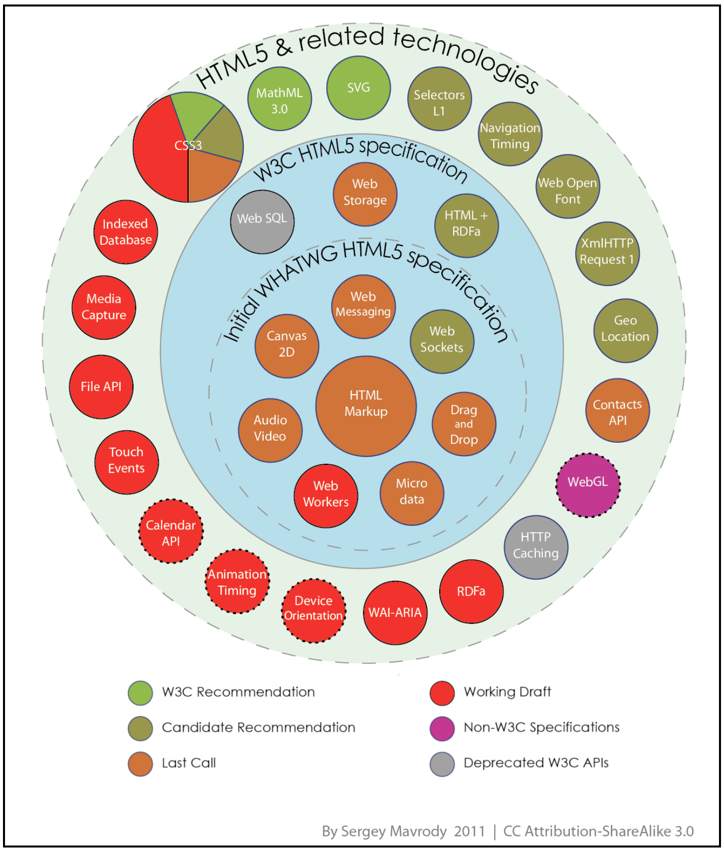
But it must be stated that HTML5's geolocation abilities are not adequate for professional LBS application in general. If professional workflows should rely on LBS it has to work in every possible situation and at every time. Clearly there are professional applications which are outdoors only, but in general it is very likely that there are situations where one has to use LBS while indoors or in covered sky environments. Then the accuracy of HTML5 geolocation decreases to a level generally not suitable for indoor LBS.

IV. New Strategy to Integrate Location

In section III we analyzed three obvious ways one would be able to realize professional LBS applications by integrating the crucial location information. With each identified approach we concluded with advantages and disadvantages.

[1] http://www.w3.org, (July, 2014)

Fig. 1. HTML5-related APIs (based on [24])



In this section we propose a novel, flexible as well as future proof way to integrate location into smartphone based LBS applications like it is needed application- and technology-wise. With this approach we use the advantages of the obvious methods and avoid the disadvantages as much as possible.

A. General requirements for professional mobile LBS

Considering the characteristics of professional applications and LBS, it is very likely that some kind of online server is involved. Normally such a server would provide up-to-date data to support workflows or inform about current information around locations. As a networked environment is already needed for these professional applications, usually common web technologies are used to implement it.

So an HTML based or at least a compatible location provisioning approach would obviously be beneficial for professional LBS application developers, because they are used to it. As stated before it would be still desirable to decouple location and service components as much as possible. It should be possible to change the location provisioning part of the software with better-suited means in the future.

The hardware device for professional LBS should be a smartphone because they are already used in professional environments for other purposes like emailing and calendaring anyway. Additionally to that most people are used to the usage patterns of smartphones even from everyday casual life.

B. URI based calls to a separated context service app

By evaluating the major smartphone platforms[2] to gain some insight into possibilities to comply with the stated requirements in a general way, we could identify a suitable implementation concept, which can be used on all platforms.

