A Framework for Developing Distributed Location Based Applications

Andrej Krevl, Mojca Ciglarič

University of Ljubljana
Faculty of computer and information Science
Tržaška 25, Ljubljana 1000, Slovenia
{andrej.krevl | mojca.ciglaric}@fri.uni-lj.si

Abstract

Location based services and applications are buzzwords nowadays, yet they have been around for quite some time in a variety of applications. However these applications are scarce because of the high costs associated with the positioning equipment. This paper presents different options for determining location of mobile devices such as mobile phones and Pocket PCs. It describes positioning possibilities using WiFi networks, GSM networks, Bluetooth beacons and the GPS system. Furthermore, it proposes a framework for developing distributed location based applications. The paper specifies which components comprise the framework, data structures that are used for spatial data interchange and Web Services that are used for communication between components. It also describes a location aware application prototype built on top of the proposed framework. It concludes that building applications on top of the proposed framework is feasible and discusses benefits and drawbacks of this approach.

1. Introduction

Location based services are popular these days and if you want the media to write about it, it has to be at least location aware. Yet, location based applications are more common then the media hype suggests. We can find them in car and marine navigation systems, in military applications and in several package tracking systems that are used by companies like FedEx. If these applications are common, why should we reinvent the wheel with proposing a framework for developing distributed location based applications? Applications we mentioned are mostly proprietary software, which means that we cannot reuse them in our own applications without paying license fees. Furthermore they mostly rely on existing spatial databases that are kept locally on devices that run these applications. While local storage greatly increases application performance, it is inappropriate for applications that need up-to-date spatial data like road construction information. Another problem with these spatial databases is that their cost can rise up to thousands of dollars if coverage of a larger area is required. Since maps and applications are mostly developed by the same company, we may find it tough to switch to a competitor's product. Another drawback of existing applications is their dependence on the Global Positioning System (GPS) to learn their location. Although GPS is the most common system for positioning, most common devices like mobile phones do not have an appropriate receiver. Lastly, the fore mentioned systems rarely include the possibility of inserting new spatial data to the database.

The rest of this paper presents the framework architecture that tries to overcome some of the problems described previously with using open standards and commonly used technologies. These make the framework extensible and flexible so it can be used in a variety of location aware applications. Modular design and standard interfaces such as serial ports and Bluetooth enable us to extend our mobile devices with components that read data from various sensors and send the results to our application servers for further processing.

This paper proposes a framework for developing location based distributed applications based on open standards and with extended functionality compared to existing location based products. Some weaknesses of existing products are described in section 2. Section 3 discusses options for determining location of a mobile device and section 4 describes the framework and the prototype of the distributed application. Section 5 includes a discussion of strong and weak points of the framework and proposes further work on the subject.

This paper will be of interest to anyone who wants to get acquainted with location aware distributed applications and to those that wish to implement location based services in their information systems or research fields.

2. Related work

Much work on location based applications has been done by Intel Research Seattle. Their engineers developed software called Place Lab [1] which focuses on using radio beacons other than GPS for location discovery. Place Lab utilizes IEEE 802.11 (WiFi) access points, GSM network base stations and Bluetooth devices to define its location. Decoupling location based applications from GPS receivers is an excellent idea, but it relies on existing spatial databases, that hold radio beacon locations. This information can be provided by organizations that use WiFi networks on their campus or acquired by war-driving. War-driving is a process of collecting radio beacon locations by driving around cities equipped with different wireless receivers and a GPS device. War-driving logs are then filtered and inserted into beacon location databases.

German mobile network operator T-mobile has recently offered a navigation service to its users [2]. They provide software for road navigation. A user that wants to get to a certain destination enters its address into the mobile application. The mobile application contacts the appropriate server which calculates the optimal route from user's current location to the destination. The mobile application then displays driving directions similarly to conventional road navigation systems. A Bluetooth GPS receiver is required for positioning.

Place Lab's ability to determine location based on radio beacons other than GPS and T-mobile's thin client and server side processing approach bring are both great ideas that should be incorporated in modern location aware applications. On the other hand, both solutions lack the ability to insert new spatial data on the fly, independent of the current location.

3. Positioning

People determine their location by objects that are in their eye sight. These can be hills, distinct buildings in a city or stars on the night sky. Despite the advances made in computer vision research, computers remain unreliable at detecting objects, so they have to rely on other means of determining their location.

