Middleware Models for Location-Based Services : a Survey

Ana Roxin, Christophe Dumez, Maxime Wack, Jaafar Gaber

Middleware Models for Location-Based Services: A Survey

Ana-Maria Roxin
SeT, UTBM
90010 Belfort, France
+33384228502
ana-maria.roxin@utbm.fr

Christophe Dumez
SeT, UTBM
90010 Belfort, France
+33384583038
christophe.dumez@utbm.fr

Jafaar Gaber, Maxime Wack
SeT, UTBM
90010 Belfort, France
+33384583038
gaber@utbm.fr
maxime.wack@utbm.fr

ABSTRACT
Embedded computing systems, sensor networks, LBS pervasive deployment environments, and worldwide computing systems have common characteristics. They are large scale, decentralized and dynamic networks, and needing context-awarness to automatically adapt their behaviour and continue their execution despite network dynamics. Identifying innovative software engineering approaches that take into account all the above mentionned characteristics is a real challenge. This paper focuses on LBS applications and the middleware models required for supporting their operation and characteristics.

Categories and Subject Descriptors
H.3.5 [**Information Storage and Retrieval**]: Online Information Services – *commercial services, data sharing, web-based services.*

General Terms
Documentation, Performance

Keywords
Context-aware systems, pervasive computing, ubiquitous computing, LBS, LBS middleware, middleware model, publish/subscribe model, subject space model, tuple space model.

1. INTRODUCTION
A location-based service (LBS) is defined by the Open Geospatial Consortium as "a wireless-IP service that uses geographic information to serve a mobile user" [1]. A LBS takes into account various user information (context information, past and present user locations, user profile, etc.) in order to deliver information that meets at the most the user's needs.

The design and development of complex distributed LBS applications always call for identifying a coordination model that facilitates the overall design and development process. In the case of LBS, such a coordination model should be able to facilitate adaptive self-organization of activities, and should be complemented by a proper middleware to support the execution of distributed applications.

AUPC'08, July 6, 2008, Sorrento, Italy.

For distributed LBS applications, the two main scopes of a coordination model and the associated supporting middleware are to:

- Provide suitable means to promote context-awareness (contextual knowledge)
- Promote location information both in interactions and in the acquisition of contextual information.

This paper presents LBS characteristics and main middleware models that support the identified LBS characteristics.

This paper is organized as follows: in section 2, basic LBS applications are illustrated in order to extract main LBS characteristics and the corresponding middleware requirements. Section 3 presents the LBS communication model. LBS middleware models are described in Section 4. Finally, conclusion and future works are given in section 5.

2. LBS Application Characteristics
In the LBS context, the role of a middleware is to offer services, models and abstractions implementing the coordination of mobile users, the correlation of information, and information dissemination [3]. LBS present specific requirements and implications for middleware platforms, defined by the LBS applications characteristics. Three main types of LBS applications are presented below. The main LBS application characteristics are identified right after.

2.1 LBS applications examples

2.1.1 Enquiry and information services
The objective of such services is to provide the user with nearby points of interest.

A well-known example would be a service that locates the restaurants near the user's current location. This kind of service must take into consideration the user's preferences, like the type of food he wants, the price rannge, etc.

2.1.2 Traffic telematics
Traffic telematics is a service aiming at supporting car drivers with various services related to their vehicles.

Such LBS is intended to exploit data exchange between a service provider and vehicles in order to enhance traffic information and

improve traffic management processes. They can deal with navigation, diagnostics of malfunctions, warning messages, etc.

2.1.3 Fleet management and logistics

Traffic telematics deal with single, autonomous vehicles. Fleet management applications must manage and coordinate fleets of several vehicles. Generally, LBSs for this kind of applications must obtain the position of a vehicle, place it on a map and display this information to the user.

LBSs can also support each form of logistics as they allow supporting faster transportation, different transportation modes, and development of fallback scenarios in case of failures.

2.1.4 Others

Several other applications exist, but they won't be presented in this article. More information about LBS applications can be found in [2]

2.2 LBS characteristics

From the above presentation of applications, the following LBS characteristics can be extracted. These characteristics are independent from each other and one application can implement more than one characteristic. These characteristics must be taken into account when implementing the middleware technology [3].

