The Knowledge Engineering Review, Vol. 23:1, 7–19. © 2008, Cambridge University Press
doi:10.1017/S0269888907001300 Printed in the United Kingdom

Situational reasoning for task-oriented mobile service recommendation

MARKO LUTHER[1], YUSUKE FUKAZAWA[2], MATTHIAS WAGNER[1] and SHOJI KURAKAKE[2]

[1]*DoCoMo Euro-Labs, Landsbergerstr. 312, 80687 Munich, Germany*
e-mail: luther@docomolab-euro.com, wagner@docomolab-euro.com;
[2]*NTT DoCoMo Inc., 3-5 Hikari-no-oka, Yokusuka, Kanagawa, 239-8536 Japan*
e-mail: fukazawayuu@nttdocomo.co.jp, kurakake@nttdocomo.co.jp

Abstract

We study the case of integrating situational reasoning into a mobile service recommendation system. Since mobile Internet services are rapidly proliferating, finding and using appropriate services require profound service descriptions. As a consequence, for average mobile users it is nowadays virtually impossible to find the most appropriate service among the many offered. To overcome these difficulties, task navigation systems have been proposed to guide users towards best-fitting services. Our goal is to improve the user experience of such task navigation systems making them context-aware (i.e. to optimize service navigation by taking the user's situation into account). We propose the integration of a situational reasoning engine that applies classification-based inference to qualitative context elements, gathered from multiple sources and represented using ontologies. The extended task navigator enables the delivery of situation-aware recommendations in a proactive way. Initial experiments with the extended system indicate a considerable improvement of the navigator's usability.

1 Introduction

Within the growing market for mobile Internet, NTT DoCoMo is today providing services to over 50 million mobile phone subscribers in Japan. The majority of these users enjoy widely diverse contents such as entertainment services (ring-tone downloads, games, etc.), transaction services (money transfer, airline reservation, etc.) and information services (weather forecast, maps and local information, etc.) through DoCoMo's high-speed 3G mobile network. Already today, the number of commercial i-mode sites—DoCoMo's brand of mobile Internet services—ranges in the region of many tenth of thousand. With 4G networks at the horizon that promise still substantially higher bandwidth for data transmissions, the market for services with rich content is expected to expand further.

Key to support such growth is the availability of intelligent service platforms that mediate between services and users by observing the users' activity. These platforms have to assist the user in selecting the most appropriate service from the fast growing service pool to support their real-world activities, anytime and anywhere.

Our previously developed task-based service retrieval system for the non-expert mobile user makes it easy to retrieve appropriate services for tackling the users' challenges in managing his or her everyday life (Naganuma & Kurakake, 2005). The term task refers here to 'what the user wants to do' as an expression of the users current activity. A task knowledge base contains semantic descriptions of potential activities and links to corresponding services that may be helpful.

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

Although this system enables effective service retrieval, it behaves passive in requiring a user's initial input to trigger the problem-solving process.

We propose a proactive extension of our basic system that suggests tasks and services actively, without the need for initial user input. This is achieved by the integration of a situation engine and a situation-based task filter, meant to expose only those tasks that are relevant for a user in a given situation. Taking the user's situation into account avoids the necessity of an initial task query. This leads to a considerable improvement of the navigator's usability, especially for non-expert users who are often not willing to input queries.

The abstract characterization of a user's situation is computed by inference mechanisms on several pieces of context information gathered from multiple context sources (Luther *et al.*, 2005b). We formulate high-level qualitative context elements in the Web Ontology Language (OWL) (McGuinness & van Harmelen, 2004) and concrete situations as instances within the assertional component (Abox) of a situation ontology. To profit from sound, complete and high-performance classifiers such as FaCT++ (Tsarkov & Horrocks, 2005), Pellet (Sirin *et al.*, 2006b) and Racer (Haarslev & Möller, 2003), we restrict ourselves to the OWL DL fragment of OWL. To separate concerns we assume that probabilistic aspects of context representation and reasoning are dealt with at lower representation levels applying Bayesian Networks, Markov Clustering or Fuzzy Logics.