By defining custom *URI-schemes* a developer of custom functionality registers this functionality with the operating system. On the other hand developers, who need specific services, can *call* the appropriate URI for a needed external service with query-parameters to adjust the execution as needed. The operating system then executes the service, which is registered with the called URI-scheme. It is just like linking to another website from within an HTML document with the `<a href="URL">` tag.

The different platforms support this custom URI-scheme concept in similar ways for their native app programming languages. But more important the URI-scheme approach works from within HTML documents too. That means one could ask the operating system to start an app with defined URI-scheme by including the tag `<a href="URI-scheme of app">` within the HTML document. This is really just like standard hyperlinking.

The HTML URI-scheme linking behavior is the same on all platforms, which means a broad compatibility for that kind of HTML functionality. Further it means that the application developers can continue with familiar web technology and do not have to know specifics about new location sensing technologies or lower level interfaces.

So after calling the functionality behind a specific URI-scheme it will be started by the operating system. In the context of location provisioning one can imagine what calling something like the following will do:

```
<a href="onpoi://getLocation
   ?requestId = 1234abc
   &locationType = outdoor
   &result_url = http://server.com/context.php">
```

It will trigger the app, which registered itself with the scheme `onpoi`. Further the host part of the URI - `getLocation` - would identify the general action to be taken by that onpoi app. Here it is obviously a request to get the current outdoor location of the device. By using optional query parameters like common with web URLs, the caller can configure the action for specific needs. Here it is used to send parameters for identification and a web URL for results, but could be anything or nothing.

[2] Google Android, Apple iOS, Microsoft Windows Phone and BlackBerry OS

C. Smartphone supported inspection use case

The whole formulated concept of URI-scheme based location provisioning on smartphones was evaluated in a research cooperation with a construction software company.

The construction software company's core competence is an application, which supports construction site workflows by providing a web server infrastructure, and HTML based client application which run in a web browser on various devices. The online server stores and manipulates data of the construction site i.e. mainly building related data and process workflows.

The construction software company recently needed to extend the application to support the process of on-site inspections by mobile devices to collect and access context data from the construction site and synchronize it to the existing building information on the server.

The task was identified to be a data intensive task on small mobile device displays and building informatics research (e.g. [21],[22],[23]) states that data manipulation on construction sites is more efficient by using location-aware software (e.g. for data filter purposes). So the construction software company started to implement location-awareness with the HTML5 based location provisioning approach described in section IV.C. As the application was adopted for construction sites with a majority of covered sky locations, it became clear that HTML5 based GPS respectively network positioning was insufficient for the task.