2.2.1 Criterion 1 – Information delivery

According to the information delivery policy, two types of applications exist:

- Push-based applications rely on the traditional publish/subscribe paradigm, where information is pushed to the user, based on a given event occurrence or a given condition trigger [6].

- Pull-based applications imply that the user polls the server in search for information updates. It is the user who must request information from the server.

2.2.2 Criterion 2 – User profile gathering

As mentionned earlier, LBS applications benefit from the existence of user profiles, in that they allow personnalized services. There are two ways in which the user profile may be gathered:

- Direct mode – the user's profile is obtained directly from the user.

- Indirect mode – the user's profile is obtained from third parties or by analyzing the user's interaction pattern.

2.2.3 Criterion 3 – Interaction scenarios

The interactions between the user and the service provider depend on whether these are mobile or stationnary entities. A stationary entity has a well-defined and invariant location. Four interaction scenarios are possible:

- Both the user and the service provider are mobile – this applies in the case of mobile ad-hoc location-based applications, notably for friend finder applications.

- Only the user is mobile – notably in vehicle tracking applications and targeted adevrtising.

- Only the service provider is mobile – for example for an automatic airport check-in.

- Both the user and the service provider are stationnary – in this case no dynamic management is needed for location information.

2.2.4 Criterion 4 – Statefulness of interaction

There are two types of interaction depending on whether previous locations of the user are saved or not:

- Stateful interactions characterize applications in which the LBS maintains state across multiple service requests [6].

- Stateless interactions chracterize applications where each request is processed independently from other requests [6].

2.2.5 Criterion 5 – Sources of information

An LBS integrates information from different sources. There are mainly two types of information sources:

- Static information sources mostly concern POI databases or traditional GIS.

- Dynamic information sources vary acording to user's position, the time of day. One may cite traffic information or wheather forecasts.

2.2.6 Criterion 6 – Sources and accuracy of location information

Figure 1 illustrates the main existing positioning techniques, depending on two main criterions [4, 7]:

- The site performing measurments and position calculus,

1. Network-based positioning – the network performs position calculus.

2. Terminal-based positioning – the terminal performs position calculus.

3. Terminal-assisted positioning – the terminal only performs measurements and then forwards the results to the network, which performs position calculus.

- The type of network on which they are implemented and operated.

1. The satellite infrastructures cover large geographical areas and are achieved by stand-alone infrastructures of several satellites. Satellite positionning is always terminal-based.

2. The cellular infrastructures refer to cellular networks (GSM, GPRS, etc.). Cellular networks operators use several methods to obtain the position of a mobile device.

3. The indoor infrastructures are based on radio, infrared or ultrasound systems, deployed in indoor environments and having limited communication range.

