Multimodal positioning support for ambient intelligence

M.A. Martínez, F.J. Villanueva, M.J. Santofimia and J.C. López
Computer Architecture and Networks
School of Computer Science, University of Castilla-La Mancha
Ciudad Real, Spain
[miguela.martinez, felix.villanueva, mariajose.santofimia, juancarlos.lopez]@uclm.es

***Abstract*— The indoor location problem, despite being a hot topic for the research community, it is still an issue with great room from improvement. The poor results reported by the use of stand-alone positioning systems are leading current research efforts to work in the combination of different technologies. This work presents an object-oriented distributed architecture that, supporting different location technologies, is capable of providing a multimodal location system. This work has two major contributions. On one hand, the proposal of a system in which the information retrieved from different location technologies can be combined in order to provide high level services, such as user device tracking or watch areas. On other hand, the tool that evaluates the system precision.[1]**

Keywords-distributed information systems; indoor environments; multimodal sensor; navigation

I. INTRODUCTION

People location for indoor environments is the cornerstone of higher level applications such as real-time tracking, user activity recognition, user and robot navigation or target-of-interest monitoring, just to name a few. The goodness of the results provided by those high level applications are therefore dependent on how precisely people can be located. In this regard, the literature review yields different approaches to people location, ranging from those that cover mathematical aspects for estimation purposes[1], to those that resort to available indoor technologies[2] such as WiFi[3][4], 802.15.4[5][6][7], RFID[8], or Bluetooth[9].

Relevant research is also being done in providing ad-hoc solutions to the positioning problem through different techniques, such as augmented reality[10][11][12] or audio[13][14].

However, last advances in indoor location are being achieved by approaches based on the composition of the information retrieved from different location technologies, improving therefore the resultant indoor location system[15][16][17]. The authors of this work strongly believe that efforts must be addressed on this direction, at least until a more accurate technology appears on the scene. This work has been mainly motivated by the need to overcome the lack of precision of the current positioning technologies. To this end, the range of technologies used to gather location data must be broaden, rather than being just constrained to a single or a small group of technologies.

The multimodal location architecture proposed in this paper is characterized for being technology independent, scalable and auto-configurable. Adopting the object-oriented philosophy, this work provides a high level interface in order to offer location services to external systems in a technologically- transparent way.

The remainder of this article is organized as follows. Section II presents the foundation of the location-based system (LBS) proposed here. Section III describes the proposed architecture. Next, the Section IV shows design considerations, and the Section V explains the evaluation method. Finally, Section VI presents the conclusions drawn from this work along with the future work directions.

II. A DISTRIBUTED LOCATION BASED SYSTEM

The main drawback of current location systems is that positioning is based in the information retrieved from single location technologies, rather than in the combination of several (cooperating) technologies. Furthermore, those systems provide location services to third-party applications by means of monolithic modules in which the location technology cannot be decoupled from the service itself.

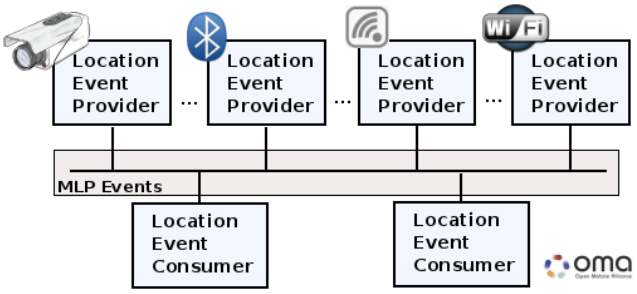

Figure 1. System overview

For the sake of minimizing the coupling between the location technology and the service itself, this work proposes the definition of two roles, as known: the Location Event Provider (LEP) and the Location Event Consumer (LEC) (figure 1). The former supports the technology-dependent role while the latter is technology independent. The adoption of such a decoupled approach entitles LEP to propagate location events to any LEC.

Nevertheless, role separation it is neither the silver bullet for the LBS, since challenges such as the heterogeneity

[1] This work has been funded by the Spanish Ministry of Industry and the Center for the Development of Industrial Technology under the project ENERGOS (CEN-20091048), and by Indra under the Indra chair UCLM 2010

problem arises. The differences among the location events provided by the distinct technologies makes unfeasible to compose information unless that some homogenization tasks are performed upon such events.

For the sake of homogenization, we have followed an implementation based on the Mobile Location Protocol[18] standard adapted to incorporate the tenets of the object-oriented paradigm. This standard defines geometrical shapes, interfaces to manage the location information and the location concept, that is to say, the location event format.