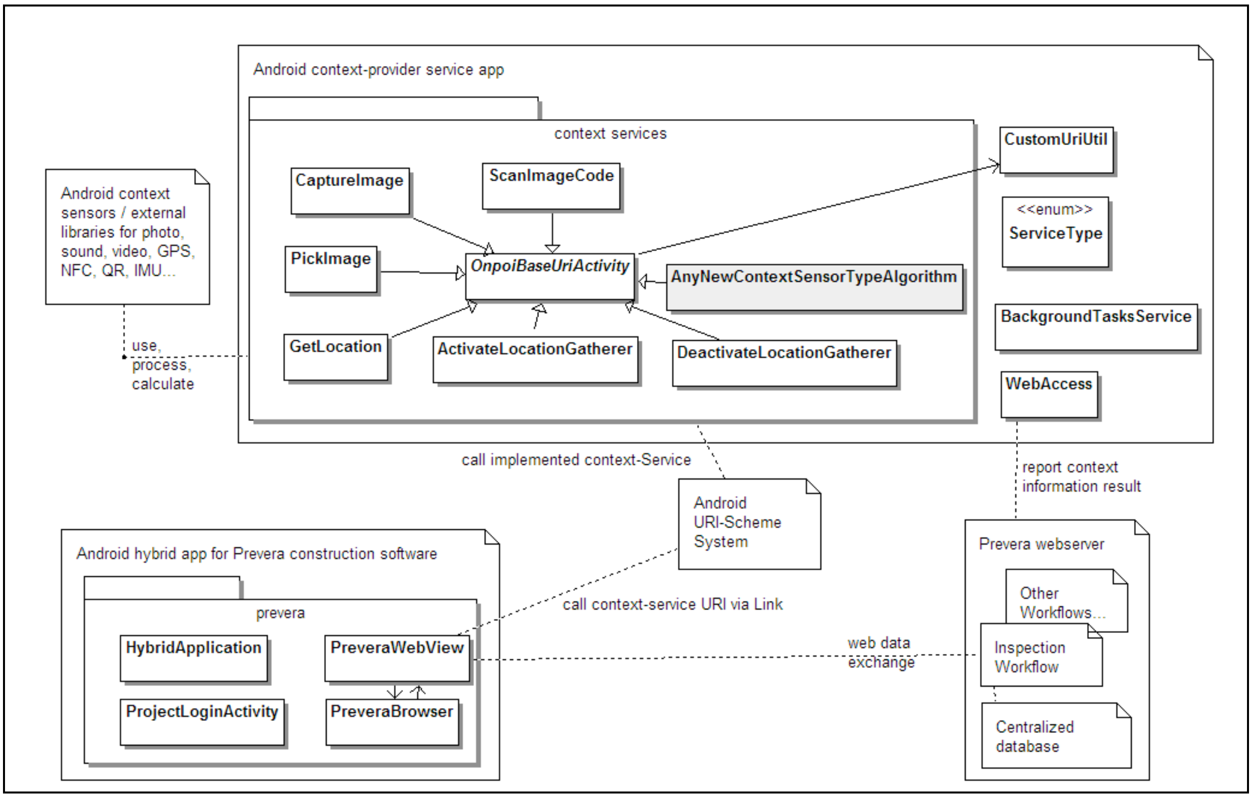
Following that insight a research cooperation with the construction software company to solve the problem had been agreed on. Within the scope of the project we could therefore evaluate our proposed URI-scheme based approach with a real world example. By integrating an established IMU based positioning technology (which has the desired flexibility for construction site demands and which we have developed in previous research projects) into the URI-scheme concept we achieved to add suitable, flexible and future proof location provisioning to the existing professional web based LBS application.

D. Embedding the novel context provisioning into a web application

In this section we outline the proof-of-concept design and implementation characteristics how we could integrate an IMU based positioning system into a professional web based LBS application. In fact the design is capable of providing various kinds of context information from different technologies and as there is nothing specific to be mentioned about integrating the IMU technology, the following is kept technological neutral. This reference implementation was done on the Android smartphone platform.

As can be seen in Fig. 2, we designed the software to be a decoupled component with the sole responsibility to acquire and provide context information. The context component was decoupled as much as possible by implementing the context module as a completely independent context provisioning service app called *ONPOI Services*. By

Fig. 2. Software design to enable flexible context provisioning with web applications



registering URI-schemes for all implemented context types of the app, all context services became available for the professional web application with integrated workflows. The context provisioning modules for the different context types within the ONPOI Services app are also completely independent i.e. decoupled from each other.

So the web application just has to link to the appropriate context services whenever needed within a defined workflow like stated in section IV.B. In response the result(s) of the request will be uploaded to the webserver of the web application according to workflow state specific query parameters given with the URI call. The webserver in turn will update the website according to the current workflow state which can be derived from the identification string also given as a query parameter with the URI call.

The beauty of this context provisioning service app lies in the generality, extensibility and as shown the ease of handling by context consumers which could be every other app[3] or website on the smartphone. The generality was possible because of the extensive decoupling between business application workflow and context provisioning modules.

