Integrating Location and Context Information for Novel Personalised Applications

Matthias Strobbe[1], Olivier Van Laere[1], Femke Ongenae[1], Samuel Dauwe[1],
Bart Dhoedt[1], Filip De Turck[1], Piet Demeester[1] and Kris Luyten[2]

[1]Ghent University - IBBT
Department of Information Technology
Gaston Crommenlaan 8 bus 201, 9050 Gent, Belgium

[2]Hasselt University - tUL - IBBT
Expertise Centre for Digital Media
Wetenschapspark 2, B-3590 Diepenbeek, Belgium

Contact author: Matthias Strobbe (matthias.strobbe@intec.ugent.be)
Keywords: Location-Based Services, Context-Awareness

Abstract— **With the rapid spread of GPS enabled smart phones and the fact that users are almost permanently connected to the Internet, we see an evolution towards applications and services that adapt themselves using the user's context. To facilitate the development of such intelligent applications, new enabling platforms are needed to collect, distribute and exchange context information. We present CASP, a Context Aware Service Platform taking care of the aggregation and abstraction of context information. Three use cases in as many different domains are detailed: a personal content service, a desk sharing office service and a person oriented nurse call system. These services combine different kinds of context information in an easy way using the proposed platform.**

I. INTRODUCTION

Today, location and context based services are well established. One of the most well-known examples is a GPS based navigation system, such as the one widely used in cars. Another example is a context based city guide that gives information about interesting locations in the neighbourhood of the user taking the user's preferences into account [1]. Other popular applications include friend finder applications, news and weather services tailored to the user's current location or music recommendation services (such as Pandora or the Genius playlists in iTunes).

Due to the rapid spread of GPS enabled smart phones and accompanying application stores (such as the immensely popular iPhone and App Store) the market of location and context based services is expected to continue to expand the next years. We expect to see various new context-aware applications in a wide range of domains. To name a few: marketing campaigns containing special offers that are tailored to the user's location and interests or location-based digital coupons, context aware information services (for example relevant local traffic information or information about the neighbourhood taking into account the user's current activity), location-aware social networks and improved recommender systems.

All these new user-centric services will transform our homes, cars, offices, cities, etc. into smart environments where highly adaptive services are dynamically deployed, updated and removed.

An inherent problem with context information is that it is often incomplete and/or inaccurate. There might also arise inconsistenties when several context sources deliver the same kind of information, for example a system that uses GPS signals for outdoor localization and WiFi signals for indoor localization. Developing context-aware services is labor-intensive in terms of collecting and combining different kinds of context information to high level information. Facilitating the development, deployment and management of these services requires new context-aware service environments. An important component of these environments will be an enabling service dealing with the collection, distribution and exchange of context and location information.

To address these challenges we developed CASP, a Context Aware Service Platform that takes care of the aggregation and abstraction of context information using ontologies for representing the information and rule-based reasoning for validating information and deriving high-level contexts.

Before discussing context frameworks and the architecture of CASP, we give an overview of some common context sensing techniques and technologies. We then present three use cases in different domains illustrating the capabilities of CASP to derive new information combining different context aspects (personalized content service), handling several context sources that deliver the same kind of information (desk sharing office use case) and the aggregation and abstraction of lots of different context aspects according to a formal context model (person oriented nurse call system).

II. Context sensing

Two key aspects of a person's context are his/her location and profile. Both have been subject to a lot of research resulting in a wide variety of techniques for determining the location of a user and building up a user profile.

A. Location Determination

Location systems generally use one or more techniques to locate objects, people or both. There are three principal techniques for automatic location sensing: triangulation (exploiting the geometric properties of triangles to compute the location of an object), scene analysis (analyzing a visual image or mapped electromagnetic characteristics) and proximity (sensing that an object is near a sensor with a known location).

Global Positioning System (GPS) currently is the most common and accurate means of outdoor location determination generally available. Main disadvantage is the dependency on a direct line of sight to the satellites making it useless for indoor localization.

A simple localization method often used in mobile phone networks is cell identification. The accuracy is rather low and depends on the size of the cells (typically tens to hundreds of meters in urban areas and several kilometers in rural environments).

In infrared based systems persons wear a small infrared badge which emits a unique pulse signal. Ultrasound based systems use an ultrasound time-of-flight lateration technique in combination with radio signals for time synchronization purposes and Microsoft's EasyLiving [2] uses real-time stereo cameras. A common aspect of these systems is that they are expensive due to the significant installation and maintenance costs.