The rest of this paper is organized as follows. After discussing related work in the field of ontology-based context reasoning in the next section and presenting our vision on the interplay between context and ontologies in Section 3, we introduce our task-based service navigator application together with some usage scenarios in Section 4. Its overall system architecture is presented in Section 5. Details on the underlying task-oriented service navigation system, and the context representation and classification-based reasoning approach are given in Section 6 and Section 7, respectively. Finally, we report on the experiences gained from this development in the closing section.

2 Related work

Several projects consider the use of ontologies as a key requirement for building context-aware applications. Closely related to our approach is the work done in the CALI project (Khushraj & Lassila, 2004) as it explores the use of Description Logics (DL) (Baader *et al.*, 2003). To overcome the limitations of pure DL-based reasoning, a hybrid approach is proposed. However, our earlier experiments (Mrohs *et al.*, 2005) indicate that the suggested loose coupling of a DL reasoner with an external generic rule engine leads to serve performance problems. To achieve completeness both reasoners have to be applied successively until no new facts have been derived. Furthermore, it remains unclear how consistency can be guaranteed taking both the knowledge base and the rule base into account.

Other approaches such as CONON (Wang *et al.*, 2004) and SOUPA/CoBra (Chen *et al.*, 2005) solely rely on rule-based reasoning which cannot be complete for OWL (not even for OWL Lite (de Bruin *et al.*, 2005)) and easily leads to undecidability, as generic rules can be used to simulate role value maps (Grosof *et al.*, 2003).

CONON is an OWL DL encoded upper-context ontology for pervasive computing applications defining almost 200 concepts. Rule-reasoning is used to derive high-level context information and to check its consistency. To cope with the observed delay of several seconds caused by the reasoning process, complex reasoning tasks are computed offline. However, this approach is not feasible in our dynamic setup.

SOUPA, another OWL DL ontology designed for ubiquitous applications, is about the same size as the CONON ontology. Its extension CoBra-Ont is used by a context broker architecture to realize a scenario where people on a university campus come together for a meeting. To limit the reasoning overhead caused by importing standard ontologies, single concepts are mapped to foreign ontology terms. Still, the SOUPA ontology is of a rather high-complexity corresponding to $\mathcal{SHOIF}(D)$, because it contains nominals.

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

An interesting approach to speed up the rule-based inferencing on complex ontologies is to determine relevant contexts required to answer queries using the query-tree method (Lee *et al.*, 2005). It remains to be seen how this method extends to our classification-based approach.

3 Context and ontologies

Ontologies promise to play a pivotal role for different semantic-based and semantic-aware applications. This is particularly reflected in the strong research movements and ongoing standardization efforts in the area of the Semantic Web that are focusing on a machine readable, meaningful description of the elements in the World Wide Web. Besides the usage in Web applications, ontology-based and logically well-founded annotations can be beneficial in various fields. Especially systems to support context-awareness in mobile services and applications can benefit from such semantic-based descriptions.

Context has to be modelled and managed adequately to exploit context-awareness (Floréen *et al.*, 2005). For our applications context becomes primarily user-specific and is thus described from a user-specific point of view as any information that can be used to describe the current situation of the user (including fragment descriptions of other relevant entities in the user's environment). For pragmatic reasons, we are focusing on meaningful context, that is, any context which has some potential to be exploited for some (imaginable, meaningful) context-specific service or task. This can include a broad variety of ambient information such as identity, spatial, temporal and environmental information, social situation, communication resources, terminal resources, physiological measurements, activities, and personal agenda information.

