Title
Location Management for Mobile Devices

Author
Wilde, Erik

Publication Date
2008-02-01

Location Management for Mobile Devices

Erik Wilde (School of Information, UC Berkeley)

UCB ISchool Report 2008-016
February 2008

Available at `http://dret.net/netdret/publications#wil08d`

Abstract

Location-awareness, in the form of location information about clients and location-based services provided by servers, is becoming increasingly important for networked communications in general, and wireless and mobile devices in particular. The current fragmented landscape of location concepts and location-awareness, however, is not suitable for handling location information on a Web scale. Providing users with mechanisms which allow them to control how they want to expose their location information, and thus allow control over how to share location information with others and services, is a crucial step for better location management for mobile devices. This paper presents a concept for representing location vocabularies, matching and mapping them, how these vocabularies can be used to support better privacy for users of location-based services, and better location sharing between users and services. The concept is based on a language for describing place name vocabularies, which we call "Place Markup Language (PlaceML)", and on various ways how these vocabularies can be used in a location-aware infrastructure of networked devices.

Contents

1 Introduction

Location management on mobile devices will become increasingly important in the new future, considering the increasing number of location-enabled mobile devices and location-based services. On the technical side, location-enabled devices and location-based services have been deployed and used for a number of years already (mostly in closed scenarios). However, two issues discussed in this paper have not been investigated in greater detail, the first one being how to make location information openly available on the Web, and the second one being the question of how to provide users with privacy control in such an environment.

Cell phones have provided "location-based services" for a long time, the *E911* [14] legislation in the U.S. and similar laws in other countries have mandated for some time now that cell phone carriers must make cell phone locations available for emergency response teams in cases of emergencies. Other location-based services have been deployed and used in cell phone networks for a long time as well, but mostly in the form of closed environments, where a cell phone carrier provides some sort of location information, which providers of location-based services get access to, if they pay for it. i-mode and WAP were the two main frameworks for this, with i-mode being hugely successful, and WAP being largely a failure in the marketplace.

With the increasing availability of Internet-enabled mobile devices, the lack of location-awareness on the Web becomes more and more apparent. Various services, communities, and technologies have invented their own location concepts, but there is no agreed-upon way how location-aware application can exchange information. Section 2 provides a brief overview of the current landscape, and concludes that there is no way how location information is handled on the Web today. We thus introduce a minimal concept of coordinates and place names, with the goal to (a) turn location into a concept that is represented in a uniform way, and (b) enable users to control how detailed they want their location to be exposed. This includes the option to handle locations as purely name-based concepts, which is sufficient for scenarios such as location matching.

The goal of this paper is to describe our efforts to develop an infrastructure where location is handled as a Web-level concept, location vocabularies can be defined as application-level concepts, and location disclosure can be controlled by users as part of their privacy control on location-enabled mobile devices. While our implementation (described in Section 7) currently only functions as a simple Web proxy, we envision most of the concepts to be applicable to scenarios with programmable location-aware devices, such as *Android* or *OpenMoko* devices.

2 Location Concepts

Geo-location as a concept is used in many scenarios. One of the most common scenarios are navigation services, where users enter two addresses (which in most cases are provided in some user-level address format such as street address and city name), and based on the geo-coding (i.e., the mapping of the address to a geographic location) of these addresses and maps of traffic infrastructures and routing algorithms, users are led from their starting point to their destination. While first navigation systems were stand-alone systems (self-contained units with GPS receivers and all maps built-in), they increasingly move towards more networked devices, using real-time traffic information to optimize the route, and providing additional information about the route (such as nearby restaurants and gas stations) using online services.

More sophisticated systems for managing geographic information use richer information models than just points, they support concepts such as regions and detailed taxonomies for categorization of geographically relevant information. These systems are summarized under the term *Geographic Information Systems (GIS)* and often are specialized for certain application areas. Because of the complexity and specialization of these systems, it is impossible to define a unified data model for all of them.