A lot cheaper are visual code systems using 2D barcode tags. These barcodes can be used to encode information. Decoding the tags only requires a low-resolution camera and few processing power, so mobile devices containing a built-in camera can decode them and can infer their location by matching the ID of the tag with a stored location in a database or by decoding the location information embedded in the tag [3]. Such a system requires user interaction to scan the tags, but the advantages are a high accuracy, a low cost, as barcodes can be printed on paper or displayed on electronic screens, and a low investment cost in infrastructure as more and more people own a cell phone with built-in camera.

A variant of the barcode tags are Radio-Frequency IDentification (RFID) tags [4] that can be scanned without user interaction. Accuracy depends on the type of RFID tag: passive systems with a short range (maximum 3m) are available, as well as active systems with a range of over 100m. Downside to RFID is the cost of the readers.

By far the most popular amongst indoor location systems are the ones using WiFi technology. The reason for this is that a lot of WiFi infrastructure is available that is relatively low priced. Some systems use a radio propagation model to determine the distance to the various access points. Multipath distortion limits the accuracy of this method. Alternatively fingerprinting could be used where a radio frequency map is constructed during an off-line phase. Disadvantage of this technique is the cost to build up the radio map. An example of a WiFi based location service is Skyhook [5] where the MAC address of a nearby access point is matched with a location in Skyhook's database. If available also GPS data and information from mobile phone networks is used to improve localization.

Other radio technologies include Bluetooth and Ubisense. Bluetooth devices typically have a signal range of about 10m (class 2) which makes them suitable for proximity based positioning. Ubisense [6] developed a positioning system based on ultra-wideband (UWB) radio technology. As UWB signals are much less affected by multipath distortion than conventional RF systems, a very high accuracy can be achieved.

Table I gives an overview of the accuracy and the used location estimation technique of each presented positioning technology.

TABLE I
SUMMARY TABLE CONTAINING THE MOST IMPORTANT CHARACTERISTICS OF EACH POSITIONING TECHNOLOGY

Technology	Accuracy	Location Estimation
GPS	15m	Trilateration
WAAS/DGPS	3-5m	Trilateration
Mobile phone	50m-10km	Proximity, Triangulation
Infrared	5-10m	Proximity
Ultrasound	1-10cm	Trilateration
Vision	1cm-1m	Scene Analysis
Visual Codes	1m	Proximity
RFID	5cm-5m	Proximity
WiFi	2-100m	Trilateration, Proximity, Scene Analysis
Bluetooth	2-10m	Proximity
Ultra-wideband	10-15cm	Trilateration

B. User Profiling

Acquiring an accurate user profile is a cumbersome task, as users usually are reluctant to complete (extensive) forms or give explicit feedback and often forget to mention essential details. Moreover, user interests may change over time, which is hard to cover by filling in a form once. Ideally a user profile should be obtained in an automatic non-intrusive way.

Examples of automatically obtained user profiles can be found in recommender systems, of which a well known example is the Amazon recommendation engine. Based on the matching between the shopping cart items of a user and shopping carts of other users, items that might interest the user are suggested. Another popular technique is analyzing the browse and search history of a user to improve web search results [7], [8]. An alternative source of information are tags posted on social websites by the users [9]. In [10] a method is described for building up user profiles using implicit input provided by online analysis of the exchanged instant messaging traffic and the consumption of user generated content.

III. Context Platform

In recent years a number of context platforms have been developed to relieve application developers from the aggregation and abstraction of location and context information, and the derivation of high-level contexts. One of the first platforms was the Context Toolkit [11], a Java framework allowing the rapid prototyping of sensor based context-aware applications. However, the Context Toolkit does not focus on a context model for exchanging information. In recent years a number of context platforms have been developed with formal context models based on ontologies. Examples are SOCAM [12] and COCA [13]. Using ontologies provides semantic uniformity and interchangeability of context information in a heterogeneous setting such as pervasive computing environments.

Our platform CASP belongs to this latter series of platforms. Figure 1 gives an overview of the architecture of CASP.

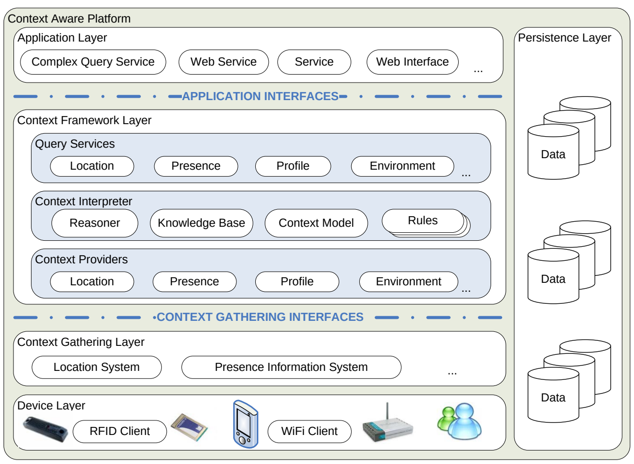

Fig. 1. Overall CASP architecture

The *persistence layer* ensures the persistence of context information. This comprises static context information about users, devices, the environment, etc. More dynamic information, such as positions of users, can also be stored. In this way, the history of context information can be exploited and trends or habits can be derived.