Additionally, the wide variety of systems that would be using the LBS should be also taken into account. It is therefore necessary to offer a high level interface that allows different LBS users to transparently obtain location information. The idea behind the use of a high level interface is to provide a unique way of managing and dealing with location events. It is also desirable to consider supporting different operative systems, programming languages, mobile devices, distributed components, etc. Both challenges -a common interface and heterogeneous software- are to be faced through the use of a CORBA-like object-oriented middleware[19]. Adopting a strategy based in physically decoupling the LEP from the LEC allows LBS to provide a common interface to any third-party system while at the same time abstracting it from the software heterogeneity.

The distributed nature of the proposed system poses some key issues such as scalability and fault tolerance. In this regard, a service discovery protocol (SDP) has also been designed and implemented providing the means to compose and replicate the LBS services.

The main contribution of this work is a technology-independent and scalable LBS, designed on the basis of the aforementioned challenges and the named solutions proposed to address them.

III. ARCHITECTURE

The multimodal indoor location system is composed of two logic subsystems: the LEP -which implements the location event provider role- and the Location Service (LS). The LS, besides from implementing the LEC role, it also provides highly elaborated services in contrast to the raw ones provided by LEPs.

The LEP is a technology-dependent subsystem intended to detect the presence of those devices involved in the location task (via Bluetooth, WiFi, audio, augmented reality, etc). The LEP is also in charge of propagating location events, using the MLP format, to those other services or systems interested on such events. There should be a LEP for each device of each supported technology, and therefore the role of the service discovery protocol is essential for managing purposes.

The interfaces used by the LEP, in listing 1, have been implemented in slice (the interface specification language for the used middleware). Through the *MLP.LocationListener* interface the LEP propagates the location events while the *MLP.LocationAdmin* interface allows the LEC to subscribe to the LEP. In other words, it can be said that the LEC uses the *MLP.LocationListener* interface in order to retrieve location events while the LEP manages the LEC interests in receiving location events by using the administration interface *MLP.LocationAdmin*. Moreover, a new interface, based on the use of generic properties, has been designed in order to deal with arising desires to subscribe to a certain type of location events (that covers a specific area, or represents a concrete technology, etc.).

```
interface LocationListener {
  idempotent void locateReport(Position pos);
  idempotent void locateSeqReport(PositionSeq pos);
  idempotent void locateRangeReport(PositionSeq pos);
};

interface LocationAdmin extends LocationDataProvider {
  void addListener(LocationListener* listener);
  void removeListener(LocationListener* listener);
};

interface LocationAdminProps extends LocationAdmin {
  void addListenerWithProps(MLP::LocationListener* listener,
                            PS::Properties properties)
                  throws PS::UnsupportedProperty;
};
```

Listing 1. Partial MLP slice definition

The property-based behavior is inspired in the CORBA Property Service[20], and it enacts the system flexibility; for example it is possible to subscribe to a particular LEP that satisfies specific conditions such a concrete resolution in a specific area.

These interfaces support the role distinction. However, further challenges need to be faced in order to provide a transparent way of dealing with location events. The proposed solutions are presented underneath.

A. *High level interface*

In order to support the first issue -to provide a high level interface to other systems- we have created the LS. The LS is a LEC service interested in receiving location events regarding a particular area. LS propagates events according to a specific semantics (see figure 2).

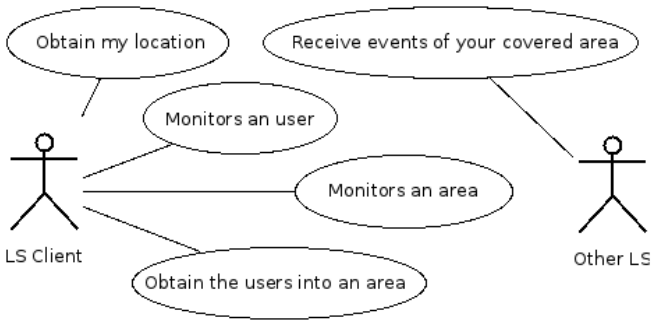

Figure 2. Location Service uses case diagram

Location events look similar in any LBS. We have adopted a structure according to the definitions stated in the MLP standard. The Position structure (see listing 2) consists of three fields, where the first field -msid- determines type and identifier event; the second field -time- defines the event timestamp, and the last field -shape- represents the area where the user has been detected. The shape field has also been defined accordingly to the MLP standard

```
struct Position {
  string msid;
  long time;
  Shape theShape;
};
```