On the other end of the spectrum, the most basic geolocation concept is that of coordinates. There are several coordinate systems which can be used to express locations, but due to the popularity of positioning using the *Global Positioning System (GPS)*, the coordinate system used by this system, the *World Geodetic*

System (WGS84) [10], has become the most popular coordinate system for geolocations. WGS84 coordinates use simple latitude/longitude pairs which identify a point on the earth's surface.[1]

GPS is becoming increasingly important due to its high precision and the low costs for GPS chip sets. Because of the E911 legislation, however, cell phone carriers must be able to localize all cell phones, and the cell phone localization methods used for this are based on various methods of using the topology of cell phone networks, the exact method depends on the cell phone network technology [15, 16]. For the purpose of this paper, we do not specifically mention this kind of location information, which very often is not available to regular cell phone applications anyway, and if so, has to be mapped to GPS coordinates to be compatible with other coordinate-based locations.

While locations as coordinates are often useful (and readily available from navigation devices), it sometimes is more appropriate to use more abstract concepts for locations. These concepts are *place names* or simply *places*, they are described in Section 3. It is important to note that there is no 1:1 correspondence between location coordinates and place names. Certain coordinates may have no names, other may have exactly one name, and even others may have several names. Furthermore, the names may be managed by different entities (there is no global registry for place names), so whenever *places* are mentioned, it is important to know from which *vocabulary* these places are referenced.

3 Places

Places in their most general definition are named regions which are identified by some group of people. They may be recognized by only one person ("my backyard"), a social group of people ("our club house"), or more or less globally ("Manhattan"). Places can be seen as "emerging" from social and cultural groups [6]. Previous work has looked at making them available through explicit location information [13], or through implicit information that can be found in tagging systems [7]. Practices and opportunities around sharing place information are also being investigated [8], and the general observation is that location and place concepts are becoming increasingly prevalent in today's networked applications.

While it is hard to describe all different ways how places can emerge, our goal was to describe a simple language for representing place name vocabularies. The language is called *Place Markup Language (PlaceML)* and focuses on providing a simple set of basic concepts for describing places. PlaceML is an XML-based language and supports the following concepts:

- *Places:* A place is identified by an ID (which must be unique in the PlaceML vocabulary) and a name. Optionally, places can have aliases (alternative place names), descriptions, and images associated with them. The only mandatory information for a place is an ID and a name, all other information is optional.
- *Place Locations:* Places can be identified by locations. Locations are defined as combinations of geometric primitives. The PlaceML primitives are rather limited, we support polygons and circles, and primitives can be combined by inclusion or exclusion. As a result, PlaceML locations can consist of disjoint regions and can have cutouts.
- *Place Hierarchies:* While places can simply be defined as a set, they can also be organized in hierarchies. Places can have parents, and while it is allowed for places to have more than one parent, the place hierarchy must not contain cycles. As a result, the PlaceML place structure is a set of *Directed Acyclic Graphs (DAG).*

[1] Elevation sometimes also is included in location coordinates, but introduces additional complications, because of the different variants which can be used for measuring elevation (WGS84 vs. mean sea level).

- *Place Groups:* While hierarchies organize places in some location-oriented hierarchy, groups can be used to organize places thematically. Groups have the same descriptive elements as places (ID, name, description, images) and list all their group members, which can be places or groups. As with place hierarchies, the group membership graph must be a set of DAGs.

All information in PlaceML can be provided in different languages, so that it is possible to provide localized versions of vocabularies, which are derived from the same PlaceML vocabulary. PlaceML is a rather simple language and has been primarily designed as a language for defining place name vocabularies for user groups.

If places are simply named concepts, there is no need for mapping facilities, because a place name vocabulary is a set of hierarchically structured concepts. In many cases, however, at least some of the places in a place name vocabulary will be described by locations, which is most conveniently implemented by a map-based UI. Figure 1 shows an example of how a PlaceML vocabulary can be produced, by using the *Google Maps* interface and defining polygons for place locations.