Concerning the extensibility of the approach it is obvious that firstly it is easy to exchange one algorithm with a better one within the context provisioning service app because of the designed independence from each other. Secondly it is completely transparent for a context consumer, because it always only calls an URI. And which context-provisioning algorithm is behind an URI is the sole responsibility of the context provisioning service app by registering the new algorithm plus URI with the operating system. Considering the rapid advances with location technologies, this concept is a future-proof way to get the best location at any time without thinking too much about details of location sensing for a developer of the service part of LBS.

3 As stated in section IV.B, URI-scheme based calls can also be done with native app programming languages on the smartphone.

V. CONCLUSION

In this paper we stated that LBS applications are a broad area of interest and at least a key part of LBS – location sensing - has been a research topic for several years and is still ongoing for high accuracy and non-outdoor use cases. Applications of outdoor LBS are available already in everyone's pockets following from the success of smartphones with built-in GPS capabilities.

It was stated that with the successful implementation and lessons learned with "easier" consumer LBS applications, now the potential for an efficient use of LBS within professional/industrial applications is well recognized by stakeholders. Being well-established businesses with well-defined efficient workflows supported by highly developed software systems, it is also a challenging software development task to implement LBS in a useful non-destructive way. Following from an analysis of obvious, but very different LBS implementation approaches this paper draws the conclusion that two very different expert software development domains should be separated wisely.

A novel, platform independent and very flexible way to decouple the location-provisioning task from the business application side on smartphones was formulated in theory. The evaluation of the theoretical approach by easily providing context data - including very specialized location technology – was shown in a research project concerning the development of a professional LBS application. The outcome was a prototype design and implementation of a flexible and general context provisioning service app called *ONPOI Services* on the Android smartphone platform.

VI. Outlook and Future Work

The ONPOI Services app which currently also contains the specialized location sensing algorithms should be abstracted to an ONPOI Services Adapter app. This means that the specialized implementations should be externalized as - once further - decoupled, specialized "apps". This context provisioning functionality could then be reintegrated by a plug-in architecture. It would create a further separation between expert domain knowledge.

Experts e.g. in different location sensor technologies could implement their methodologies on smartphones designing them according to the ONPOI Adapter plug-in architecture specification. The specialized plug-in is a separate app registered with the ONPOI Adapter. On the other side of the ONPOI Adapter, context consumers still call ONPOI services via registered URI-scheme calls like shown in this paper. Changes on the context sensing side are still transparent to them.

The advantage of the plug-in architecture is the decoupling of different expert knowledge in the context sensing domain and the ONPOI Adapter could become open-source software without specialized, maybe patented algorithms.

Another useful addition to the ONPOI app would be a device-local ONPOI web server. With that extension, context consumer websites could communicate with the context provisioning side locally, preventing the current necessity to upload context results to an external web server.

References

[1] A. Popescu, "Geolocation API Specification", W3C Recommendation, W3C, 2013, Available: http://www.w3.org/TR/2013/REC-geolocation-API-20131024.

[2] A. Dey, J. Hightower, E. de Lara, and N. Davies, "Location-Based Services", IEEE Pervasive Computing, vol. 9, no. 1, pp. 11-12, 2010.

[3] M. Krammer, T. Bernoulli, S. Muhic, and U. Walder, "Findings from a Location Aware Smartphone Application for a novel Retail Shopping Experience", in Proceedings of The European Navigation Conference (ENC) 2013, Vienna, Austria, 2013.

[4] V. Shrinath, T. Yagna, and K. Jayanth, "Overview of Various Location Based Services and Their Deployment & Adoption Trends Around the World", Market research report by Convergence Catalyst Telecom Research & Advisory firm, Bangalore, India, 2013, Available: http://www.convergencecatalyst.com/downloads.php

[5] D.L. Parnas, "On the Criteria to be Used in Decomposing Systems into Modules", Communications of the ACM, vol. 15, no. 12, pp. 1053-1058, Pittsburgh, PA, 1972.

[6] W.P. Stevens, G.J. Myers, and L.L. Constantine, "Structured Design", IBM Systems Journal, vol. 13, no. 2, pp. 115-139, 1974.