3.1. Global Positioning System

The most commonly used positioning system is GPS. It comprises of 24 satellites orbiting around the Earth and enables us to determine our location anywhere on our planet with an accuracy of roughly 10 meters. To determine their location, GPS receivers need to obtain signals from at least four different satellites. Four satellites define four spheres defined by the difference between time of sending (from satellite) and time of reception (GPS receiver). The intersection of these spheres presents our current location.

3.2. GSM networks

Mobile phones are gaining on their popularity and are very commonly used in our everyday life. Most of these mobile phones are connected to GSM networks. GSM networks are cellular networks and every cell has its own base station that that has a unique cell identifier. Since the mobile phone always knows which cell it is connected to, we could use this information to determine our location. However, this solution has one major drawback - its accuracy. While the distances between base stations in urban regions are between 200 and 500 meters, they can grow to a couple of kilometers in rural regions and that greatly decreases positioning accuracy. Accuracy can be increased by considering amount of time advance and hand over time information, but this can only be obtained from the mobile network operators who usually charge for such services. Some of the operators are already offering web services that can be used by location aware applications. Unfortunately, there is no common API for accessing location information on operator's servers, which renders such services useless for wide spread applications at this moment.

3.3. WiFi networks

With WiFi networks gaining on their popularity, they can provide a good means of determining our location, especially in residential areas and in the vicinity of larger organizations. Similarly to GSM networks, WiFi clients connect to WiFi access points (base stations), which have a unique identifier (SSID). To determine optimal transfer rates, clients always measure the strength and quality of access point's signal. These two parameters enable us to determine our location. If we receive signals from different access points, we can use triangulation to determine our location with the accuracy of around 15 meters.

3.4. Bluetooth

Another wireless technology that is quickly gaining on its popularity is Bluetooth. There are many devices that use Bluetooth to communicate, and their discovery can help us determine our location. Since Bluetooth was designed for Personal Area Networks (PAN) and is typically short ranged it enables us to determine our location very accurately. However, Bluetooth sources require some additional filtering before we add them to our spatial databases, because most Bluetooth enabled devices are mobile phones and PDAs which do no have a static location.

Positioning based on WiFi, GSM or Bluetooth has one important advantage over GPS. Besides determining our location outside, in open areas, it can also help us with determining our locations inside buildings. On the other hand GPS performs excellently even in rural areas where other radio beacons are very scarce.

4. Framework architecture

The framework comprises of three layers (Figure 1): the mobile device (location aware client), the application server (broker) and the database server that holds spatial data. This approach is used to maximize inter-operability between different components, meaning that any component can be replaced by a new one as long as their APIs are compatible. On the other hand, three-layer architecture enables us to store and process the spatial data on the server side to overcome limitations of mobile devices.

Spatial data is stored in a relational database on the database server. There might be performance disadvantages to this, but interoperability and the variety of available database management systems currently outweigh developing a special geographical database system. Spatial database contains locations of radio beacons and locations of interesting objects or interesting paths. Because of its simple design, the relational database can be easily extended to hold any kind of location or meta-data.

The application server provides web services for mobile clients and communicates with the database server. Client communication is based on SOAP messages while native communication is used when querying the database server. This is against our goals to use only open standards, but it doesn't represent a huge constraint since we are using web services to wrap the proprietary communication. Native database communication can be easily replaced by ODBC or JDBC (dependant on the server platform) however using a native database communication protocol is recommended for best performance.

The mobile device obtains either geographical coordinates or a list of radio beacons (base stations, access points or Bluetooth devices) in its vicinity from one of the wireless receivers. This data are encapsulated in a SOAP envelope and sent to the appropriate web service on the application server. A web service executes a query against the spatial database and returns location information (if any) to the mobile client. The mobile device displays received location information over a map if one is available from the web service. Location information consists of interesting places and paths in the area around our current location.