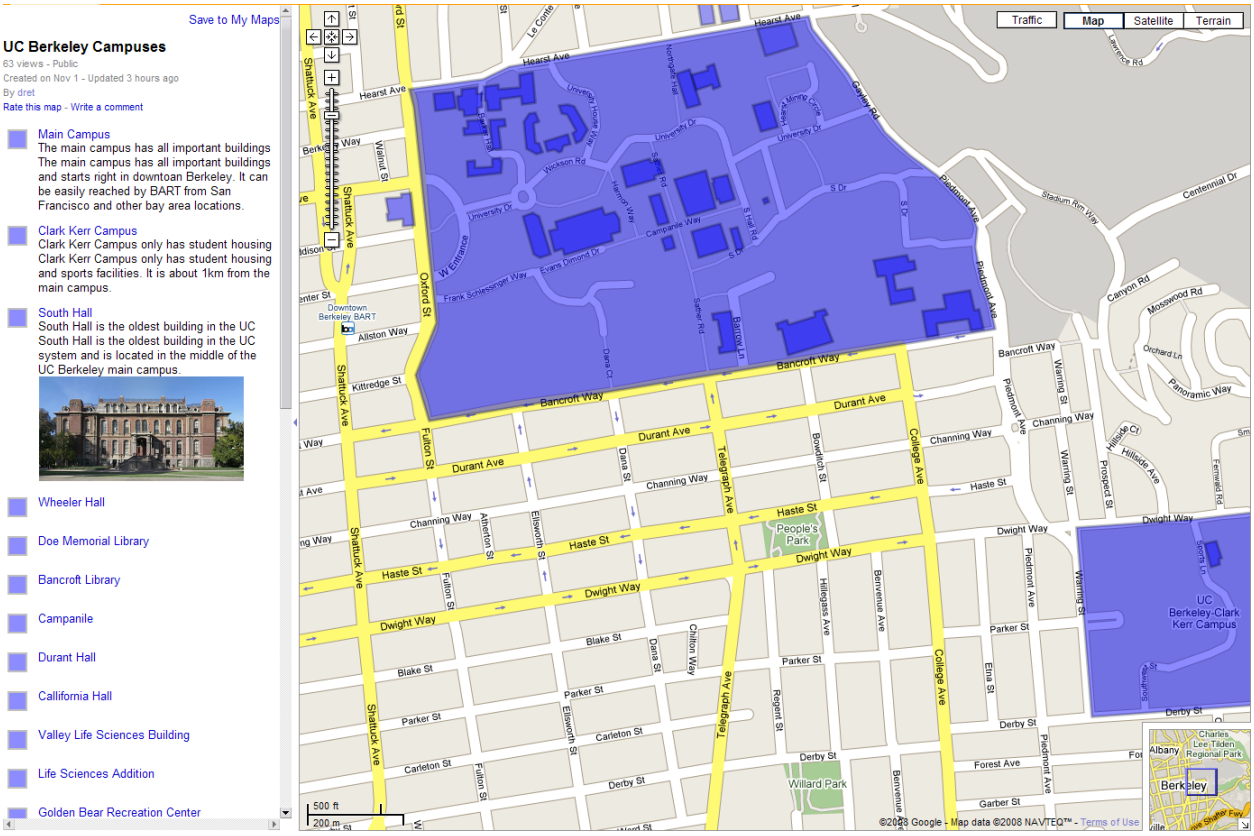

Figure 1: Google Maps Interface for Defining Places

Figure 2 shows the resulting PlaceML, which can be produced from the Google Maps export by applying a transformation for KML (the output format of Google Maps) to PlaceML. It should be noted, however, that PlaceML's features include information which cannot be produced in the Google Maps interface, such as complex locations (anything more complicated than one polygon), place hierarchies, and groups. Thus,

PlaceML creation typically starts with defining the location-based places with the map-based UI, and the resulting KML is then enriched with additional information.

```
<placeml xml:lang="en">
 <name>UC Berkeley Campuses</name>
 <place id="MainCampus">
  <name>Main Campus</name>
  <description>...</description>
  <location>
   <polygon type="include">...</polygon>
  </location>
 </place>
 <place id="ClarkKerrCampus">
  <name>Clark Kerr Campus</name>
  <description>...</description>
  <location>
   <polygon type="include">...</polygon>
  </location>
 </place>
 <place id="SouthHall">
  <name>South Hall</name>
  <parent place="MainCampus"/>
  <description>...</description>
  <location>
   <polygon type="include">...</polygon>
  </location>
 </place>
 <place>...</place>
 ...
</placeml>
```

Figure 2: PlaceML Example

The PlaceML resulting from the KML transformation typically has to be edited to include additional concepts which should be part of the place name vocabulary. Since Google Maps uses AJAX for the map painting application, it would be possible to extend the code with the features which are required for PlaceML and thus produce a complete graphical interface for PlaceML, but since PlaceML is a research effort and might undergo changes, we decided to use the two-step process of map drawing, and some editing afterwards.