Listing 3. MLP Event definition

The LS can also be seen as an additional LEP that encapsulates and manages location events, and provides them for other LEC. Specifically, the LS offers a common interface, as described in Listing 3, for all the LEPs whose area intersects with the LS area. So, the LS subscribes to all the LEPs whose areas are covered by the LS area. Federation and composition, as it will be explained later on, allow us to build LSs with the capability of covering any area of a physical infrastructure (building, campus, etc.) by composing low-level LS (room, floor, etc.).

```
interface LocationService extends
MLP::LocationDataProvider {

  void federate(MLP::LocationListener* lsListener,
                PS::Properties properties)
    throws PS::UnsupportedProperty;

  void unfederate(MLP::LocationListener* lsListener);

  MLP::Location getLocation(LS::DeviceProfile device)
    throws UnknownIdentifier;

  void trackingDevice(LS::DeviceProfile device,
                      MLP::LocationListener* listener)
    throws UnknownIdentifier, InvalidProxy;

  LS::DeviceProfileSeq usersIntoArea(MLP::Shape area)
    throws UncoveredArea;

  void watchArea(MLP::Shape area,
                 MLP::LocationListener* listener)
    throws UncoveredArea;
};
```

Listing 3. Location Service slice definition

It should be highlighted that the LS may be receiving location events coming from different technologies. Therefore, it is necessary to merge the different user positioning events in just one event. In order to do so, the LS receives events from different LEPs and performs the geometric intersection between all of them (it is worth noting, once again, that a location event, for being MLP-compliant, is described as a geometrical shape). If the intersection is a null set, the LS propagates the event whose technology is more accurate. In any other case, the LS propagates the event that results from the intersection.

In order to support the aforementioned merge mechanism, the LS must know the different features of each technology. For this purpose, the LS recover (via configuration file) three main features of each technology: the accuracy, precision and emission frequency. That is, the LS will use this knowledge to estimate positions.

Additionally, location event management is enhanced with some semantic knowledge that supports event filtering by specific areas or from particular devices, allowing device tracking. Providing semantic knowledge also entails the LS to determine the identity of those users located at a specific area, and to federate several LSs.

In order to illustrate the federation capability we can consider several LSs at a building floor level (one for each floor) and one LS at the building level, which is actually a federation of floor level LSs. Moreover it is possible to configure the federation, using the properties mechanism. For example a federate system could only be interested in the events coming from a specific technology, while another could need all the raw location events.

B. Composition mechanism

To support the aforementioned geographical semantics, the LS needs to know where LEPs are placed and the area they represent. We assume that each LEP knows their covered area. At this point the LS needs to discover (via the Service Discovery Framework[21]) the LEPs that represent areas that intersect with the one covered by the LS.

Area definition has been supported on the standardized Well Known Text (WKT)[22] format. Therefore, each LEP has to state its shape property in its configuration file using the WKT format. On the other hand mechanisms to deal with the Service Discovery Protocol are needed. For this purpose, some middleware features have been used. In this particular case, the ZeroC Ice[19] middleware has been considered, because it provides an efficient publish/subscribe event service called IceStorm, and the capability of creating a grid of computers remotely manageable through the IceGrid service.

The composition mechanism, which allows LS to find those LEP located in its area, is carried out in two different ways.

1) LEP discovery: In this method the LS starts the composition mechanism, therefore finding the LEPs. The LS looks up the LEPs (figure 3) whose areas are covered by the LS. To do so, the LS retrieves the discover channel, and creates an event channel to retrieve the search results. Next, the LS sends a look up message (into the discover channel) that specifies the area, the specific type of server to be discovered (a LEP) and the response channel. When the LEPs receives the discovery event they check their represented area and reply to the LS consequently. Finally the LS subscribes to the location event channel of the corresponding LEPs.

LEP announcement: Through this method the LEP notifies its presence to the system, and the LS subscribes to it

(figure 4). When a LEP starts, it announces itself to the advertising channel. The LS was subscribed to that channel and consequently receives the LEP advertisement. The LS requests its subscription to be added to the LEP location event channel. The LEP checks the LS covered area and subscribes the LS consequently. When the LS is subscribed to the LEP location event channels it will receive the location events. The LS will then propagate these events using the LocationService interface.