To reflect the varying nature of context and to ensure a universal applicability of context-aware systems, context is typically represented at different levels of abstraction. Starting at the level of raw context sources, such as sensor devices, context data are either directly consumed as a data stream or tagged for further reference with some low-level context vocabulary. At this level of context management and context fusion, sources may compete and thus introduce inconsistency amongst competing sources. Examples include the fusion of temperature measurements from different temperature sensors in one room or the estimation of the user's location from competing GPS sources. Complementary to these low-level requirements, logical consistency amongst context items is often a required feature of higher levels of context abstraction. To this end, logically well-founded context ontologies are introduced to enable the input of the current context representations for situational reasoning. Situational reasoning (Luther *et al.*, 2005b) uses the background knowledge specified in the ontologies to infer additional high-level context elements from the given ones by automatic classification. This purely deductive process ensures consistency at a higher abstraction level.

In addition to basic ontology-supported services in context management, such as the mentioned check for consistency, the qualitative nature of ontology-based context models is helpful in many practical applications. On the one hand, this is due to the fact that users simply lack the precision and adequate computing machinery to deal with and to describe their physical context quantitatively. On the other hand, qualitative descriptions are crucial for efficient communication. Often descriptions of complex contextual setting in the physical world cannot be communicated quantitatively in a meaningful and efficient way between users or from the service to the user. Examples include the mapping of detected temperature values into a value range that reflects a comfortable temperature or the interpretation of multiple context value constellations as a complex situation of a certain type, such as a meeting situation within an office environment.

4 Situation-aware service recommendation

We build on a task-oriented service navigation system (Naganuma & Kurakake, 2005) that supports the user in finding appropriate services by querying a rich task ontology that represents common sense knowledge about typical complex tasks (cf. Section 6).

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

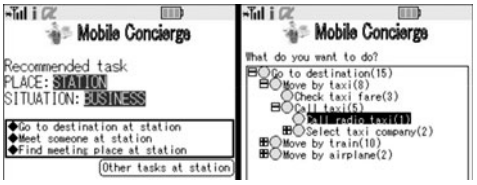


Figure 1 Situation-aware service recommender



Figure 2 Felica device

The usage of this basic task navigator is as follows. After having specified a task-oriented query such as 'go to theme park' a list of tasks that match this query is sent to the mobile device. Now the user can select the most appropriate task and a corresponding detailed task-model is displayed accordingly. In a final step, associated services can be invoked by establishing an Internet connection to the actual i-mode services.

Figure 1 shows the user interface of the situation-aware variant of the basic service recommender. To explain its functionality, let us assume the following situation.

Situation 1: Important Business Meeting at Tokyo Station

Two travellers, Dawson Campbell and his boss Fiona Davidson, arrive on a Friday morning at the Tokyo main station. Gordon Green, a project partner, is already waiting for them at the platform. The group is looking for a quick transfer to the airport.

To detect the user's location we further assume that the cell phones of Dawson, Fiona and Gordon are equipped with Felica[1] contact-less RFID tags, enabling a two-way communication with Sony's Felica Reader-Writer devices. Whenever a user puts his phone close to a Felica Reader-Writer device (e.g. to make a mobile payment at a train gate) the recommender application retrieves the corresponding location information as a semantic description of this place (cf. Figure 2). Since Sony and NTT DoCoMo just started to deploy their mobile Suica[2] system for JR East at all stations in the Tokyo region, this assumption is not a fiction but reality.

After having passed the gate at Tokyo station, Dawson's phone displays a basic list of tasks, associated with the concept *Station*. This list may include entries such as 'Prepare to ride a train', 'Buy souvenirs', 'Meet someone', etc. While displaying this task-list, Dawson's phone connects to the situational reasoning engine and updates his location to *Tokyo station*.

Before having passed the gate, no tasks are shown on Fiona's phone. Once her location has been detected, a connection to the reasoning engine is established and her current location is updated. As a result, the situation reasoner infers that Dawson Campbell and Fiona Davidson are travelling together, based on their proximity at the station. In addition, a look-up in the knowledge base reveals that Dawson and Fiona are colleagues and that the scene takes place at a weekday's afternoon.