It is worth mentioning that place hierarchies and grouping do not necessarily follow strict geographical rules. Places may not even have geographic locations associated with them. And even if they do, place hierarchies are not necessarily strictly defined by geographic locations. For example, even though some of the buildings of UC Berkeley's main campus are outside of the main campus area, they are on directly adjacent areas and are usually considered to be part of the main campus. Consequently, the three northern-most small blocks on the campus map show in Figure 1 are all considered to be part of the main campus, even though two of them are located outside of the campus area.

The main goal of PlaceML is not so much to capture strict geographic location information. The main goal is to capture *place concepts* which are meaningful to user groups. This can be as loosely defined as just three places named *work*, *home*, and *elsewhere*, with no location information associated with them. For social user groups, this kind of information might already convey a lot of information, without disclosing more than

these three "keywords". In fact, BENTLEY and METCALF [1] report that as little information as the *motion status* ("moving" and "not moving") already provided some interesting location-oriented information for users in small social settings.

On the other hand, bigger user groups or more organizational efforts might result in much bigger place name vocabularies. One possibility would be the ZIP codes of the U.S. postal system, which could be defined as such a vocabulary. Other examples are research institutions which have assembled large collections of place names (these collection are often referred to as *gazetteers*), such as the *Getty Thesaurus of Geographic Names* (1.1 million names) or the *Alexandria Digital Library Gazetteer* (5.9 million names). PlaceML is too simple to represent the richness of these data sources, but could serve as a least common denominator for representing the most important concepts of these collections.

PlaceML has two major uses in our location management concept for mobile devices: It serves as a location provider for mobile devices, so that users of mobile devices can choose how they want to expose their location (described in Section 7). Furthermore, PlaceML can be used for *location negotiation* (described in Section 6), a scenario where applications supporting different place name vocabularies can map these (described in Section 5). As some kind of *glue* between these building blocks, our approach is to, as described in Section 4.

4 The Locative Web

The *Locative Web* is a term for referring to a Web which supports location as a first-level concept. The current Web has no concept of location at all, and thus any location-related interaction must rely on mutual agreement. In a location-aware Web, there would be a Web-level concept of location, and this would be embedded into all relevant parts of Web architecture, most importantly as URIs identifying locations and places, HTTP carrying location information, and content formats (such as HTML and XML) embedding location information in a well-defined way.

While the locative Web currently is a vision that yet has to be implemented, there is a lot to gain from such a perspective. Many applications today, even though not all of them may have publicly accessible Web interfaces, rely on parts of the Web's architecture. Well-defined Web services use resource representations which encode resource relationships with URIs. HTTP access to resources uses content negotiation to retrieve the "best" resource for a given request. Crawlers and search engines use information embedded in Web resources to improve searching and provide new ways of providing access to the ever-increasing amount of information on the Web.

The location management concept presented in this paper does not depend on the locative Web to exist, it can be used as a method for managing and mapping locations between location-aware applications. However, a locative Web would provide a better platform, and would allow a better way to integrate location management *into* the Web, rather than implementing it as an application that is based *on* the Web.

5 Location Mapping

PlaceML as introduced in Section 3 is designed to express location vocabularies for user groups. As long as user groups agree on one place name vocabulary, it is not necessary to do any mapping, and users rely on the shared understanding of the place name vocabulary.

However, for communications among users or applications who do not share a place name vocabulary, it is necessary to perform mappings, which can be grouped into three different scenarios, described in the following sections.

5.1 Location to Place

This scenario can be seen as something also discussed in Section 7, where coordinates should be mapped to a name. Locally, this can happen if a user has a GPS-equipped device (which generated location coordinates), but wants to map this information to place names. Based on a PlaceML vocabulary, the following cases can occur:

- *No locations:* If the PlaceML contains no locations, the device has to ask the user to choose a place name. The device may cache the location associated with that name so that it can present the user with a good default the next time this location has to be mapped. But apart from this interface optimization, places have to be selected manually.
- *All locations:* If all places in the PlaceML have locations, then the device can use this information to set the current place name. If there is more than one matching place, place hierarchy and preferences can be used to make the "best" choice. Whether this should be fully automated or confirmed by the user depends on the scenario and user preferences.
- *Mix of location and non-location places:* If there is a mix of location and non-location places, the device can still match the location places, and present the non-location places as alternatives or based on previous interactions. Hierarchy and group information can also be used to influence the selection presented to the user.

Mapping of location to places thus largely depends on the PlaceML. The two most important factors are location information in the PlaceML, and size of the PlaceML. PlaceML with location information can be filtered by the user's current location, whereas large PlaceML vocabularies with no location information will make it next to impossible to design a simple selection process.