Since mobile devices are cumbersome for text input there is a web application running on the web server that can be used for meta-data administration. However new

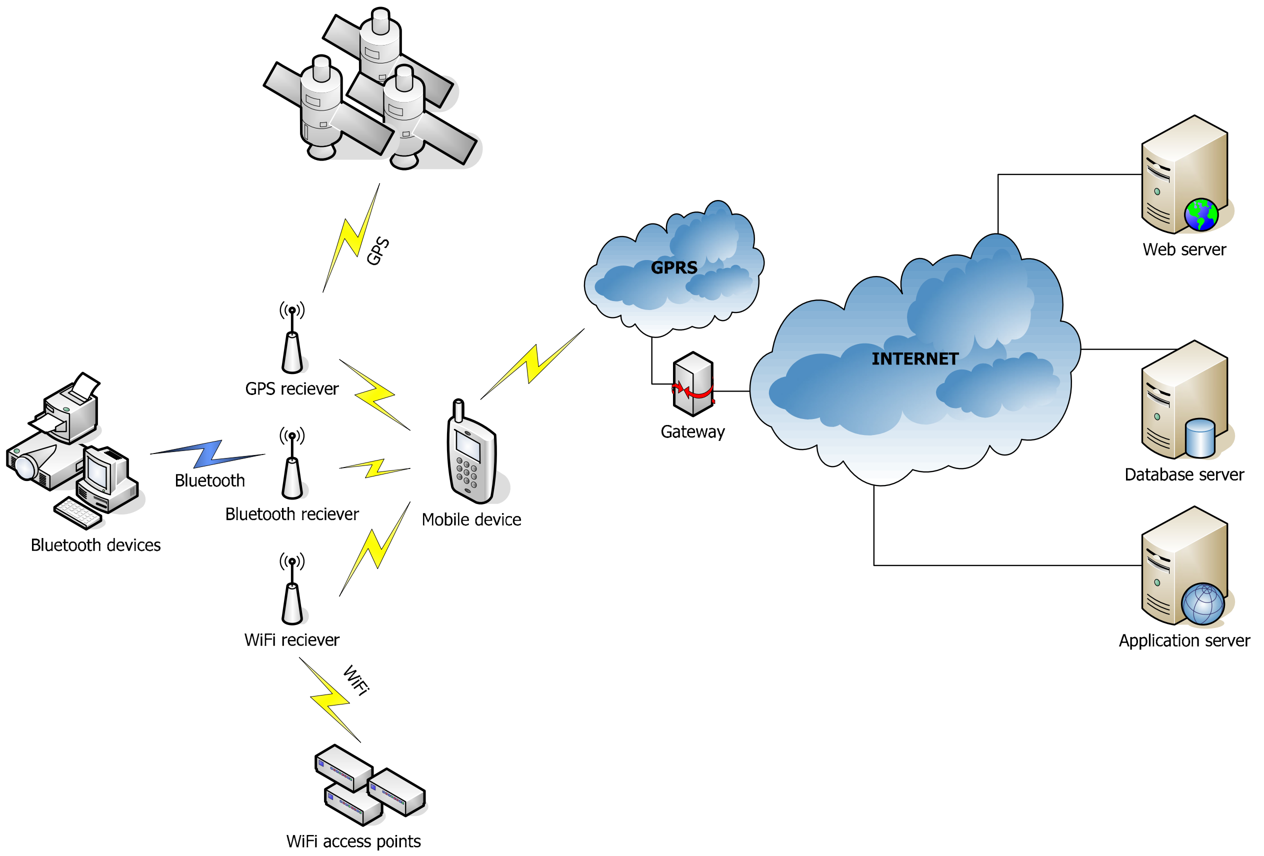

Figure 1: Framework architecture

location data can only be inserted via a mobile device.

4.1. Usage scenarios

Figure 2 shows usage scenarios of a mobile application that is supported by our framework. Locations are interesting places that can be found in the proximity of our current location. Location information is read from the spatial database on the database server. The basic task of the mobile application is to display these places in relation to our current location. If a map of the currently displayed area is available, places are drawn over the map for easier orientation. This functionality can be found in almost all location aware applications.

A path is a set of places identified by the same meta-data. It can be a street, a bicycle tour, a walk around the city centre, directions how to get from point A to point B etc. In addition to displaying places, the mobile application also needs to be able to show paths that are in the vicinity of our current location.

In contrast to commonly available location aware applications, our framework adds the ability for inserting new places into the spatial database. Open source software proved to be successful, why not apply the same principle to spatial databases. Say you are just enjoying your meal at a nice restaurant. You could insert the location of this restaurant to the spatial database along with your comments and let others know about it. Inserting a new location (place) is simple, since it requires only the entry of meta-data for the current location.