[7] R.C. Martin. Agile Software Development, Principles, Patterns, and Practices, 1st ed. ISBN-13: 978-0135974445, Prentice Hall, 2002.

[8] B. Meyer. Object-Oriented Software Construction, 1st ed. ISBN: 0136290493, Prentice-Hall, USA, 1988.

[9] P. Tarr, H. Ossher, W. Harrison, and S.M. Sutton, "N Degrees of Separation: Multi-Dimensional Separation of Concerns", ICSE '99: Proceedings of the 21st International Conference on Software Engineering, pp. 107-119, 1999.

[10] M. Conway, "How do Committees Invent?", Datamation Journal, pp. 28-31, Relyea Pub. Corp., New York, N.Y., 1968.

[11] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software, ISBN:0-201-63361-2, Addison-Wesley, Boston, MA, 1995.

[12] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. Pattern-oriented software architecture: a system of patterns, Vol. 1. ISBN: 978-0-471-95869-7, Wiley, USA, 1996.

[13] H. Taylor, A. Yochem, L. Phillips, and F. Martinez. Event-Driven Architecture: How SOA Enables the Real-Time Enterprise. ISBN-13: 978-0321322111, Addison-Wesley, 2009.

[14] A. Berson. Client/Server Architecture. Mcgraw-Hill, 1992.

[15] R. Subramaniam, and B.D. Goodman. Peer to Peer Computing: The Evolution of a Disruptive Technology. ISBN-13: 978-1591404293, Idea Group Publishing, USA, 2005.

[16] D. Salber, A.K. Dey, and G.D. Abowd, "The Context Toolkit: Aiding the Development of Context-Enabled Applications", Proceedings of the 1999 Conference on Human Factors in Computing Systems (CHI '99), pp. 434-441, Pittsburgh, PA, 1999.

[17] V. Singhvi, and M. Terk, "PROPHET: A contextual information system framework", Proceedings of 20th International Conference on Information Technology for Construction - Construction IT Bridging the Distance, Auckland, New Zealand, 2003.

[18] L. Fei, S. Sehic, S. Dustdar, "COPAL: An adaptive approach to context provisioning", Wireless and Mobile Computing, 6th International Conference on Networking and Communications (WiMob 2010), pp. 286-293, IEEE, Niagara Falls, ON, 2010

[19] J. Viterbo, V. Sacramento, R. Rocha, G. Baptista, M. Malcher, and M. Endler, "A Middleware Architecture for Context-Aware and Location-Based Mobile Applications", 32nd Annual Software Engineering Workshop 2008 (SEW '08), IEEE, pp.52,61, Kassandra, Greece, 2008

[20] C. Musciano, and B. Kennedy, HTML and XHTML: The definitive guide, 6th ed., ISBN-13: 978-0-596-52732-7, O'Reilly Media; US, 2006.

[21] Z. Aziz, C.J. Anumba, and F. Pena-mora, "A road-map to personalized context aware services delivery in construction", The Journal of Information Technology in Construction (ITcon), Vol. 14, Special Issue Next Generation Construction IT: Technology Foresight, Future Studies, Roadmapping, and Scenario Planning, pp. 461-472, 2009.

[22] A. Behzadan, A. Zeeshan, C.J. Anumba, and V.R. Kamat, "Ubiquitous location tracking for context-specific information delivery on construction sites", Elsevier Journal of Automation in Construction, Vol. 17, Issue 6, Seite 737-748, New York, 2008

[23] K. Menzel, K. Eisenblätter, M. Keller, and R.J. Scherer, „Context-sensitive process and data management on mobile devices", Proceedings of the 5th European Conference on Product and Process Modelling, Slowenien, Portoroz, 2002.

[24] R. Clark, C. Murphy, O. Studholme, and D. Divya. Beginning HTML5 and CSS3. ISBN13: 978-1-4302-2874-5, Apress, 2012.