5.2 Place to Location

If a user using place names interacts with a service expecting coordinates, then a place name has to mapped to a location. PlaceML supports this scenario by latitude/longitude information which can be associated with place names, so that these coordinates are used for mapping places to locations. If there is no such information, but there is location information (in the form of PlaceML's location primitives using polygons and circles), then it is possible to compute the center of the location and use this as the place's location. If there is no location information associated with the place, then it is impossible to perform such a mapping, and the result of the mapping is the *undefined location*.

5.3 Place to Place

If two place name vocabularies need to be mapped to each other, then there are two ways how this can be done. A typical scenario for this is the case where two applications, which are both based on place names, are interacting, and in order to do so, they must map between the different place name vocabularies they are using. The two possibilities are:

- *Two-Step Mapping:* Based on the mappings described in Sections 5.2 and 5.1, it is possible to first map the place name to a location, and then map that location to a place in the target vocabulary. This process is rather expensive in terms of processing, and can be problematic if the places that have to be mapped are only partially overlapping and the computation described in Sections 5.2 results in a location which cannot be mapped in the second step.

- *Direct Mapping:* Because the two-step mapping is expensive and error-prone, for mappings which are well-known and must be performed often, it may make sense to define a direct mapping. This mapping defines for each place in the source vocabulary to which place it maps to in the target vocabulary (possibly qualified with additional information that needs to be used if the mapping is a 1:n mapping).

Mappings from places to places often will be "fuzzy" mappings, in particular if the place names are not defined by locations and there can be very different levels of granularity. For example, one user might simply have a "campus" place which indicates that he is on campus, while the target vocabulary has a detailed place vocabulary of campus buildings. Mapping the broader place to the more granular vocabulary can only be done reliably with additional location information, which can be acquired by using more accurate location information (such as the user's GPS coordinates) or user interaction (providing a selection of campus buildings). However, in both cases, user confirmation is required, because the user discloses more accurate location information than defined by his vocabulary (the topic of location privacy is discussed in more detail in Section 8).

6 Location Negotiation

Based on the mapping scenarios presented in Section 5, location mapping could be regarded as a fundamental part of Web-based location management. In the same way as the Web today supports content negotiation for content types, languages, content encodings, and character sets, a similar mechanism could be envisioned for *location negotiation*. This goes back to the idea of the locative Web described in Section 4, using Web architecture principles to turn location into a first-level concept on the Web.

Our current approach only focuses peer-to-peer mappings, which can be implemented using PlaceML and optionally mapping information. However, we envisage the locative Web to become a reality in the next few years, and our design is informed by this goal so that PlaceML could, for example, be used in scenarios such as client-side, server-side, or transparent content negotiation (these are the three content negotiation mechanisms described by HTTP [5]).

7 Location Handling

Since today's mobile and wireless devices are not necessarily location-aware, we started implementing our location management framework as a Web proxy. Figure 3 shows the implementation, which is based on a mobile device with Web access, but not necessarily location-awareness. It is important to note, though, that the implementation is even useful for location-aware devices, because these often have no way to use location information for their Web-based interactions.[2] Thus, the proxy-based architecture can be used to simulate a Web as it would exist if the locative Web (as described in Section 4) became a reality.

Figure 3 shows the *Location Proxy (LocProxy)*, a simple Web proxy for making mobile device location-aware. Users of mobile devices have to visit a Web-based UI which lets them configure their location, shown as step 1 (this configuration can either use coordinates as described in Section 2, or place names as described in Section 3). This information is then stored in a database. Furthermore, users have to configure their browsers so that all Web traffic uses the proxy part of the LocProxy. Subsequent requests (as shown in step 2) are sent to the proxy, which adds the location information for that users as a `Geoloc` HTTP header.[3]

[2] PDAs or cell phones with GPS receivers have their location information available on the device, but their browsers send HTTP requests which do not contain this information, so it is impossible to use location-based services with these devices outside of the proprietary infrastructures provided by cell phone carriers.

[3] Our prototype implementation relies on some Web architecture concepts which we have introduced to simulate a locative Web (as described in Section 4 for supporting location-based services, and the `Geoloc` HTTP header is one of these concepts. This paper does not go into the details of our locative Web concepts, though.)