Paths can be entered in a similar manner. First the meta-data is entered. Then, the current location is sampled in regular time intervals and stored in the spatial database as a part of the path. Sampling stops on users demand.

Since spatial databases are bound to grow, displaying all locations on the small screen of a mobile device at once would be more confusing than helpful. Filtering can be applied to location data, to show only locations of a certain type (e.g. restaurants) or a certain type of paths (e.g. bicycle tours).

4.2. Mobile application architecture

The prototype navigation application that we have developed implements all of the functionality described earlier with use cases, but only uses GPS to determine the current location. Nevertheless, figure 3 shows that the application is fairly complex, considering it has to run on limited devices. However, our tests have shown that devices like Nokia Series 40 mobile phones have no problem coping with our application.

The center of the application is the *MainControl* class that spawns different *Display* classes for user interaction. Besides switching between displays it also holds current location information that is obtained from *GPSinterface* utility class. Location data is retrieved from the web service *MobileUserWebService*, which is wrapped in *WSInterface* utility class for easy use.

The *AddControl* class is in charge of adding location and paths. If we want to add a new location to the spatial

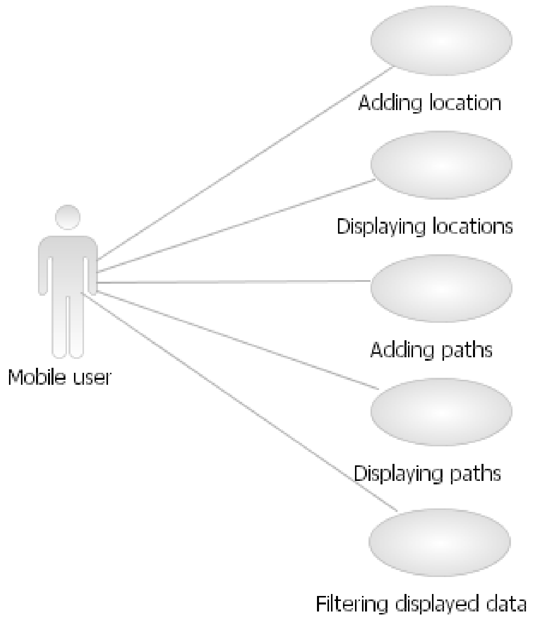

Figure 2: Use case diagram

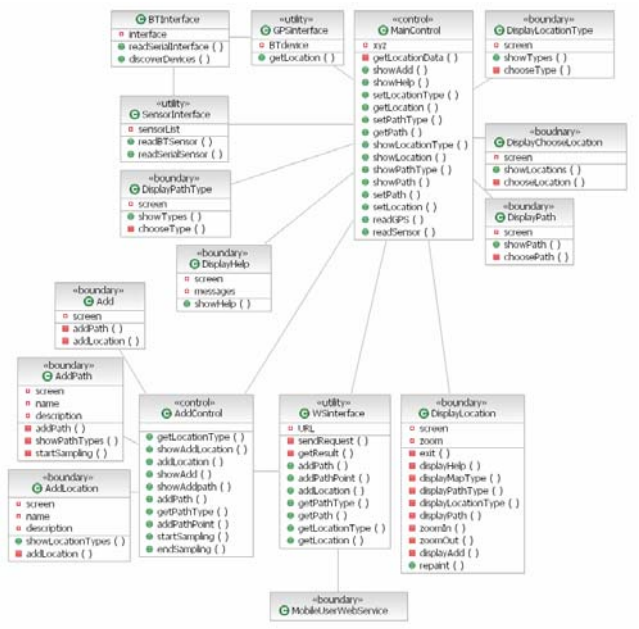

Figure 3: Mobile application architecture

database, we need to enter appropriate meta-data (name, description and location type). The *AddControl* class will obtain current location information from the *MainControl* class, and then *WSInterface* will be contacted to send the newly entered location information to the database. Detailed sequence of interactions is shown on figure 4.

Adding a new path is similar to adding a location up to the point of entering meta-data. Afterwards the *AddControl* class starts sampling current location data from the *MainControl* class in regular time intervals. Location data is sent to the spatial database via *WSInterface* class where it is stored among other locations that belong to the same path. Sampling is stopped by the *AddControl* class on user demand. A detailed sequence of interactions between classes is shown in figure 5.