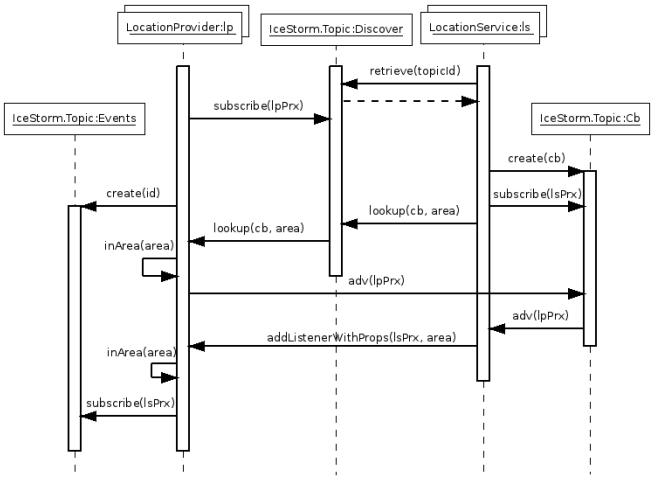

Figure 3. Sequence diagram of the LS discover of LEPs

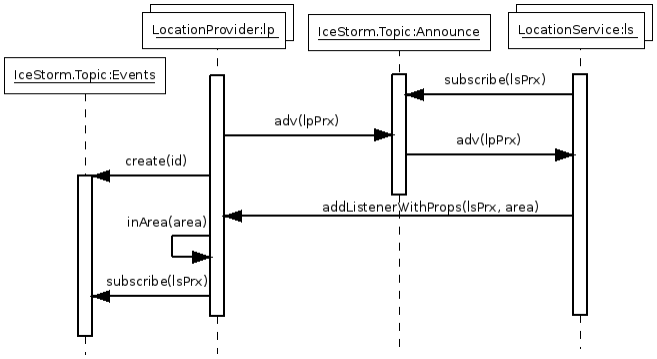

Figure 4. Sequence diagram of LEP announcement

It should be noted that the discovery process uses properties in order to localize the LEPs. That is, the LS sends the "look up" request using a property that represents a shape (in MLP format). Additionally, different properties might also be used for a more restricted search; By example, the LS might be interested in recovering only events of Bluetooth technology.

C. Multiple technology identifiers

The LS can operate in two modes. In the first mode, the LS can dispatch the location events in a semantic way. In this mode the LS does not carry out a batching procedure. It can detect the event area, technology, and identifier and propagate them to the interested LEC -the LECs subscribed via *trackingDevice* or *watchArea* methods-. In the second mode, the LS carries out a batching procedure in order to apply technology merging algorithms, and to only propagates the estimation result. This operation mode presents a new handicap: it is necessary to create one common device identifier which supports the association of all the different technology identifiers.

In order to tackle this issue we have defined the *LS::DeviceProfile*. This class has an attribute which describes a dictionary of pairs of the form technology (as key) and identifier (as value). Thus the system can manage the different devices of each system user.

The aforementioned approximation can only partially solve the problem. It provides the means to manage several technology identifiers. However, a mechanism that allows the retrieval of the device profile with a single technology identifier is also need. For this purpose we have de- signed the *LS::ProfileResolver* which offers resolution methods (see listing 4) such *resolve* and *resolveSeq*, and the *LS::ProfileResolverAdmin* that offers administration methods such bind, rebind, unbind and binds (recover all the binding device profile).

```
interface ProfileResolver {
  LS::DeviceProfile resolve(string tech, string id)
    throws NotFoundProfile;

  LS::DeviceProfileSeq resolveSeq(LS::StringDict techIdDict)
    throws NotFoundProfile;
};
```

Listing 4. LS::ProfileResolver definition

Thanks to the Profile Resolver, the LS can carry out the batching procedures. In fact, if the LS works in this mode it must know the Profile Resolver service, since otherwise the LS will not start. In case of location events without associated devices, for example, motion sensor based LEP, these location events are labeled with an "anonymous" label, and will be used by the system to improve the accuracy.

The component diagram of the figure 5 shows the used interfaces of each role and how it interacts. Note that one location system will have several nodes of each kind. technology.

D. Extra features

The proposed architecture provides some features which are very interesting from the distributed systems perspective, helping to deal with complexity and offering reliability and efficiency.

The composition mechanisms are complementary in the sense that LEPs have to implement both the announcement mechanism in order to notify the system about their presence, and the discovery mechanism so as to allow other services to find them. Additionally, the LS also has to listen to the announcements published in the system, in order to subscribe to LEPs. Previously, LEPs need to be sought in the covered area.

Using both mechanisms LSs are always updated. In this way, some fault tolerant methods can be implemented, since replication mechanisms can be implemented in a transparent way: if two LSs covering the same area are started at same time they will find the same LEPs and subscribe to them.

Hence, both LSs will receive the same location events and therefore will have the same state.

Event filtering is an additional interesting system feature. So, a LS can subscribe to the different LSs just deployed to receive a specific type of events (e.g. estimated events). This behavior reduces the number of location events propagated to higher levels, and along with the federation flexibility it can be used, so as to efficiently handle a complex spaces hierarchy.