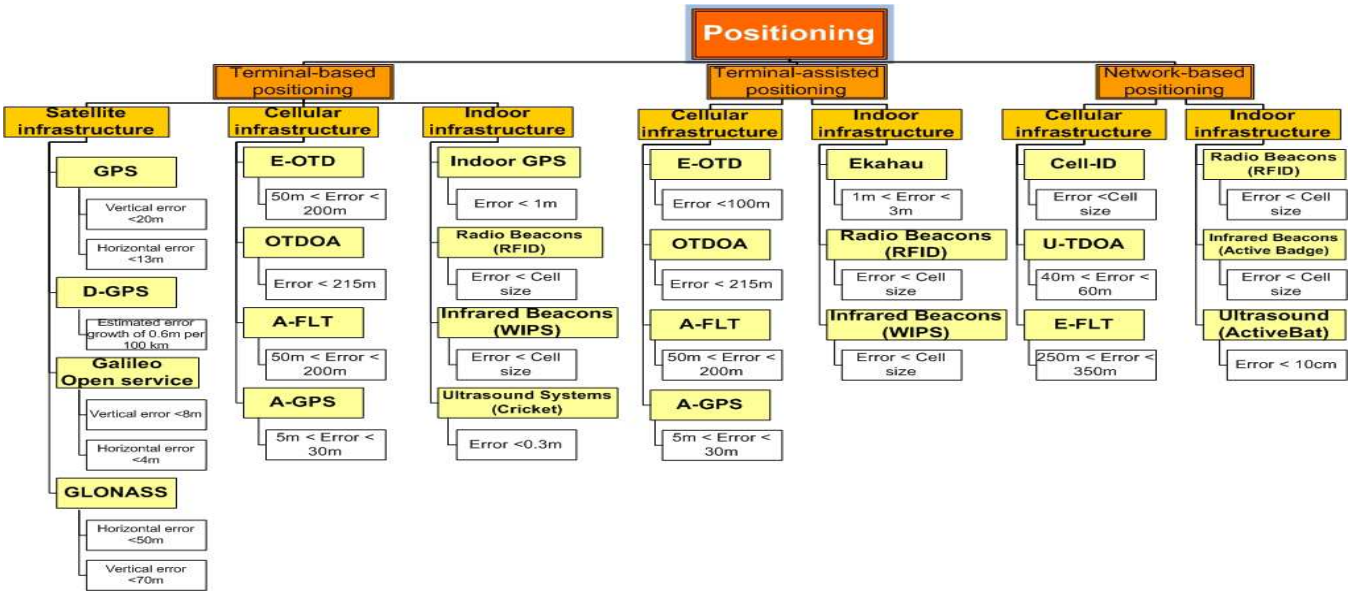

Figure 1. Existing positioning techniques [4].

2.3 LBS Middleware characteristics

Based on the above criterions, the following requirements apply for LBS middleware platforms:

- Support disconnected operations due to user's mobility;
- Support dynamic network topology;
- Allow to take into account a large number of content providers;
- Support various content formats (images, videos, texts, etc.) ;
- Support various notification channels and delivery protocols;
- Support user privacy.

3. LBS Middleware Characteristics

An LBS middleware serves as an interface between users with mobile devices, the Internet and network operators. It offers a single location-based application portal, which means several individually customizable services. An LBS middleware must hide all aspects concerning the operating system, the network protocols, the interacting sub-systems, etc.

As defined in [5], we consider that an LBS has three parts, each of them characterizing a type of data:

- Location data part,
- Geographic data part,
- Bussiness data part.

Based on this assumption, we define the communication model. This model includes the following layers:

- The location information layer
- The middleware layer
- The business-specific layer

The location information layer integrates location and geographic data in order to compute the location of the user's mobile device. Location data is integrated by means of a Position Determination Equipement (defined in [9]). Geographic data is integrated by means of a Geographic Information System (defined in [5]).

The business-specific layer integrates business-specific and user-specifc data in order to deliver highly targeted and detailed information to the user [5]. The ability to provide this user-specific, contextual content is a "key service differentiator" [8].

A middleware's role is to reduce "the complexity and the heretogeneity of distributed infrastructures" by providing an easier programming environment [7]. Therefore, the middleware layer comes between the two previous layers, as it serves as an interface between these layers. The LBS integration is therefore easier. The middleware layer is connected to the network by means of a communication module and manages the user's LBS.

4. Middleware Models for LBS

As presented earlier, there is no unique middleware model. Each LBS middleware differs from the others in that it offers different services to the end user. In this section, we present different middleware models, each of them showing evidence of a wide spectrum of different characteristics.Subsections

4.1 Publish/Subscribe Models

Publish/subscribe models rely on the following principle: producers publish events and consumers subscribe to the events they are interested into [10]. The main component of the model is the event broker for it manages subscriptions and forwards events to subscribers.

Several data models exist for publish/subscribe systems. The data model depends on the subscription language implemented. Here is a non-exhaustive list of existing data models [6]:

- Topic-based models define subscriptions as classes of events groupped by subjects or themes. The publication of an event is directly associated to a subscription by means of tags.

- Content-based models define subscriptions as predicates (or event filters or constraints). A publication is a list of attributes. A

publication matches a subscription if the associated predicate matches an attribute value.

• Type-based models define subscriptions as procedure/function calls allowing recording user's subjects of interest.

• Subject spaces models make no difference between subscriptions and publications. Moreover these models are stateful. This is why these models will be discussed separately below.

The publish/subscribe models have the following characteristics [10]:

• Asynchronous – the event broker ensures that subscribers and publishers operate in an asynchronous way.

• Multipoint – all subscribers with the same interests receive the same publications.

• Anonymous – publishers and subscribers do not need to know each other's identity.

• Implicit – the list of event receivers is implicitly determined by event subscriptions.

• Stateless – an event doesn't last in the system after its publication.