Utility class *SensorInterface* enables us to read from sensors that use standard RS232 interface for communication. Currently it only supports serial over Bluetooth connections, but unfortunately our heart beat sensor is still in the development phase, so we could not test this part of the framework in our prototype. Since we are using modular design and open standards, the relational database can be easily extended with a table that would hold sensor specific data. The same goes for adding sensor related methods to web services and the mobile application.

5. Discussion and further work

This paper has shown that building location aware distributed applications is feasible by using open standards and technologies. Yes, there are existing location aware applications, but there are no open, easily extensible, frameworks, that can be used to easily develop location aware applications. By using web services we can integrate any client platform like J2ME or .Net compact framework with any spatial database, be it Microsoft SQL server, Oracle 10g or a third party specialized geographic database. Even web services can be deployed on any of the application servers available today.

Of course, no framework includes every feature from the ground up. That is why we would like to include Place Lab's ability to determine user's location with the help of radio beacons in the wild. Using WiFi access points, GSM base stations and Bluetooth devices for positioning can bring location aware applications to many new users, who found GPS equipment too expensive in the past. On the other hand, our software can also be easily upgraded to use the up-coming Galileo project for positioning services.

On all, the open spatial database is a concept that should allow for rapid data growth. However, recent Wikipedia incidents [6] have shown that data insertions and modifications should be at least moderated. Limiting write access with WS-Security [5] can be easily implemented on the server side, but unfortunately more effort has to be put in on at client side due to limitations of mobile devices. Furthermore, the spatial data can quickly outgrow the proposed solution with centralized data and application servers. But since most of our queries are dependant on location data in our vicinity, we can distribute the data to different servers, each holding data for locations in their vicinity. Luckily, the user could use Universal Description, Discovery and Integration (UDDI) to list and locate the appropriate spatial server according to his location. In addition to data distribution, data replication over several servers is possible, thus enhancing service reliability.

Another interesting addition to the framework is the ability to connect different sensors to the mobile devices. A heartbeat sensor would make a great doctor's accessory, since she could be automatically warned of heartbeat irregularities of her high risk patients. Sensor integration and location awareness could spawn numerous applications, utilizing ubiquitous computing to enhance