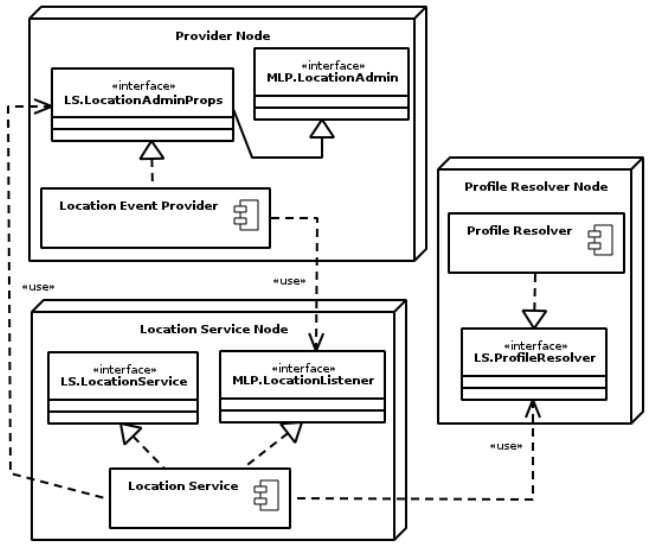

Figure 5. Component diagram of system architecture

IV. LOCATION SERVICE DESIGN

The aforementioned architectural features are supported by the LS; Therefore, the main goal of the section is shows how the LS supports them. For this purpose, it is necessary embody aspects such the composition mechanism, event filtering and event notification in the LS design.

In order to support this distributed architecture, the use middleware proxy concept is essential. On one hand, the LS need use two services: the Profile Resolver and the IceStorm. Both services has presented to LS via proxy. On the other hand, the LS must offer its high level services to other components or systems, and it is carry out through its proxy. Taking into account this concepts, the LS uses the LEP proxies and its proxy in order to support the composition mechanism.

The LS has been designed to work on different ways. On the LS side, it work on batched or single mode (aforementioned), and also it can filter events. That is, the LS can be configured only to receive events that satisfies a customizable filter. In order to illustrate the filter customization, the LS can be filter events that do not belong to users bounded or a specific technology. On the client side, the LS behavior aforementioned is transparent, and the client can use any LS interface method; However, the client could be other LS, and in this case -via federate method- the client can specify the mode -batched or single- and the events that needs via properties.

The LS main goal is provide location events that compound the different technology location events. In order to carry out this goal the LS batch location events for customizable time. When the timeout expired the batched events (and the historical location events of the user) has received by the estimator. The estimator merges the events (taking into account the technology features, the time, etc.) and return the estimated location event. Note that the estimator has been designed in order to provide several implementations, and the LS only must be choose one at start time.

Taking into account this design features the figure 6 shows the location event life from the LS receive it to it is notified. The LS receive a location event and apply the filter (the filter must be null). Next, if the LS is in batch mode, the LS store the event and wait for timeout expires. Note that the location event must be associated to one user, and for this purpose the Profile Resolver is required. When the timeout expires, the estimator receives all the batch events -and historical ones- and estimates the location. Finally, the LS notify the estimated event to its subscribers. Even if the LS is not working in batched mode, the estimator receive the event and generate the estimated event. This behavior allows -by example- use algorithms that works with historical information.

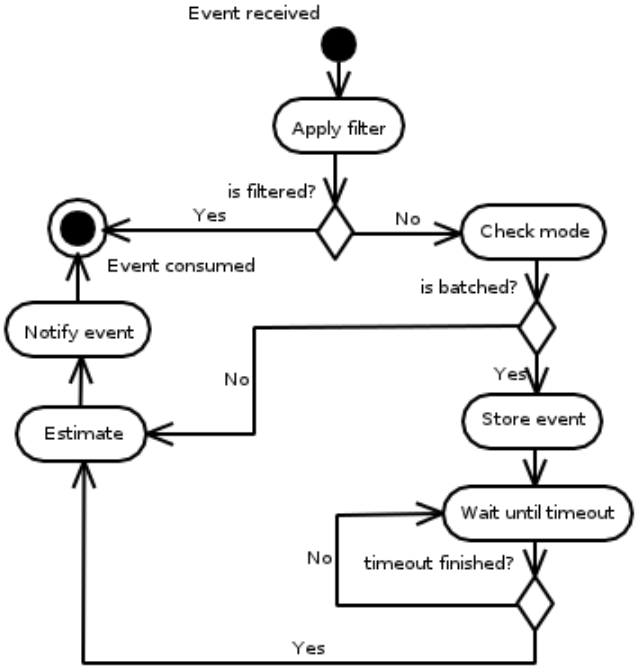

Figure 6. Functional diagram of event life

The notification task must be designed carefully because it is necessary notify all the clients interested; That is, all the clients interests in the event area, event user, and the federate LSs. For this purpose the LS creates and manages dedicated channels for areas and users (and one for the federates LSs), and notifies them base into the generated location event fields.

V. MERGE ALGORITHM PRECISION ASSESSMENT

The proposed architecture supports the integration of different technologies as well as the merging of the different location events they produce. However, in order to design good merge algorithms, it is necessary to assess the precision of the proposed algorithm. For this purpose, we have developed a tool that use all the architecture components and provide to objective appraise algorithm.