This type of model allows the deployment of applications where information data is matched acording to a given set of constraints. In order to apply this model to LBS applications, one must integrate location information of both publishers and subscribers.

4.2 Subject Space Models

Subject space models structure information into system metadata spaces, called subject spaces [11]. Each subject space represents the metadata of a publish/subscribe system. A subject space allows describing values and relationships between publications and subscriptions. This allows classifying publications and subscriptions into categories, as subject spaces group similar publications and subscriptions.

4.2.1 Subject space representation

Each subject space is represented as a tuple:

$\boldsymbol{\sigma = (D_\sigma}$ *dimensions of* σ, $\boldsymbol{V_\sigma}$ *value set allowed inside* $\sigma\boldsymbol{)}$

Each dimension is also defined as a tuple:

$\boldsymbol{D_\sigma = (}$*Dimension ID, Data type of the dimension*$\boldsymbol{)}$

The data types allowed for a given dimension form the dimension's domain of values, $dom(D_\sigma)$.

4.2.2 Subject space regions

A subject space is therefore multi-dimensional with several data regions [11]. A subject space region is also a tuple:

$\boldsymbol{R = (}C_R$ *set of constraints for R,* $V_{\sigma, R}$ *Values of R in subject space* $\sigma\boldsymbol{)}$

The subject space model defines two types of regions:

• Interest regions – a subscriber's set of interest values in a given subject space.

• Object regions – values provided by a publisher in a given subject space.

4.2.3 Subscriptions and publications

Interest regions and object regions are used to define subscriptions and publications [11].

A subscription is defined by constraints. The subject space model represents it through a tuple:

$\boldsymbol{S = (I_S}$ *Interest regions for S,* $\boldsymbol{f_S}$ *subscription filter defining the object regions corresponding to the constraints*$\boldsymbol{)}$

The constraints for the subscription $\boldsymbol{S}$ are the interest regions for $\boldsymbol{S}$.

A publication aims at delivering content to a group of subscribers. The subject model represents it through a tuple:

$\boldsymbol{P = (O_P}$ *Object regions for P,* $\boldsymbol{f_P}$ *publication filter defining the interest regions corresponding to the constraints*$\boldsymbol{)}$

The constraints for the publication $\boldsymbol{P}$ are the object regions for $\boldsymbol{P}$.

4.2.4 Matching a publication to a subscription

In order to match a publication $\boldsymbol{P}$ to a subscription $\boldsymbol{S}$, two conditions must be met [11]:

• Some object regions of P must satisfy f_S,

• Some interest regions of S must satisfy f_P.

4.2.5 Subject space modelling for LBS

Applied to LBS, subject space models define the content providers as publishers and the end users as subscribers. A user's profile and preferences are articulated as subscriptions.

The user profile information is represented as a user profile subject space $\boldsymbol{\sigma_{user\_profile}}$ having the following dimensions' tuple:

$\boldsymbol{D_{user\_profile}}$ *= {(last_name, string), (first_name, string), (gender, string), (profession, string), (age, integer)}*

Therefore, the user profile subject space is:

$\boldsymbol{\sigma_{user\_profile}}$ *= {(last_name, Dupont), (first_name, Robert), (gender, Male), (profession, Teacher), (age, 37)}*

The user preferences are also represented as a user preferences subject space $\boldsymbol{\sigma_{user\_preferences}}$:

$\boldsymbol{\sigma_{user\_preferences}}$ *= {(food, Chinese), (lunch_time, 2PM), (lunch_budget, 25), (min_fuel_level, 10), (music, rock)}*

The subject space of a restaurant, $\boldsymbol{\sigma_{restaurant}}$, has a price dimension that gives the price of a lunch menu, for example.

The user's location $\boldsymbol{l_{user}}$ and the restaurant's location $\boldsymbol{l_{restaurant}}$ are represented as regions of a location subject space $\boldsymbol{\sigma_{location}}$.