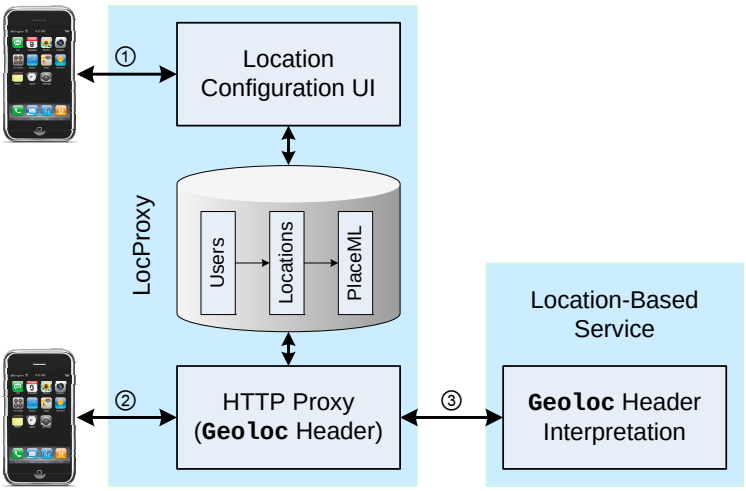

Figure 3: LocProxy Architecture

Location negotiation (as described in Section 6) might be used in step 3, when the augmented request is sent to the location-based service, but our current prototype always sends the location information configured by the user.

LocProxy is implemented as an Apache/PHP interface for the location configuration part, which stores the user and location information in a MySQL database. The proxy part uses Squid, which also accesses the MySQL database and uses HTTP proxy authentication to identify users.

While this scenario is a good starting point for experimentation which location management and location management for devices which are not location-aware, it does not scale well because of the centralized proxy. On the research side, we have built the proxy architecture to experiment with location-based services on campus, which is a rather small user population. On a larger scale, a proxy-based architecture is less advisable, and usually it also is quite problematic to require users to configure proxies for all their HTTP interactions.[4]

A better way to deploy location management would be on the client side, so that for example a client can be configured to locally use PlaceML, and thus only place names would be used by the client. Most mobile platforms are rather closed, though, and it would be hard to implement such a client-side solution. However, recent developments for open platforms (such as the *Android* platform) promise to make it easier to deploy client-side code in a portable way. Android, for example, supports `location.manager` and `location.provider` concepts, which allow applications to manage a variety of various location providers, and also allows customized location providers. Thus, an Android platform could be used to support the scenario shown in Figure 4, where the mapping from a location to a place name (as described in Section 5.1) is done on the client device, based on a PlaceML configuration.

Step 1 allows users to configure their PlaceML, either by simply selecting one they have used or defined before, or by defining a new one. In step 2, the PlaceML is downloaded to the client device, where it is used as the configuration file for the device's location manager. If the device makes a GPS location fix (step 3), instead of directly using the coordinates as location information, they are mapped to the place name defined in the PlaceML. If the user now accesses a Web site, the browser gets the location information from the location manager (step 4), and the HTTP interaction with the location-based service (step 5) is based on

[4]The *Proxy Auto-Config (PAC)* file format allows proxy configurations to be a bit smarter than just proxies for all HTTP interactions, but it still requires all users to install the PAC files so that they are used by their browsers.

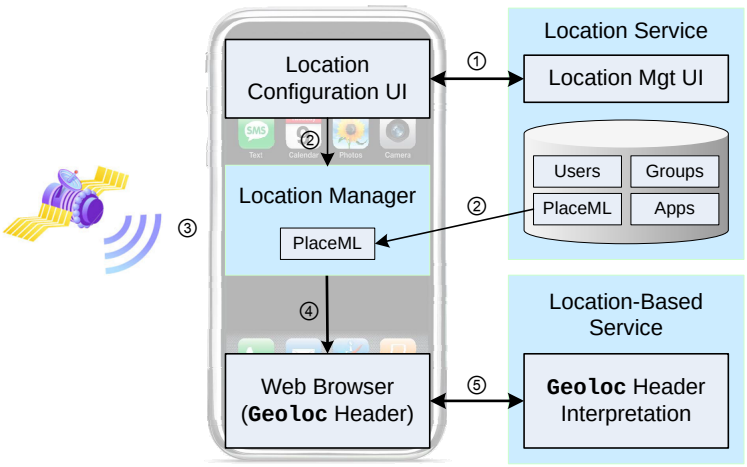

Figure 4: Location Managing Device

the location information that has been configured by PlaceML.