The *device layer* includes all devices (and software components on those devices) that deliver context information. For example, a WiFi client on a PDA can measure the received signal strengths of access points in the surroundings and send this information to the location system.

The *context gathering layer* takes care of the acquisition of specific context information. For instance location or profile info. To improve the modularity of the platform, context gathering interfaces are defined between the context gathering layer and the context framework layer for important types of context information such as location, presence, profiles, etc. It is sufficient to implement such an interface to add a context gathering component to the platform.

The *context framework layer* is responsible for the aggregation of the context information according to a formal context model and the derivation of implicit information by reasoning. Context information coming from the context gathering layer is translated to OWL [14] (an ontology language proposed by W3C) by the context providers and interpreted by the context interpreter. All context providers implement a common interface making them easily pluggable into the platform. The context interpreter consists of a knowledge base where the information from the context providers is gathered and a rule-based reasoner to derive high-level knowledge. The rule engine provides a forward chaining, a backward chaining and a hybrid execution model. Rules can also be used for validation purposes. For example, if several context providers deliver the same kind of information, there will probably be inconsistencies from time to time. Based on reliability and accuracy parameters, a decision can be made on the correctness of the information. The query services enable and facilitate the retrieval of context information. They translate OWL constructs to objects and expose an application interface towards the services. This relieves application developers from writing error-prone queries and translating the results to objects themselves.

The *application layer* contains interfaces and web services that permit easy querying of the knowledge base and applications that use the context information.

Each layer, is designed in a modular way with a limited number of dependencies using the OSGi platform as basis. If a context gathering component is added or removed from the context gathering layer, the context provider in the context framework layer will detect this and the context information in the knowledge base will be updated. On the other hand, if a context provider is removed from the context framework layer, the context gathering components will notice that there is no suitable context provider available and no information will be exchanged. This way, components can be added, started, updated, removed or stopped while the rest of the platform keeps running. The modular design allows a deployment in a distributed manner.

IV. Use Cases

In this section we present three use cases illustrating the use of the CASP framework.

A. Personalized Content Service

In this first use case we combine the location of a user with personal interests to provide content that might interest the user in his or her current situation. Such content could for example be shown on a personal start page like iGoogle or in the form of advertisements.

The location of the user is abstracted to a presence state saying that the user is ‘atHome’, ‘atWork’, ‘atTennisClub’, etc. Based on this presence state, content is automatically fetched from the Internet and suggested to the user. At work this might be RSS news feeds and at home entertaining movies from YouTube. To select content from a certain source (e.g. YouTube), the personal interests of the user are used. Besides location and profile information, also the capabilities of the device of the user are taken into account. For example on a mobile phone, no movies are shown as this might require too much processing capabilities. Instead pictures could be shown.

The Personalized Content Service is implemented as a web application. Figure 2 gives an example. It shows Flickr pictures for user *olivier@achterberg* who is logged in on his desktop.

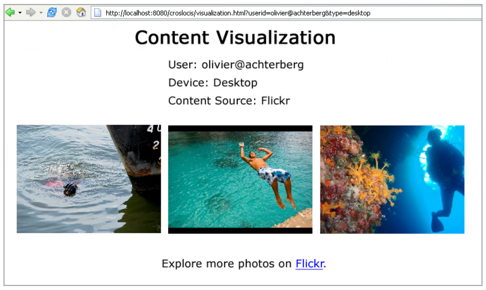

Fig. 2. Screenshot of the Personalized Content Service: pictures from Flickr matching with the interest 'scuba diving' of user 'olivier@achterberg' are shown.

As Olivier is interested in scuba diving, some pictures related to diving are shown.

Derivation of the high level presence states of the user is accomplished by the reasoning component of CASP. Rules look at the location of the user and match these with the coordinates of known places for the user like his home address, work address, sports club location, etc.

A second group of user specific rules determines the preferred content source of the user for a certain situation taking his presence state and the device capabilities of the device he's using into account. Besides determining the content source, extra parameters can be set, such as extra keywords to be used when searching for actual content (apart from the interests in the personal profile) or some important keywords to be exclusively used for the content search in a special situation. For example when a user is at the diving club he only wants to see pictures about diving. This is illustrated with the following rule for user 'olivier@achterberg'.

```
[content_Olivier_at_diving_club:
(?x rdf:type User)
(?x userID 'olivier@achterberg')
(?x hasProfile ?y)
(?y hasContentSource ?z)
// User is at the diving club
(?x presenceStatus 'atDivingClub')
// No pictures from Flickr are shown yet
noValue(?z name 'flickr')
// The user's content source is set to Flickr
// and only pictures about diving are shown
-> fireRuleEvent('contentsource'
'olivier@achterberg'
'flickr?importantkeyword=diving')]
```

Using these rules it is very easy to add new situations, cope with changing user preferences or take new kinds of context information into account (an example could be the specific activity of the user).