[1] http://www.nttdocomo.co.jp/english/p_s/i/felica
[2] http://www.jreast.co.jp/suica/

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

Because Dawson is located at a *public place* during *office hours* together with *colleagues*, his situation is classified as a *business situation*. His phone shows the inferred situation together with a corresponding list of filtered tasks (shown on the left part of Figure 1). To further specify his needs, Dawson may select one of the recommended tasks ('go to destination' in this case) and finally invoke an associated service (as shown on the right part of Figure 1).

Let us assume another situation taking place at the same location.

Situation 2: Private Meeting at Tokyo Station

Dawson Campbell arrives on a Saturday around noon at the Tokyo main station where Mark Buchanan, his father-in-law, is awaiting him. They plan to shop for a birthday present for Dawson's wife.

This situation is classified as *private family meeting*, because it takes place during *leisure hours* and only *relatives* are in the proximity. In this case, the situation-aware recommender application suggests tasks that are related to private activities such as 'go to movie theatre', 'go shopping', etc.

The key statement of these scenarios is that task-lists are actually tailored to different situations of the user, even if some context conditions are the same (location in this case). In this respect, our system facilitates users to access the mobile services that fit best to their current situation, purely based on qualitative context information.

5 Architecture

Figure 3 depicts the overall system architecture. The implementation contains two main parts, the situation engine and the task navigator.

The situation engine receives context information that has been collected by the task navigator on the mobile device. Furthermore, this information is enriched by context artifacts, such as environmental data, social relations between companions and a qualitative representation of time, all gathered from a distributed network of context providers. Thereupon, an axiomatized situation instance is constructed and sent to the inference engine. According to the world-knowledge encoded in the situation ontology, this instance is classified and the inferred situation is propagated back to the task navigator. A subcomponent of the task navigator, the task filter, detects the most appropriate task nodes within the task ontology by matching the derived situation with the task-specific categories. Finally, a representation of the resulting task-list is constructed by the task navigator and presented to the user on his mobile device for further navigation and service selections.

The task ontology stores descriptions for abstract as well as concrete tasks and their interrelations as semantic descriptions. Large and abstract tasks are thereby described by sequences of

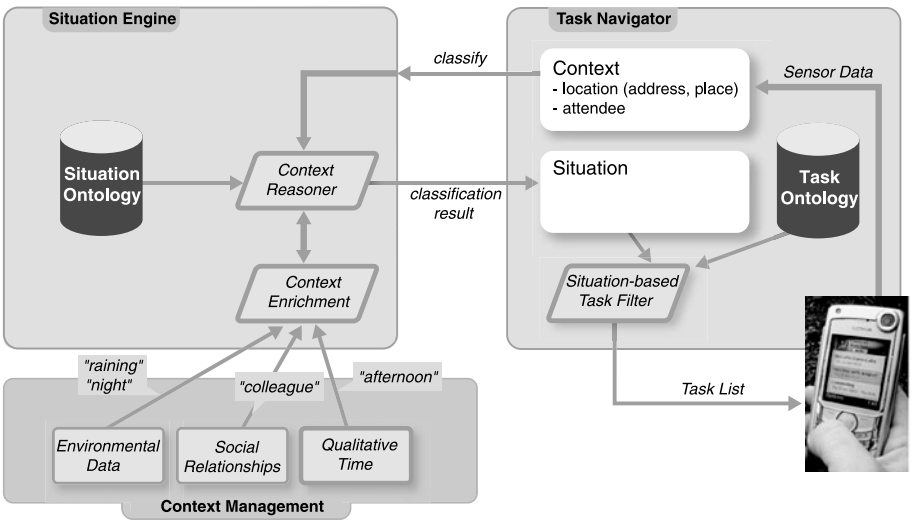


Figure 3 Architecture

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

smaller sub-tasks. In addition, abstract tasks are annotated with enabling context conditions and concrete tasks are linked to appropriate information services via Uniform Resource Identifiers. The task structures are defined in terms of the process model of the OWL-S ontology (Martin *et al.*, 2004). Each task node is represented as a service class and categorized according to the high-level context concepts such as *Business_meeting*, defined within the situation ontology. The context conditions describing the applicability of a task node are thereby encoded as corresponding OWL-S service profiles.