This scenario not only avoids the scalability issues that surround the proxy-based scenario, it also has advantages in terms of privacy issues, because the GPS location never leaves the device. Thus, users have complete control over the way how their location information is made available by their device.

8 Location Privacy

While the Web could be extended to support location concepts as described in Section 4, an important question is that of privacy. People will increasingly use devices with built-in positioning services, but it is questionable whether this information should be sent whenever they are interacting with services on the Web and elsewhere. The architecture shown in Figure 4 thus not only allows for efficient mapping from locations to places, it also allows users to control the extent to which they want to release their location information.

While the scenario shown in Figure 4 only shows one PlaceML, it is easily conceivable that users have more than one PlaceML. They might choose among various configurations based on some external context (they are at work or at home), or there even could be a way how different PlaceML mappings are used for interacting with different services.

Currently, we have not investigated the issues surrounding privacy in such an advanced location management scenario in great depth, but we think that the basic architecture of configurable PlaceML vocabularies could provide a good foundation for implementing privacy measures. As usual, this implementation will have to cope with the inherent struggle between flexible and powerful configurability, and usability. In its simplest form, a location-aware device could have very simple user controls and displays, such as today's *lock metaphor* for secure HTTP connections, to indicate whether location information is being transmitted or not.

9 Related Work

Yahoo Research's *Fireeagle* has not yet been released at the time of writing, but seems to target a similar problem space: using location information from mobile devices and making it available to different services

in different forms. The big difference between the approach presented in this paper and Fireeagle is that the latter is a central service, which assumes that users are sending their exact location information to the service, which will then make it available selectively. On the other hand, the approach presented in this paper relies on a distributed approach, where users control the release of location information locally, so that there is no central service which has a perfect and complete record of the location trails of all users.

The new generation of open mobile devices such as *Android* and *OpenMoko* will make it easier to embed this kind of service on the client side, without the need to re-implement the service for every single device. Android's location API looks promising, but on the other hand has no obvious way to hook into the Android browser. So while Android allows implementing a location manager, there probably is no way how this manager could be made available in the browser, which means that in order to deploy a location manager and a location-enabled browser on an Android platform, a complete browser must be deployed as a new component. For non-browser applications, however, the PlaceML-based location manager could be used without additional complications.

With regard to better support of location concepts as a Web-level concept, there have been proposals for location information in URI [9], HTTP [4], and HTML [3]. However, all these proposals have been individual submissions and do not share a common location model. We believe that for better location management in general, it is necessary to have a common location framework, so that location information can be more easily shared and recombined by different components in location-aware communication systems.

Scenarios and strategies for handling the privacy aspects of location-aware mobile devices will become more important with the increasing availability of mobile devices with build-in positioning functionality. It is unclear how much user demand there will be for better privacy handling for location information. For example, the W3C's *Platform for Privacy Preferences (P3P)* [2], a standard that allows Web sites to express their privacy practices in a standard format that can be retrieved automatically and interpreted easily by user agents, never gained a lot of traction. Still, with the prospect of permanent meter-accurate tracking, users may become more aware of privacy issues and ask for some control of how their location is made available to others.

In terms of other formats for languages describing place vocabularies, the most popular language in that area is the *Geography Markup Language (GML)* [11], but this language has a much wider scope and aims to be a comprehensive exchange format for GIS data. The other language in that area is the *Keyhole Markup Language (KML)*, which is used by Google Maps and Google Earth. GML is mainly for encoding geographic content, whereas KML focuses on visualization of geographic information. KML is a good starting point for visual interfaces (as shown in Figure 1), but not capable of capturing any non-visual information. GML, on the other hand, does have the main concepts required by PlaceML. For example, places could be represented by GML *features*, and associated *geometry objects* could be optional. So while we have not yet defined such a mapping, a transformation from PlaceML to GML would be rather easy to define, so that place name vocabularies could be imported into location management system through standard GML import. However, we believe that PlaceML's simplicity and focus on places and place associations is required for a scenario where GML often is too heavyweight for applications.

10 Conclusions

The location management concept for mobile devices presented in this paper is based on the assumption that location-aware devices will become increasingly common, and that some form of location management is necessary for location-based services to become useful. The declarative approach described in this paper allows users and user groups to define their own place name vocabularies, and allows them to build location-based services on these customized locations. This concept has the advantage that locations can be controlled in a way which is most appropriate for users or user groups, and that this control also implies some kind of privacy control.