B. Desk Sharing Office Environment

This second use case was developed for the premises of a large company in Belgium. After moving to a new location desk sharing was introduced. In a desk sharing environment, employees no longer have a dedicated desk anymore. When they arrive at work, they can choose whatever desk available in the building to start working at. In such an environment, several problems arise. Visitors who have an appointment with an employee do not know where to look. Even an employee who needs a colleague for a meeting is not always able to locate him or her.

The desk sharing office use case is a web based application consisting of several services. Its main function is to allow employees or visitors to locate other users and to see how someone can be reached. A user has to fill in the name of the contact and as a result the path from his or her current location to the location of the contact, together with the status of that person and some extra information such as email address, phone number, etc. is returned.

When the person the user is looking for is not available or present, a default location can be returned, e.g. the reception desk or a colleague who might be able to help. The application provides the presence status of persons, which allows to see who is present or available. If a person is available, one can click on a telephone icon to setup a phone call to the person through VoIP. A call is then set up between the phones that are located the closest to both persons.

To enable the different services offered by the web application, an ontology for an office environment has been designed. This ontology is shown in figure 4. The most important concept is a *Person* containing a lot of properties relevant to his job. Each *Person* has one or more *Personal Devices*. Examples are a laptop, a PDA or a mobile phone.

To determine the location of a user, the *Personal Devices* of that user are tracked using wired and wireless techniques, making use of the existing network infrastructure to limit costs. The laptop computer of the employees can be located by the network cables that connect them with the company network. Such a network cable connects the portable with a certain port of a network switch. Each port corresponds with a specific location. Via the Simple Network Management Protocol (SNMP) the mapping from this port number to the MAC address of the connected device can be queried. Thus, by combining the mappings from the MAC address of the portable to the switch port number and this port to a certain location, the location of the portable can be determined.

A second localization technique uses the company's WiFi network. This allows to determine the location of a user walking around with a WiFi enabled PDA or smartphone. The signal strengths originating from the access points are measured and compared with a radio map. To improve the accuracy we also use a user motion model taking the last known position, the current movement direction and speed into account. More information on the use of motion models for localization can be found in [15].

As an employee can have several *Personal Devices*, possibly tracked by different location systems, a person might have several locations. These locations can be different at the same instance of time, e.g. an employee can be walking around the building with his PDA, while his laptop is still on his desk. Moreover, the accuracy of wireless location determination techniques is not perfect. To infer the most probable location of a user we defined a number of rules that are executed by