A Context Management Framework (CMF) (Yau *et al.*, 2002; Duran-Limon *et al.*, 2004; Floréen *et al.*, 2005; Kernchen *et al.*, 2006; Klemettinen, 2007) provides the basic structures for discovery and distribution of context information from heterogeneous sources in a standardized way (see Moran and Dourish (2001) for an introduction and overview). It is defined by a generic system architecture of interconnected components for the management of contextual data, which are connected dynamically at run time, and a Context Representation Framework (CRF) for standardizing the data representation (cf. Section 7) and exchange within this framework. Different types of CMF components can be identified. A *context provider* encapsulates a context source or otherwise aggregate contextual data to be stored and distributes among *context consumers* via different mechanisms (query or subscription) and in different transport formats (e.g. XML or CSV). Context providers need to be registered in a *context broker* by providing their advertisement to be discovered by context consumers. This advertisement describes uniquely the functionality of the provider, the types of context information it can provide and the relevant entities playing a role in this information. A provider might range from simply wrapping a body sensor for a single user to a full-fledged inference reasoner that combines information gathered from other context providers for multiple users. What binds them together is that they all implement a minimal set of common interfaces. Further management components such as an *identity manager* and a *privacy manager* provide basic functionality for authorization and the management of user-defined privacy directives, respectively.

6 Task-oriented service navigation

Task-oriented service navigation systems support users in finding appropriate service by offering a rich representation of activities linked to appropriate services that may be able to solve a user's task (Naganuma & Kurakake, 2005). A part of the task ontology underlying our system is shown in Figure 4. Abstract tasks are refined via a *is-achieved-by* relation to more concrete tasks which

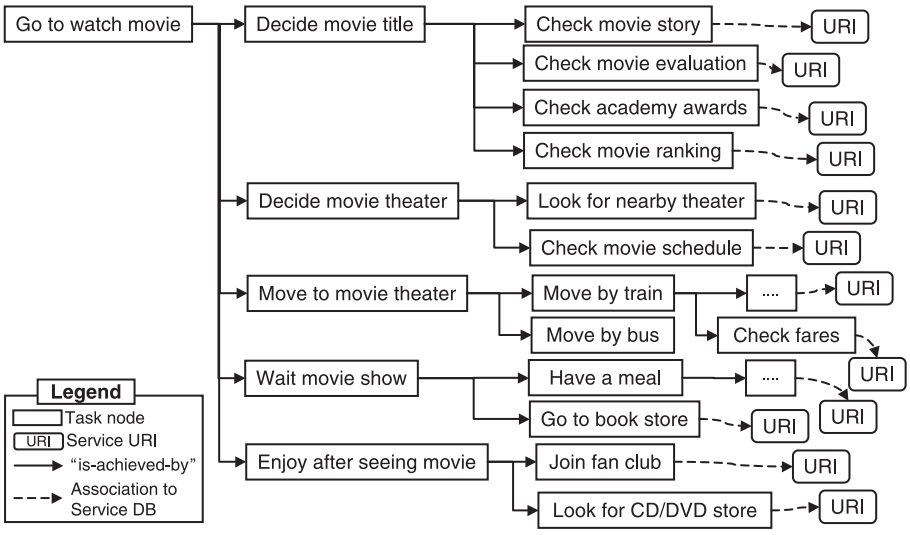


Figure 4 Task ontology fragment

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

are eventually associated with specific services or contents via Uniform Resource Identifiers (URIs). This task ontology allows users to navigate within the task space by repeatedly expanding tasks to sub-tasks thereby refining the problem specification until a leaf node is reached and an associated service can be activated. This process allows users with only a vague aim to find services that match their demands without knowing any concrete service name in advance. More details about the task ontology and its construction can be found elsewhere (Naganuma *et al.*, 2006).