The tool (figure 7) makes the most of the system modularity, and uses four services: a generic LEP, the LS, the PR and one specific LEC. Note that the tool uses the existing LS and PR services, and only creates two tool-specific ones, the generic LEP and one LEC.

One evaluation defines four factors: the environment variables, the users, the raw location events and the real path. On the one hand, the fist factor allows configures low-level features of each service -such as covered areas, ports, proxies, modes, etc.-, output results directories, resources for graphics generation, etc. The tool recovers all this information via configuration file. On the other hand, it is necessary to define the devices involved -the location service users- with its associated time-dependent location events, and the path that follows along time. This factors are retrieved by the tool using csv[23] files.

The tool uses the environment information to manage the four services. First, the tool starts the PR and the LS. Next, the tool starts the LEC. Then the tool recovers the raw location events and starts the LEP using this events. The tool monitors its specifics process, and when both finish it carries out the evaluation. For this purpose the tool retrieves the estimated events -generated by LS and received by the LEC- and the real path information, and generates the measure algorithm and several graphics that illustrate all the estimation process.

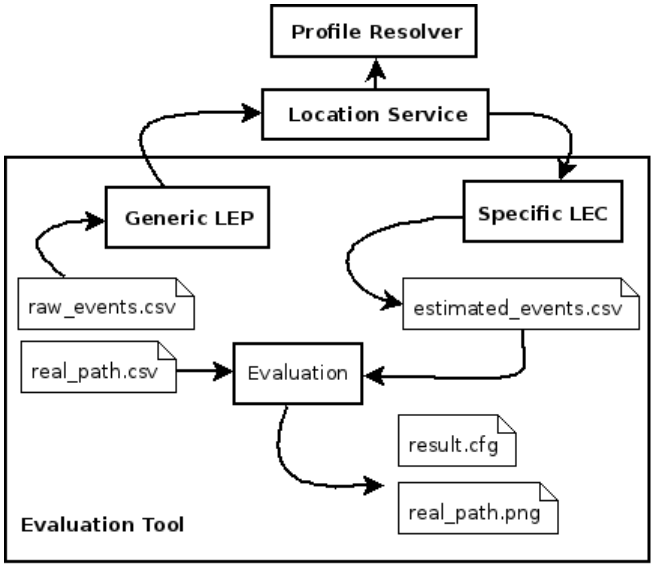

Figure 7. Schema of measure tool

There are three important aspects in the tool: the file formats -that stores the events the real path and the users-, the evaluation algorithm and the result of algorithm evaluation.

A. *File formats*

The file formats affects the LS and the LEP. The LEP must recover the events that will notify at the LS. For this purpose each line of the LEP event file has three values separated by "|" symbol: the msid, the relative time and the shape (listing 5). Each line field corresponds with *MLP.Position* field, except the time one that it is the wait time between the event and the next. Note that shape fields has presented in WKT format, and that this LEP can be provide events of any technology.

```
wifi:00:1e:8c:7b:7b:b1|300|POINT(60.0 19.0 0.0)
rfid:160.15.24.18|500|POINT(60.0 20.0 0.0)
bt:00:02:72:06:20:9D|1000|POINT(55.0 10.0 0.0)
```

Listing 5. LEP event file example

The LS must be work in batched mode, therefore it is necessary that the simulator binds the simulated users into the Profile Resolver. For this purpose, the tool recovers the users from other csv file (listing 6). Each line of the file identifies one user. Each technology identifier is delimited by the "|" symbol, and the key and the value identifier are separated by the "," symbol. Besides, the last line field identifies the more relevant technology identifier.

```
bt,00:02:72:06:20:88|rfid,12345|primary-key,bt
wifi,00:1e:8c:7b:7b:bb|rfid,54321|primary-key,wifi
```

Listing 6. LS user file example

The real path is provided to the tool through other csv file that defines the position -in WKT format- and the relative time of the user along the time.

B. *Evaluation algorithm*

The evaluation algorithm (listing 7) is based on the polygons common area. The algorithm use the real path polygon - it is created a polygon using the shapes defined into the real path file-, and the estimated polygon -created with the estimated shapes received by the LEC-. With both polygons the algorithm carries out the difference, the intersection and the union.

First the algorithm checks the area intersection -the common area-, and if there is zero the similarity is zero. In other case, the algorithm checks the area difference -the uncommon area-, and if it is not zero the common area is then calculated and weighted, and finally returned. If the uncommon area is empty and the real and estimated areas are equals the polygons are equals. In other case, the polygons have a partial common area that is calculated through the subtraction of the real and the estimated area and weighting by the real area. Note that it is necessary to differentiate the real and estimated area, in order to penalty by the uncommon area.