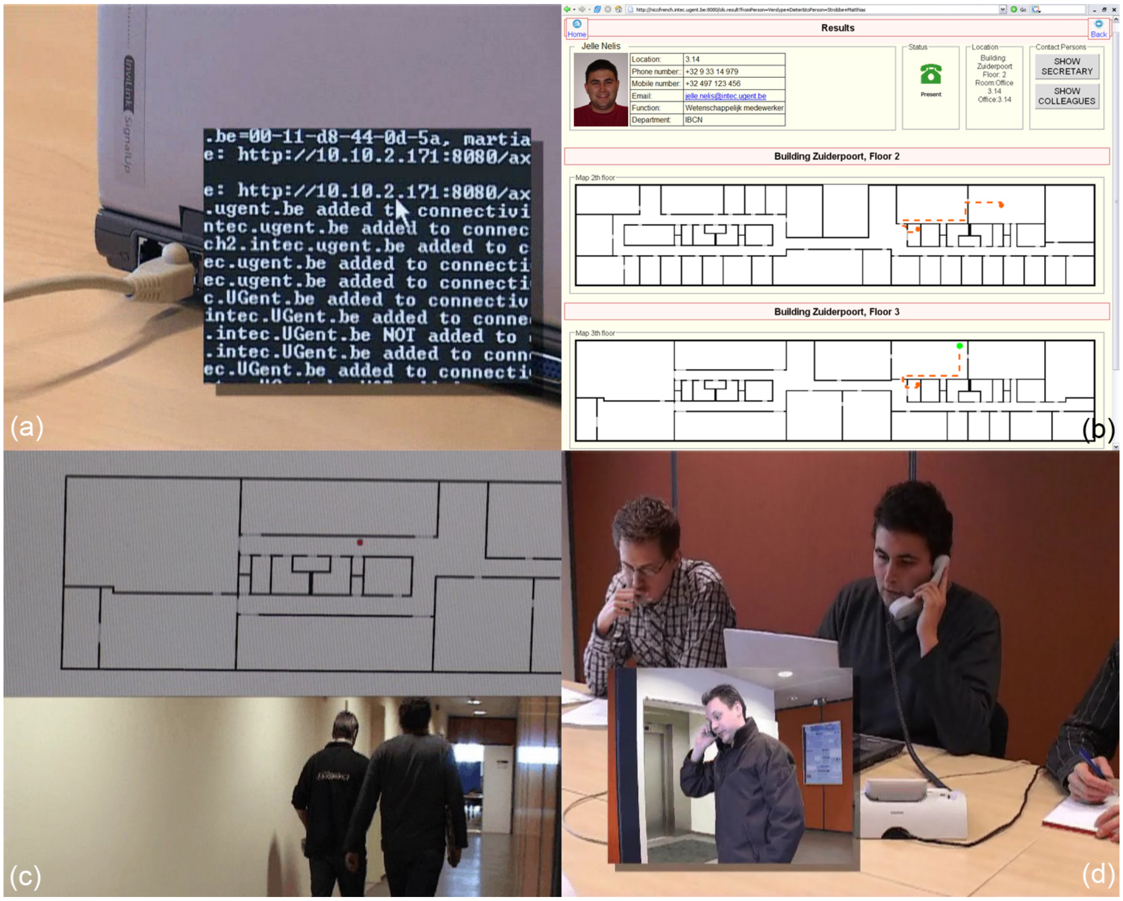

Fig. 3. Desk sharing office use case. (a) SNMP based localization: a laptop plugged into the company network is detected and localized. (b) Screenshot of the web application. (c) WiFi based localization: walking employee is tracked. (d) By clicking on the telephone icon, an automatic call is set up between visitor and employee.

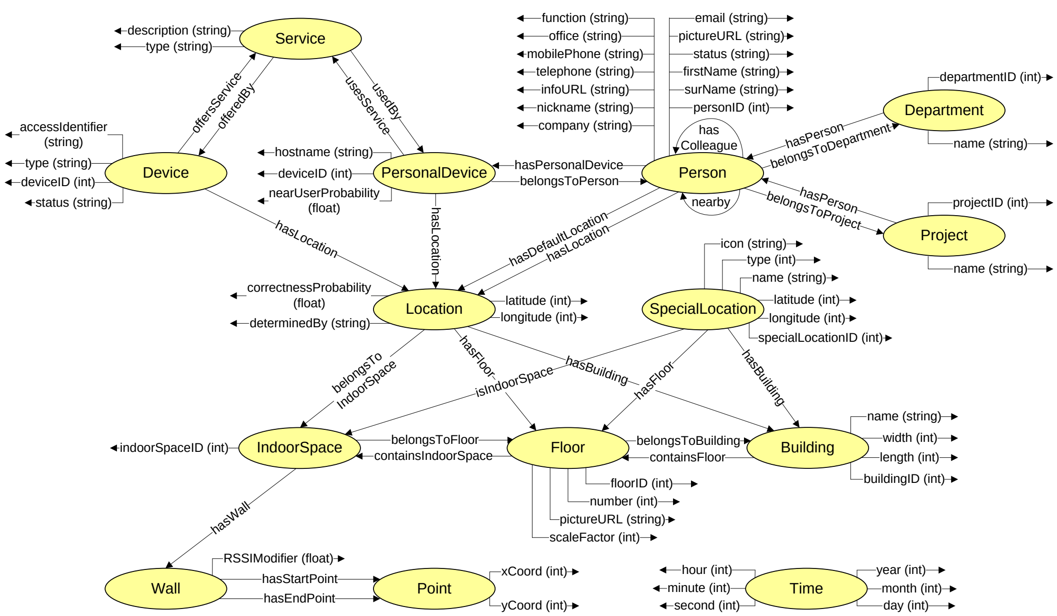

Fig. 4. Ontology we designed for the desk sharing office use case.

the reasoner of CASP.

These rules take advantage of some of the defined properties of the ontology: the *nearUserProbability* for a device indicates the probability that a device is in the vicinity of a user. This probability will typically be higher for a PDA than for a portable and even higher for a mobile phone. Furthermore the *correctnessProbability* property was added to every location entity, indicating the accuracy of the location determination technique. The rules we defined take the values associated with these properties into account and choose the location with the highest *correctnessProbability* of the device with the highest *nearUserProbability*.