We evaluated our task-based service navigation system by comparing its performance with the Japanese mobile search facilities provided by Yahoo[3] and Google[4] (Fukazawa *et al.*, 2006). In this study we assumed a situation in which a user just missed his train and had to wait for a few hours at the station and the 40 subjects were to find mobile sites that could provide recommendations on how to spend this time. It turned out that our service navigation system supported the subjects more effectively in this task than the pure search-based portals.

7 Context representation and classification

We adopted the IST MobiLife[5] Context Management Framework (Floréen *et al.*, 2005; Kernchen *et al.*, 2006; Klemettinen, 2007) to achieve interoperability between context sources from diverse domains by defining an XML-based context meta model. The elements of this meta model are linked to ontologies that define the basic contextual categories, used to represent qualitative aspects of context information.

We refer to an ontology as a logical theory accounting for the intended meaning of a formal vocabulary, that is, its ontological commitment to a particular conceptualization (cf. Section 3). Therefore, the decidability of the selected ontology language is crucial. The OWL DL fragment of the OWL fulfils this requirement, is highly expressive and has the potential to become the standard ontology language for the Semantic Web. Its selection as the ontology language of choice resulted in the construction of high-quality ontologies (i.e. ontologies that are verified to be consistent by fully automatic inference engines available for OWL DL). It is important to note that we do not propose the ontologies described hereafter as the main representation format for all aspects of context modelling, as ontologies are limited to the formulation of qualitative aspects and the available inference engines are generally weak in handing large amounts of data efficiently.

The context ontologies were developed in an incremental way guided by our application scenario. In the kick-off phase of the development we defined the initial baseline ontology in the $\mathcal{SHIF}(D)$ fragment of OWL DL, consisting of six components covering already most of the intended vocabulary. A first (naive) refinement of those ontologies resulted in a considerable extensions by a factor of five in size, adding detailed descriptions of the initial concepts, formulating additional vocabulary informed by the vCard standard[6] and the FOAF (Friend-Of-A-Friend) vocabulary[7], linking to standard ontologies like time-entry (Hobbs & Pan, 2004), a OWL DL variant of Time OWL as used in OWL-S.

The evaluation of these refined ontologies revealed deficits in reasoning performance, mainly caused by the size of the extensional knowledge formulated and due to the use of modelling constructs known to be computationally expensive (such as nominals in combination with inverse roles) pushing the underlying logic to $\mathcal{SHOIF}(D)$ (cf. Section 8). In a second refinement step we avoided expensive constructs not strictly necessary, restricted the logic to $\mathcal{SHIF}(D)$ and split all components in their extensional and intentional parts resulting in a total number of 12

[3] http://www.yahoo.co.jp
[4] http://www.google.co.jp/imode
[5] http://www.ist-mobilife.org
[6] http://www.imc.org/pdi/vcard-21.txt
[7] http://www.foaf-project.org

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

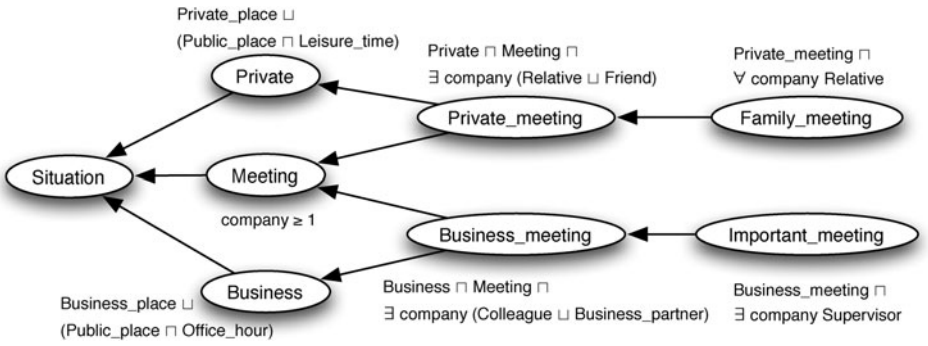


Figure 5 Situation ontology fragment

interrelated components defining more than 300 concepts, 200 properties and 300 individuals. We experienced a much improved reasoning performance although we added vocabulary covering most of IETF RPID[8] and Microsoft's Mappoint[9].