The algorithm provides successfully results, however it is necessary to take time into account. That is, two equal polygons must be equals at same time, must be synchronized.

Note that the event files hold the triggered relative times, and therefore it is possible to evaluate the procedure using a sampling technique. The evaluation measure is the average error between several samples along time.

```
difference   = polygon_difference(real, estimated)
intersection = polygon_intersection(real, estimated)
union     = polygon_union(real, estimated)

if area(intersection) == 0.0 :
  return 0.0

if area(difference.area) != 0.0 :
  common = (area(union) - area(intersection))
  common_weight = common/area(union)
  return  1 - common_weight
else:
  if area(real) == area(estimated) :
    return 1.0
  elif area(real) < area(estimated) :
    uncommon = area(estimated) - area(real)
    uncomon_weight = umcommon/area(estimated)
    return 1 - uncommon_weight
  else:
    return (area(real) - area(estimated))/area(real)
```

Listing 7. Evaluation algorithm

C. *Result of algorithm evaluation*

The main goal of this tool is to evaluate the similarity grade between the estimations and the reality; However, the results could be offer extra information about the system behavior. For this reason, the result is composed of two kinds of files: texts and graphics.

On one hand, a text file provides information about the evaluation algorithm results. That is, the average evaluation measure, and the sampling measure of each time interval. Moreover, an additional file stores the estimated events received by the LEC (with the same format used to define the raw location events in the LEP).

On the other hand, there are several graphic files that provide a system overview. The graphics can split in two groups: single graphics, and composed graphics. There are three single graphics: one graphic shows the real path followed by the user, other graphic shows all the raw events that the LEP uses, and the third graphic shows the union of all the estimation events received by the LEC. The composed graphics illustrates the system behavior along time, and its are made up by an image sequence. There are two composed graphics, the first shows the user real position along the time. The second shows the estimation events received by the LEC along the time.

D. *Tool features*

The main feature of this tool consists in the fact that it works directly with the real LS. This feature support -using real data- the retrieval of nearby results reality. In fact, we use one LEC in order to recover the location events used by the tool. That is, the LEC subscribes into the real LS and recover and store the events via federation. With this data we work with the tool to increase the accuracy of the merge algorithm.

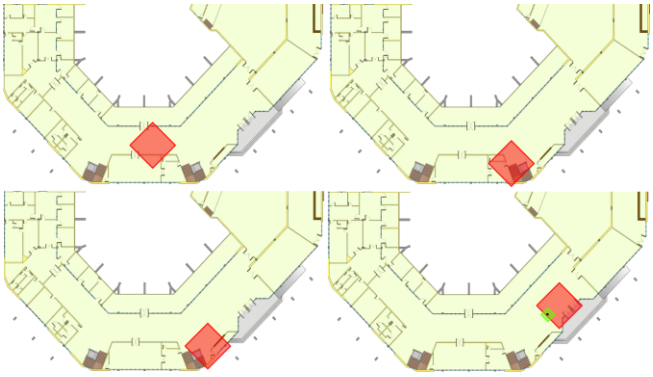

Figure 8. Graphics sequence of estimated events along time

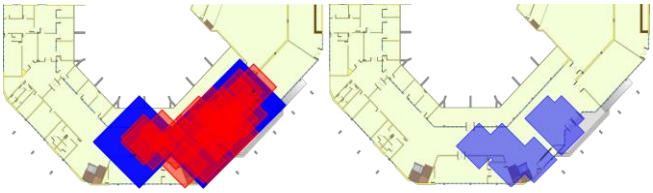

Figure 9. Graphics of raw (left) and estimated (right) location events

VI. CONCLUSIONS

This paper describes a multimodal indoor location system capable of combining several positioning technologies so as to enhance location accuracy. To this end we propose an object- oriented distributed architecture which is characterized by its scalability, technology independence and flexibility.

There are two crucial aspects in the system. On the one hand, the use of a middleware places an abstraction layer in between the services and the location technologies. On the other hand, the use -and implementation- of standards increases the integration capabilities of the system, so as to be reused or to adopt other technologies.

In order to obtain an objective assessment of the merge algorithms, this work also proposes a design and development of a tool that evaluates the algorithm and provides information that allows the understanding of the location events evolution along the time.

Future works are focused in the design, implementation and comparison of several merging algorithms. We will use common-sense reasoning so as to estimate positions based on enhanced knowledge about how the world works, or the generation of synthetic events. That is, the LS can generate location events from received ones. For example, the LS can generate an event location -with a meter of precision lost and with one delay second- from an RFID event. This, synthetic events could help to reduce the noise of other technology with less precision. On other hand, we also are focused on

integrates new location event providers that enhanced the system accuracy. After the text edit has been completed.