In order to receive an alert when, at lunch time, the user comes near a Chinese restaurant which menus cost less than 25€, the following subscription $\boldsymbol{S}$ must be formulated:

$\boldsymbol{S = (I_S, f_S)}$, where:

$\boldsymbol{I_S = \{}l_{user}$, $i_{user\_preferences}\boldsymbol{\}}$, where l_{user} and $i_{user\_preferences}$ are interest regions in $\boldsymbol{\sigma_{location}}$ and $\boldsymbol{\sigma_{user\_preferences}}$ respectively. The set of constraints for the $i_{user\_preferences}$ interest region are:

$\boldsymbol{C}i_{user\_preferences}$ *= {food = Chinese, price<25}*

f_S = {*Object region O* | □ $l_{restaurant}$, $o_{user\_preferences}$ □ *O:* $|l_{user} - l_{restaurant}| < min\_distance$ *AND* $o_{user\_preferences}$ *enclose* $i_{user\_preferences}$}

A region A encloses a region B when all values of B are enclosed by region A:

$V_{\sigma, A} \cap V_{\sigma, B} = V_{\sigma, A}$

A restaurant that would like to send alerts to people in its vicinity must make the following publication:

$P = (O_P, f_P)$, where:

O_P = {$l_{restaurant}$, $o_{restaurant}$, o_{user}}, where $l_{restaurant}$, $i_{user\_preferences}$ are interest regions in $\sigma_{location}$ and $\sigma_{restaurant}$ respectively. The set of constraints for the user's and the restaurant's object regions are:

Co_{user} = {*food = Chinese, price<25*}

$Co_{restaurant}$ = {*food = Chinese, price=15*}

f_P = {*Interest region I* | □ l_{user}, $i_{user\_preferences}$ □ *I:* $|l_{user} - l_{restaurant}| < min\_distance$ *AND* $i_{user\_preferences}$ *overlap* $o_{restaurant}$}

A region A overlaps a region B when: $V_{\sigma, A} \cap V_{\sigma, B} \neq$ □

As presented above, the subject space model provides flexible semantics that make it very easy to apply to a large panel of LBS application scenarios. This model proves to be better suited for LBS modelling than the previous publish/subscribe model in that it allows stateful interactions.

4.3 Tuple space model

4.3.1 Model presentation

A tuple is defined as vector of typed values/fields. A tuple is a shared, associatively-addressed memory space.

A tuple space is a logically shared memory used for data exchange and synchronization control among the interactive components of a program.

Accessing the tuples on the tuple space is done in an association manner: tuples are associatively addressed by pattern matching through templates.

The following operations can be performed on tuples [12]:

- Place a given tuple into a given tuple space through a "write" primitive.
- A tuple is read from a tuple space through a "read" primitive.
- A tuple is extracted from a tuple space through a "extract" primitive.

The main advantages of tuple space models are [6]:

- Destination uncoupling – an agent needs no knowledge of the future use of a tuple it just created.
- Time uncoupling – tuples have their own life span.
- Flexibility – a tuple space doesn't restrict the format of the data it stores.
- Scalability – tuple operations are completely anonymous.

The tuple space model was originally introduced by Linda, which is a shared data space model of coordination and communication, once very popular in parallel programming. This model is now gaining interest in the domain of distributed computing and multi agent systems. The Linda model allows process communication through a repository of elementary data structures, which is called tuple space [12].

A tuple is a generic "array" that can hold an arbitrary number of arbitrary objects. Each tuple is defined as a sequence of fields. The operations available for tuples have been presented above.

A tuple is extracted from a tuple space through a "extract" primitive. Still as tuples are anonymous [12], their selection is done through templates/patterns, which are the argument of the "extract" primitive. The fields of a template/pattern can contain [12]:

- Actuals, which are values;
- Formals, which must be matched to actuals in order to select a given tuple from the tuple space.

The tuple retrieval primitives can be blocking or non-blocking:

- When no matching tuple is found in the tuple space, blocking primitives are suspended until a tuple matching the given template/pattern is found,
- When no matching tuple is found in the tuple space, non-blocking primitives return a "tuple not found" value.

The following figure illustrates the communication pattern defined by the tuple space model.

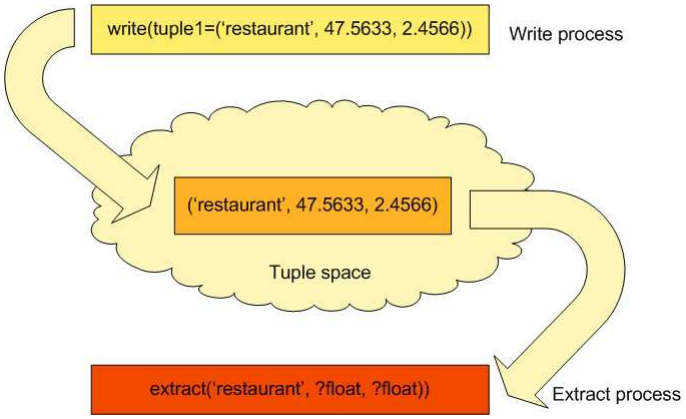

Figure 2. Tuple space model communication pattern.