The following rule illustrates this. It adapts the location of a user when a personal device is tracked by another location determination technique with a better *correctnessProbability*.

```
 [update_location_person_correctness_probability:
(?x rdf:type Person)
(?x hasLocation ?loc1)
(?x hasPersonalDevice ?y)
// Personal Device ?y has 2 locations
(?y hasLocation ?loc1)
(?y hasLocation ?loc2)
// Each location has a correctnessProbability
(?loc1 correctnessProbability ?cp1)
(?loc2 correctnessProbability ?cp2)
greatherThan(?cp2 ?cp1)
// As ?loc2 has a greater correctnessProbability,
// ?loc2 is set as the location of user ?x
-> setLocation(?x ?loc2)]
```

As we use our modular platform CASP for implementing this algorithm, it could be easily extended if extra context information is available. Suppose an extra component that measures user activity on a device becomes available. By plugging in an extra context provider and extending the ontology, this information would be available in the knowledge base. Then we can change the rules to give higher priority to devices with recent activity when determining the devices that are most likely in the neighbourhood of the user.

Rules are also used for inferring presence information which is modeled in the ontology by the *status* property of a *Person*. When a user has a location, his status automatically becomes ‘online’. When he leaves the building, his status is changed to ‘offline’. When a user manually changes his location to ‘busy’ or ‘away’, the rules will assign a user specified default location associated with the status, e.g. a colleague or the reception desk. The ontology contains also information about the structure of the building which for example allows to automatically change the status of the user to ‘inMeeting’ when he is located inside a meeting room.

Figure 5 shows the throughput of the developed web application when requesting a page similar to figure 3 (b). About 17 requests can be processed by the server. This throughput is due to the dynamic information that needs to requested and calculated: 28% of the processing time is used for requesting location and personal information about the requester and the person that is searched for from the knowledge base, 10% for calculating the shortest path using Dijkstra’s algorithm and 61% for drawing the maps and path. Especially the creation of the overlay image for the path takes a lot of time (38%). Of course, further optimisations are possible in a real-life deployment, including caching of maps, use of more powerful graphical libraries for map generation, pre-calculation of maps and deployment on additional servers for scalability reasons. In comparison, the static homepage of the application can be requested almost 1000 times per second.

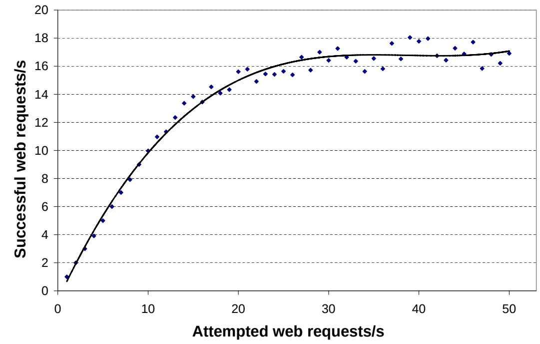

Fig. 5. Web server request throughput for page containing a lot of dynamic information.

C. *Person Oriented Nurse Call Management System*

This final use case studies the transition from a place-oriented to a person-oriented nurse call system. Location information is combined with personal characteristics, status, staff competences and infrastructure information.

Current nurse call systems are organised according to rooms: each room (or bed) has a fixed button which can be used by the patients to call a nurse. In such a place-oriented system, all nurses belonging to the group responsible for the room are called when a patient pushes a button in his/her room. Each nurse decides on his/her own if he/she is going to interrupt his/her current task to answer the call or not. The nurse who reaches the room first handles the call. Here, two important assumptions are made: the patient must still be in the room and it must be the patient who lies in the room that made the call.

The following evolution is observed: each patient receives a mobile button so that they can walk around freely and still make calls. The system becomes person-oriented. This evolution implies numerous changes, e.g., the nurse has to go to the exact location of the patient (not necessarily his/her room), the patient can make calls from anywhere in the hospital and so on. This huge impact is comparable to the introduction of the mobile phone. In the past we used to call to a telephone (a place) and ask for the correct person. Now we call a mobile phone and we immediately expect to have the right person on the line.

The transition to a person-oriented system demands a good understanding of the context of a hospital to achieve a more efficient nurse call algorithm. As patients can move all over the hospital while making a call, nurses working in the department of the patient might be far away and thus may not be the best candidate to answer the call. It is important to have access to context information such as the location and risk factors of the patient, the characteristics of the hospital staff and the infrastructure of the hospital.