All component ontologies are integrated by a situation ontology that defines a top-level concept named *Situation* (cf. Figure 5). This concept is refined by concepts such as *Private* and *Business* by referring to concepts and relations defined in the component ontologies.

Our approach to realize high-level situational reasoning is to apply dynamic assertional classification of situation descriptions represented as concrete individuals. Each situation individual is assembled out of a set of entities representing qualitative context information such as the location (e.g. office), the time (e.g. afternoon) and the persons in proximity (e.g. friends). Finally, the direct subsuming concepts of the situation individual determine the user's abstract situation.

In the following, we sketch the OWL definitions of two typical situations using standard DL syntax (Baader *et al.*, 2003). A person's situation is classified as *Business*, if he is either located at a business place (such as an office) or at a public place (e.g. a train station) during office hours.

$$\begin{aligned}\mathrm{Business} := \mathrm{Situation} \sqcap (&\exists\, \mathrm{location} \,.\, \mathrm{Business_place} \sqcup \\ &(\exists\, \mathrm{location} \,.\, \mathrm{Public_place} \sqcap \exists\, \mathrm{time} \,.\, \mathrm{Office_hour}))\end{aligned}$$

A person is participating in a family meeting if he or she is in a private meeting situation where all participants are relatives.

$$\mathit{Family_meeting} := \mathit{Private_meeting} \sqcap (\forall\, \mathit{company} \,.\, \mathit{Relative})$$

Situational reasoning is realized using a DL reasoning engine that classifies concrete individual situations w.r.t. the ontology. Let us consider the Situation 1 introduced in Section 4. First, each piece of context information such as the location (Tokyo station), the time (Friday morning) and all companions (Dawson's boss Fiona and his project partner Gordon) are represented in terms of vocabulary formalized by the context ontologies. This requires the mapping of sensed quantitative data to qualitative representations (e.g. a timestamp is mapped to an individual in the Abox representing a Friday morning). The qualitative representations are enriched by the world-knowledge formalized in the component ontologies and are combined to an Abox individual in the situation ontology.

Computed by the reasoning engine, the direct concept type for the situation instance according to Situation 1 is *Important_meeting*. In this case, the location of the scene is a public place (as *tokyo_station* is an instance of the concept *Station*, which in turn is a sub-concept of *Public_place*) during office hours (as the individual *friday_morning* is classified as *Office_hours*) and the main

[8] http://www3.ietf.org/proceedings/06mar/IDs/draft-ietf-simple-rpid-10.txt
[9] http://support.microsoft.com/default.aspx?scid=kb;en-us;884771

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

actor Dawson is accompanied by his supervisor and a business partner. Similarly, the situation instance constructed for Situation 2 is classified as *Family_meeting* as it takes place at a public location during leisure time and only relatives are detected in the proximity of Dawson.

The situational reasoning process described above is supported by deductions in all component ontologies. For example, the agent ontology specifies in detail the semantics of social relations between people. Based on the knowledge encoded within the ontology, it can be inferred that two persons (like Dawson and Fiona) are colleagues, taking into account the transitivity of this relationship in case they have a common colleague. Similarly, even if no direct relation between Dawson and Mark is specified it can be inferred that Mark is Dawson's father-in-law (defined to be the father of the spouse of a person), because Dawson's wife Madeleine is known to be the child of Mark. In this case, the sub-property and inverse property specifications within the agent ontology enable this logical inference: *wife* is defined as a sub-property of *spouse* and *father* is the inverse of *child*.