REFERENCES

[1] F. Seco, A. Jimenez, C. Prieto, J. Roa, and K. Koutsou, "A survey of mathematical methods for indoor localization," in Intelligent Signal Processing, 2009. WISP 2009. IEEE International Symposium on, 2009, pp. 9 –14.

[2] D. Zhang, F. Xia, Z. Yang, L. Yao, and W. Zhao, "Localization technologies for indoor human tracking," CoRR, vol. abs/1003.1833, 2010.

[3] M. Youssef, A. Agrawala, and A. Udaya Shankar, "WLAN location determination via clustering and probability distributions," in Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on, march 2003, pp. 143 – 150.

[4] Inc. Cisco Systems, "Wi-fi location-based services 4.1 design guide," Tech. Rep., May 2008.

[5] P. Barsocchi, S. Lenzi, S. Chessa, and G. Giunta, "Virtual calibration for rssi-based indoor localization with ieee 802.15.4," in Communications, 2009. ICC '09. IEEE International Conference on, 2009, pp. 1 –5.

[6] Y. Zhao, L. Dong, J. Wang, B. Hu, and Y. Fu, "Implementing indoor positioning system via zigbee devices," in Signals, Systems and Computers, 2008 42nd Asilomar Conference on, 2008, pp. 1867 – 1871.

[7] Y. Lim and J. Park, "Practical indoor positioning system using received signal strength in ieee 802.15.4 networks," in Consumer Electronics, 2009. ICCE '09. Digest of Technical Papers International Conference on, 2009, pp. 1–2..

[8] T. Sanpechuda and L. Kovavisaruch, "A review of rfid localization: Applications and techniques," in Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2008. ECTICON 2008. 5th International Conference on, vol. 2, May 2008, pp. 769–772..

[9] A. Salazar, "Positioning bluetooth reg; and wi-fi trade; systems," Consumer Electronics, IEEE Transactions on, vol. 50, no. 1, pp. 151–157, Feb. 2004..

[10] H. Hile and G. Borriello, "Positioning and orientation in indoor environments using camera phones," Computer Graphics and Applications, IEEE, vol. 28, no. 4, pp. 32 –39, 2008.

[11] J. M. Coughlan and R. Manduchi, "Functional assessment of a camera phone-based wayfinding system operated by blind and visually impaired users," International Journal on Artificial Intelligence Tools, vol. 18, no. 3, pp. 379–397, 2009.

[12] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ser. ISMAR '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 1–10.

[13] T. Nishimura, Y. Nakamura, H. Tomobe, T. Kurata, T. Okuma, and Y. Matsuo, "Location estimation using auditory signal emitted and received by all objects," in Networked Sensing Systems, 2007. INSS '07. Fourth International Conference on, 2007, p. 295.

[14] X. Chen, Y. Shi, and W. Jiang, "Speaker tracking and identifying based on indoor localization system and microphone array," in Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on, vol. 2, May 2007, pp. 347 –352.

[15] O. Vinyals, E. Martin, and G. Friedland, "Multimodal indoor localization: An audio-wireless-based approach," in Semantic Computing (ICSC), 2010 IEEE Fourth International Conference on, 2010, pp. 120–125.

[16] M. Papandrea, "Multimodal ubiquitous localization: a gps/wifi/gsmbased lightweight solution," in World of Wireless, Mobile and Multimedia Networks Workshops, 2009. WoWMoM 2009. IEEE International Symposium on a, 2009, pp. 1 –3.

[17] S. Zirari, P. Canalda, H. Mabed, and F. Spies, "Combined indoor and outdoor DOP criteria helpful to position and dimension," in IPIN 2010, IEEE int. conf. on Indoor Positioning and Indoor Navigation, Zurich, Switzerland, Sep. 2010, pp. 197–198.

[18] Open Mobile Alliance, "Mobile location protocol," version 1.2.1, 2004..

[19] M. Henning and M. Spruiell, Distributed Programming with Ice, Revision 3.4, 2010.

[20] Object Management Group, "Property service specification," version 1.0, 2000

[21] F. Villanueva, D. Villa, M. Santofimia, F. Moya, and J. Lopez, "A framework for advanced home service design and management," Consumer Electronics, IEEE Transactions on, vol. 55, no. 3, pp. 1246 –1253, 2009.

[22] Open Geospatial Consortium Inc., "OpenGIS implementation standard for geographic information - simple feature access - part 2: Sql option," Candidate Version 3.1, 2010.

[23] Y. Shafranovich, "Common Format and MIME Type for Comma-Separated Values (CSV) Files," RFC 4180 (Informational), Internet Engineering Task Force, Oct. 2005.