For this use case an ontology was constructed by studying relevant situations and extracting the necessary context information. The details about this ontology can be found in [16]. Based on this ontology a new nurse call algorithm was designed. This algorithm first determines which of the following kinds of calls has been made and acts accordingly:

- *Urgency call* - The closest person who is a member of an urgency team and who is not already handling an urgency call is sought. Then the entire urgency team to which this person belongs is called. The priority here lies in finding a person who is near instead of a person who is free.
- *Normal call*
  1) First, the nurse who is responsible for this patient is sought. If he is free and in the vicinity, he is called. If he is occupied and his current task has a lower priority, he is also called.
  2) If the responsible nurse cannot be called, all the nurses are sought who work in the department from where the call originated. From this group all the nurses who are not free, are not willing/qualified to treat the patient or are not in the vicinity are removed. From the remaining nurses, the one who has the most characteristics in common with the preferences of the patient is chosen. If no characteristics are given, the nurse who is the closest to the patient is chosen.
  3) If the result is empty, the previous step is repeated beyond the nurses of the department.
  4) If the result is still empty, this means that there are no available nurses in the direct vicinity. The distance becomes a deciding factor at this moment. So the qualified nurse that is closest to the origin of the call and who is free and willing to treat the patient is selected.
  5) If this still does not offer a solution, all the nurses in the hospital are considered and the one who is closest to the patient is called.

Note that this algorithm differs significantly from the place-oriented one which only looks at which patients (actually rooms) are allocated to which nurses. In the new algorithm many more factors are taken into account. It looks at the characteristics and the status of the staff members, the risk factors and preferences of the patients, the priority of the call and so on.

This algorithm is implemented using the reasoner component of CASP. Rules are activated when a new call is added to the knowledge base. The following rule gives an example of the rule that reacts to a normal call that is launched (status is *active* and no staff member has been called to answer the call). If the conditions of the rule are fulfilled, the functor findHelper() is called. This functor follows the earlier stated algorithm to find a correct staff member to handle the call.

```
[insert_nurse_normalcall:
(?x rdf:type Normal)
(?x has_status ?status)
(?status Kind 'Active')
noValue(?x treated_by_nurse)
-> findHelper(?x)]
```

Other rules handle the timeout of a call and update the status of the call when a staff member indicates that he/she is going to answer the call or ends a call (patient is treated).

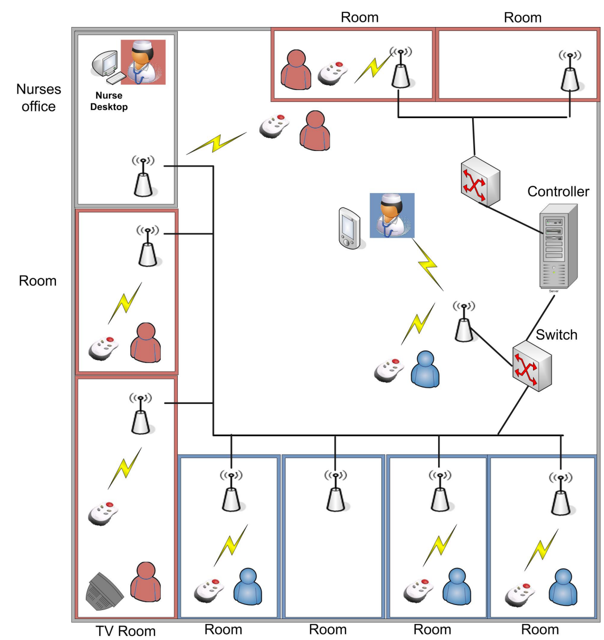

Fig. 6. Simulation setup: nurses are responsible for the patients with the same colour. Patients can make calls from anywhere in the hospital. In case of a call, the most suitable nurse is notified to go as soon as possible to the location of the patient.

To test the advantages of the system, simulations were set up. Typical situations within a hospital were simulated and the results were studied and compared with the place-oriented system. Figure 6 gives an example of such a setup. The nurses are responsible for the patients with the same colour. As patients can make calls from anywhere in the hospital they would not always be treated by the responsible nurse, but by the most suitable nurse on that moment. The performance of these simulations was tested and showed that a suitable nurse is always notified within a very short time (50 ms on average). More details can be found in [16]. The simulations reveal a number of advantages of the person oriented system in contrast with the place oriented system, by taking several context aspects into account:

- The exact location of the patient is known. In the place oriented system, a nurse just goes to the room in the hope the patient is still there.
- A staff member knows which patient made the call.
- The workload is distributed better among the different staff members, because it is known which staff members are already busy with a task and which are not.
- Patients get their treatment faster.
- The wishes, characteristics and the risks of the patient are taken into account.

- It is known for sure that the called nurse is in the vicinity (or is the nearest possible person who could be called).
- The system takes care of interrupted tasks. This leads to fewer forgotten tasks and lesser work pressure on the staff.
- The priorities of the tasks allow the nurses to make a more well-funded decision on whether to interrupt the current task or not.

V. Conclusions

In the near future our environment will be truly pervasive where users are surrounded by intelligent devices and use services that are perfectly tailored to their location, interests, current activity, status, etc.

These services will combine different kinds of context information to give the user an experience that perfectly fits with his current context at that moment.