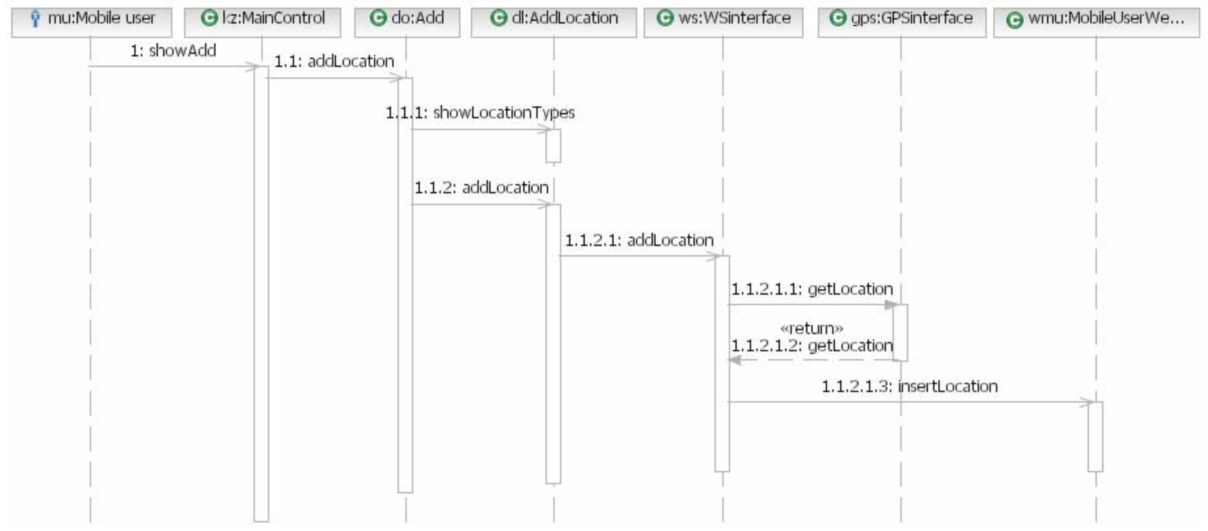

Figure 4: Adding a new location

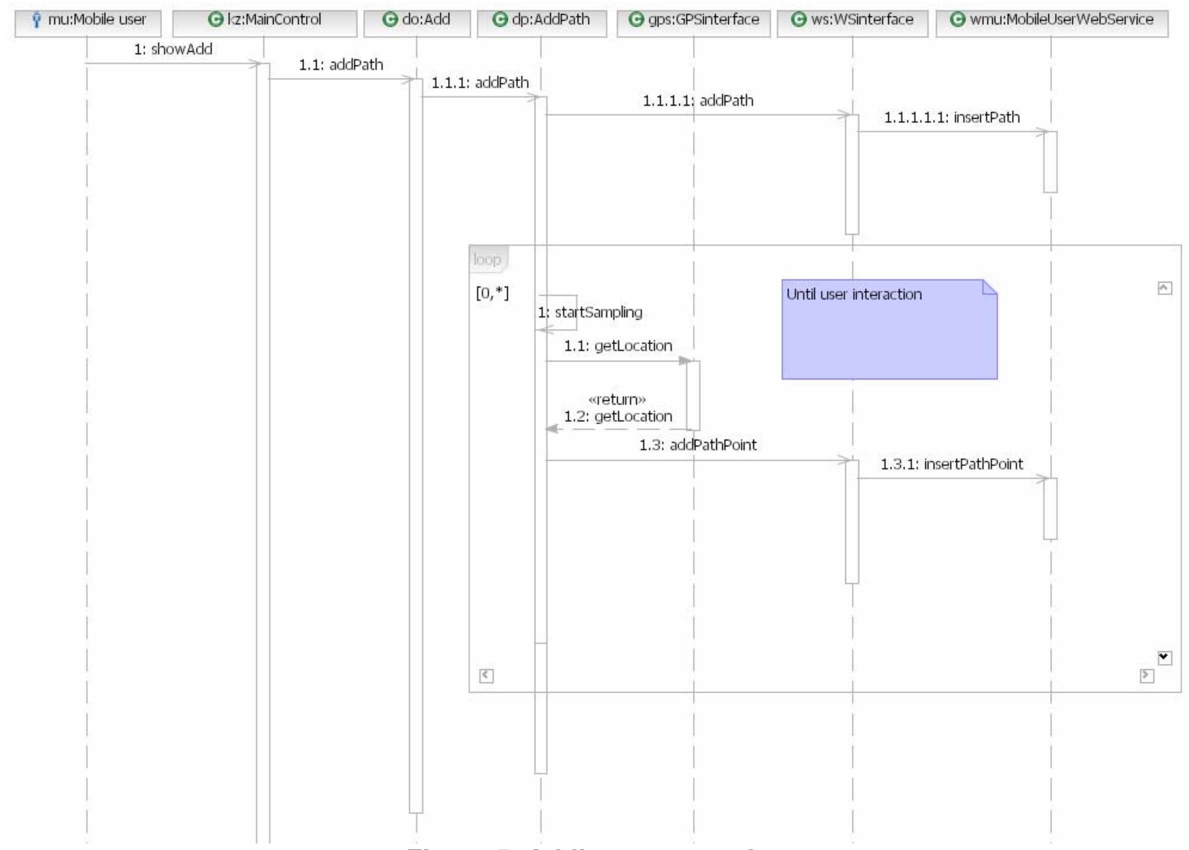

Figure 5: Adding a new path

and simplify our everyday life.

There are still some minor performance issues with transferring large datasets (raster maps) over relatively slow GPRS networks. This can be solved either by using much faster UMTS enabled mobile networks, or by replacing raster (bitmap) maps with their vectorized counterparts. We will also test the feasibility of creating our own vectorized maps from path entries in the spatial database.

Resources

[1] A. LaMarca, Y. Chawathe, S. Consolvo, et al, Place Lab: Device Positioning Using Radio Beacons in the Wild, Intel Research Seattle, 2003.

[2] T-Mobile, http://www.t-mobile.de/, December 2005.

[3] J. Schiller, A. Voisard, Location-Based Services, Morgan Kaufmann Publishers, 2004.

[4] Wikipedia, http://www.wikipedia.org, January 2006

[5] WS-Security Specification, http://schemas.xmlsoap.org/, April 2002.

[6] D. Terdiman, Growing pains for Wikipedia, C|Net, http://news.com.com/, December 2005.