From the Web architecture point of view, important concepts on the Web should use representations that can be shared across many applications, and we propose PlaceML as a representation of place name vocabularies. We think that the declarative and distributed approach is more appropriate than centralized approaches. However, integration of location concepts into the Web are only just beginning, and thus it is hard to predict to which level location will really become a first-level concept in the locative Web, or whether the Web will remain largely location-unaware, and location will be standardized or de-facto standards will emerge on some application level on top of the Web. We believe that the Web should become a location-aware information system, and while it will take some time to figure out the most appropriate abstractions for handling location information on the Web level, declarative location information and integration of location information into core Web technologies will play an important role in that process.

References

[1] Frank Bentley and Crysta J. Metcalf. Sharing Motion Information with Close Family and Friends. In Rosson and Gilmore [12], pages 1361–1370.

[2] Lorrie Cranor, Marc Langheinrich, Massimo Marchiori, Martin Presler-Marshall, and Joseph M. Reagle. The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. World Wide Web Consortium, Recommendation REC-P3P-20020416, April 2002.

[3] Andrew Daviel and Felix A. Kägi. Geographic Registration of HTML Documents. Internet Draft draft-daviel-html-geo-tag-08, October 2007.

[4] Andrew Daviel, Felix A. Kägi, and Martin Kofahl. Geographic Extensions for HTTP Transactions. Internet Draft draft-daviel-http-geo-header-05, December 2007.

[5] Roy Thomas Fielding, Jim Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, Larry Masinter, Paul J. Leach, and Tim Berners-Lee. Hypertext Transfer Protocol — HTTP/1.1. Internet RFC 2616, June 1999.

[6] Steve Harrison and Paul Dourish. Re-Place-ing Space: The Roles of Place and Space in Collaborative Systems. In *Proceedings of the ACM 1996 Conference on Computer Supported Cooperative Work*, pages 67–76, Boston, Massachusetts, November 1996. ACM Press.

[7] Lyndon Kennedy, Mor Naaman, Shane Ahern, Rahul Nair, and Tye Rattenbury. How Flickr Helps us Make Sense of the World: Context and Content in Community-Contributed Media Collections. In Rainer Lienhart, Anand R. Prasad, Alan Hanjalic, Sunghyun Choi, Brian P. Bailey, and Nicu Sebe, editors, *Proceedings of ACM Multimedia 2007*, pages 631–640, Augsburg, Germany, 2007. ACM Press.

[8] Pamela J. Ludford, Reid Priedhorsky, Ken Reily, and Loren Terveen. Capturing, Sharing, and Using Local Place Information. In Rosson and Gilmore [12], pages 1235–1244.

[9] Alexander Mayrhofer and Christian Spanring. A Uniform Resource Identifier for Geographic Locations ('geo' URI). Internet Draft draft-mayrhofer-geo-uri-02, February 2008.

[10] National Imagery and Mapping Agency. Department of Defense World Geodetic System 1984. NIMA TR8350.2, Third Edition, January 2000.

[11] Open Geospatial Consortium. OpenGIS Geography Markup Language (GML). OGC 03-105r1, Version 3.1.1, February 2004.

[12] Mary Beth Rosson and David J. Gilmore, editors. *Proceedings of the 2007 Conference on Human Factors in Computing Systems (CHI 2007)*, San Jose, California, April 2007. ACM Press.

[13] Kentaro Toyama, Ron Logan, and Asta Roseway. Geographic Location Tags on Digital Images. In Lawrence A. Rowe, Harrick M. Vin, Thomas Plagemann, Prashant J. Shenoy, and John R. Smith, editors, *Proceedings of ACM Multimedia 2003*, pages 156–166, Berkeley, California, November 2003. ACM Press.

[14] United States Code. Wireless Communications and Public Safety Act of 1999 (911 Act). Pub. L. No. 106-81, 113 Stat. 1286, October 1999.

[15] Yilin Zhao. Mobile Phone Location Determination and its Impact on Intelligent Transportation Systems. *IEEE Transactions on Intelligent Transportation Systems*, 1(1):55–64, March 2000.

[16] Yilin Zhao. Standardization of Mobile Phone Positioning for 3G Systems. *IEEE Communications Magazine*, 40(7):108–116, July 2002.