In this paper we presented CASP, a platform that takes care of the aggregation and abstraction of context information. Using rule-based reasoning different kinds of information can be easily combined to high level context information. By presenting three use cases in diverse domains we showed the strength of our platform for the development of new intelligent context-aware services.

Our platform uses ontologies for modeling the location and context information. The structure of the ontologies used for the presented use cases is predefined and fixed. Only the instances of concepts and relationships between concepts change. In the future context modeling needs to further improve with the dynamic import and removal of existing ontologies based on situation and user. This will stimulate knowledge reuse and at same time keep the system manageable. The existing reasoners still have performance issues to reason on large amounts of ontological data. In the future this performance needs to be improved to be useful for large-scale applications.

To further increase the performance of the presented use cases, user experiments could be conducted to assess the quality of the response as perceived by the users. For example to optimize the user motion model used for localization in the desk sharing office use case.

In the future we also expect to see more and more datamining on context information to derive habits and trends. Such information could be easily stored in our platform in the form of rules and be combined with real-time information.

Acknowledgment

Matthias Strobbe is funded by the Fund for Scientific Research - Flanders (F.W.O.-Vlaanderen) as a Ph.D. researcher. Femke Ongenae is funded by the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT) as a Ph.D. researcher.

References

[1] K. Cheverst, N. Davies, K. Mitchell, A. Friday, and C. Efstratiou, “Developing a context-aware electronic tourist guide: some issues and experiences,” in *CHI ’00: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM, 2000, pp. 17–24.

[2] B. Brumitt, B. Meyers, J. Krumm, A. Kern, and S. A. Shafer, “Easyliving: Technologies for intelligent environments,” in *HUC*, 2000, pp. 12–29.

[3] O. Van Laere, M. Strobbe, P. Leroux, B. Dhoedt, F. De Turck, and P. Demeester, “Enabling platform for mobile content generation based on 2d barcodes,” in *Proceedings of ICOMP2008, the 2008 International Conference on Internet Computing (part of Worldcomp2008)*, 2008, pp. 209–214.

[4] RFID.org, “Rfid.org,” [online], 2010, http://www.rfid.org/.

[5] Skyhook, “Xps overview,” [online], 2010, http://www.skyhookwireless.com/howitworks/.

[6] Ubisense, “Ubisense system overview,” *Technical Whitepaper*, 2009. [Online]. Available: http://www.ubisense.net/pdf/fact-sheets/products/software/Precise-Location-EN090624.pdf

[7] M. Daoud, L. Tamine-Lechani, M. Boughanem, and B. Chebaro, “A session based personalized search using an ontological user profile.” in *SAC*, S. Y. Shin and S. Ossowski, Eds. ACM, 2009, pp. 1732–1736.

[8] S. Stamou and A. Ntoulas, “Search personalization through query and page topical analysis,” *User Modeling and User-Adapted Interaction*, vol. 19, no. 1-2, pp. 5–33, 2009.

[9] M. Strobbe, O. Van Laere, B. Bogaerts, S. Dauwe, B. Dhoedt, F. De Turck, and P. Demeester, “Tag based generation of user profiles,” in *International Conference on Internet Computing*, 2009, pp. 125–131.

[10] M. Strobbe, O. Van Laere, S. Dauwe, B. Dhoedt, F. De Turck, P. Demeester, C. van Nimwegen, and J. Vanattenhoven, “Interest based selection of user generated content for rich communication services,” *J. Netw. Comput. Appl.*, vol. 33, no. 2, pp. 84–97, 2010.

[11] A. K. Dey, “Providing architectural support for building context-aware applications,” Ph.D. dissertation, Georgia Institute of Technology, 2000. [Online]. Available: http://hdl.handle.net/1853/9232

[12] T. Gu, H. K. Pung, and D. Q. Zhang, “A service-oriented middleware for building context-aware services,” *Journal of Network and Computer Applications (JNCA)*, vol. 28(1), pp. 1–18, 2005.

[13] D. Ejigu, M. Scuturici, and L. Brunie, “Hybrid approach to collaborative context-aware service platform for pervasive computing,” *JCP*, vol. 3, no. 1, pp. 40–50, 2008.

[14] W3C, “OWL Web Ontology Language Overview,” [online], 2004, http://www.w3.org/TR/owl-features.

[15] D. Ashbrook and T. Starner, “Using gps to learn significant locations and predict movement across multiple users,” *Personal Ubiquitous Comput.*, vol. 7, no. 5, pp. 275–286, 2003.

[16] F. Ongenae, M. Strobbe, J. Hollez, G. D. Jans, F. D. Turck, T. Dhaene, P. Demeester, and P. Verhoeve, “Design of a semantic person-oriented nurse call management system,” *Int. J. Web Grid Serv.*, vol. 4, no. 3, pp. 267–283, 2008.