4.3.2 Tuple space modelling for LBS

In an LBS application, the content providers and the end user publish and extract tuples from one or more tuple spaces. The tuple space model is well adapted to the characteristics of LBS pervasive deployment environments because:

- Tuple-based models allow temporal and spatial decoupling of communication processes, in other words they support spontaneous interactions and coordination activities [6].
- Tuple space models can store both application data and contextual data; therefore tuple-based models can be seen as a unified programming interface that allows accessing both data types.
- Tuple space models allow stateful interactions [6].

Still, the tuple space models have several specifications that don't fit the characteristics of LBS pervasive deployment environments. The two main lacks are:

- Tuple-based models only allow matching semantics that compare exact values of tuple fields. LBS pervasive deployment environments need applications that can deal with uncertain queries, like "find restaurants near me".

- LBS pervasive deployment environments also imply aggregation of data from multiple sources, with different formats and different levels of abstraction. The tuple-based models only allow storing raw data into tuple spaces.

Therefore, the tuple space model must be "upgraded" to support LBS pervasive deployment environments' characteristics [12].

5. Conclusion and future work

The main goal of this article was to extract LBS application characteristics and illustrate the related middleware models. LBS applications require coordination and interaction of multiple users with location information correlated entities. LBS pervasive deployment environments rely on push-based models, where the service activty isn't initiated by the end user but by the service provider.

LBS applications that need the maintenance of state across interactions imply middleware models as subject space or tuple space models. LBS applications that are stateless may rely on traditional publish/subscribe middleware models.

As no standard middleware exists today, one must take into account the interoperability standards defined by the Open Geospatial Consortium in order to define a standard compliant middleware model. Coordination and interaction issues remain the most important research challenges in the context of LBS middleware.

6. Acknowledgements

This work has been supported by the European Project ASSET (Advanced Safety and Driver Support for Essential Road Transport).

7. REFERENCES

[1] http://www.opengeospatial.org/ogc/glossary/l accessed on May 7, 2008.

[2] http://uc.gpsworld.com/gpsuc/article/articleDetail.jsp?id=502585&sk=&date=&pageID=3 accessed on May 7, 2008.

[3] http://lbs.gpsworld.com/gpslbs/article/articleDetail.jsp?id=311566&pageID=1&sk=&date= accessed on May 7, 2008.

[4] A. Roxin, J. Gaber, M. Wack, A. Nait-Sidi-Moh, *Survey of Wireless Geolocation Techniques*, IEEE Globecom 2007, IEEE Workshop on Service Discovery and Composition in Ubiquitous and Pervasive Environments (SUPE).

[5] A. Roxin, C. Dumez, J. Gaber, M. Wack, *Standards and Models for LBS in Pervasive Road Environments*, Research Report.

[6] J. Schiller, A. Voisard, *Location-based Services*, Elsevier 2004.

[7] A. Küpper, *Location-based Services – Fundamentals and Operation*, John Wiley & Sons Ltd. 2005.

[8] Openwave Location Services, *Developing New Revenue Streams from Location-Enriched Mobile Applications*, available online at: http://www.openwave.com/docs/products/location/location_services_ds.pdf

[9] Position determination equipment, United States Patent 4357833, available online at: http://www.freepatentsonline.com/4357833.html accessed on May 7, 2008.

[10] H. A. Jacobsen, F. Lirbat, *Publish and Subscribe Tutorial Proposal*, International Conference on Data Engineering 2001.

[11] H. K. Y. Leung, I. Burcea, H.-A. Jacobsen, *Modelling Location-based Services with Subject Spaces*, © IBM Canada 2003.

[12] G. P. Picco, D. Balzarotti, P. Costa, *LighTS: A Lightweight, Customizable Tuple Space Supporting Context-Aware Applications*, SAC'05, March 13-17, 2005, Santa Fe, New Mexico, USA.