8 Discussion

We integrated a situational reasoning engine into a real-world mobile service application. Our classification-based approach relies on ontology technology for the representation and reasoning on context information. As the scalable management of data is not a core feature of pure ontology-based context management and typical context models are usually rather large, we restricted its scope to high-level qualitative context elements. Lower-level context information is represented according to an XML-based meta model and managed separately. The arising reasoning problems are answered by a DL inference engine that provides complete reasoning support for the decidable fragment of OWL.

The use of the standard representation language OWL and the standardized reasoner interface DIG (Bechhofer *et al*., 2003) (a stateless HTTP-based protocol with XML syntax) enabled us to directly compare the influence of different context ontologies and reasoners on the overall system performance. We observed that the inference technology as implemented in modern DL reasoners made significant progress during the last years. Novel optimization techniques enabled a tremendous increase in performance, and also the coverage was greatly extended. By now most systems can be accessed via DIG, and support nominals as well as Abox reasoning directly. FaCT++ and Pellet support $\mathcal{SHOIQ}(\mathcal{D})$ (OWL DL extended by qualified cardinality restrictions) and Racer supports $\mathcal{SHIQ}$ including approximated nominals and reasoning with concrete domains.

Nevertheless we observed several limitations in the available technology (Liebig *et al*., 2005). The import mechanism of OWL, which brings all triples into the importing ontology, has a limited use for the sharing and reuse of ontologies. An appropriate mechanism on the syntactic as well as the semantic level is necessary for referencing entities in another ontology without inheriting all of its complexity. Furthermore, our modelling of context ontologies would benefit from additional constructs such as qualified cardinality restrictions and a richer object property structure that would allow the specification of reflexive, irreflexive, symmetric and anti-symmetric properties as well as property chains and disjoint property axioms. Reasoning support for the DL-safe fragment (Motik *et al*., 2005) of SWRL (Horrocks & Patel-Schneider, 2004) and for concrete domains on user-defined types would allow us to further enhance the quality of our situation engine. While concrete domain reasoning and support for SWRL is already available in some inference engines, and most of the requested additional language constructs are part of the OWL 1.1 draft[10] created by the ad-hoc OWL community, an improved import mechanism as given by the $\mathcal{E}$-connection mechanism (Grau *et al*., 2006) and implemented in Pellet is not included.

At first, we experimented with the DIG interface to realize the communication between our application and the inference engine. However, DIG 1.1 does not support the removal of specific

[10] `http://owl11.cs.manchester.ac.uk`

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

axioms making it necessary to re-submit the complete ontology for each request to our situation engine. This is especially awkward for our application where only a very small part of the assertional knowledge changes between two requests (namely, the part defining the current situation to be classified). As active members of the informal DIG 2.0 working group[11] we therefore propose a modular extension to the interface that supports incremental reasoning and retraction (Turhan *et al.*, 2006). Unfortunately, current reasoners typically only provide some kind of batch-oriented reasoning procedure. A notable exception is Racer which offers low-level retraction support for most of its statements. Still, because of the lack of algorithms for appropriately handling incremental additions as well as retractions, Racer initiates a complete reclassification after each change in the ontology. Initial empirical results, performed with an experimental version of Pellet, indicate that incremental classification algorithms for $\mathcal{SHOIN}(\mathcal{D})$ can be quite effective (Parsia *et al.*, 2006).

The ability to handle simultaneous requests is one of the key requirements in our dynamic mobile setting. However, current inference engines do not implement any transaction management. Only for Racer, support for dispatching, load balancing and caching of OWL-QL (Fikes *et al.*, 2004) queries is available via the RacerManager (Galinski *et al.*, 2005). As OWL-QL does not support modifications of an ontology, we had to implement our own transaction management system that enables the sharing of reasoning resources between requests, but avoids the necessity to maintain a separate knowledge base for each user.

It has been observed before (Wang *et al.*, 2004; Lee *et al.*, 2005) that the delay caused by ontology-based inferencing easily becomes a major obstacle for realistic applications. This is especially problematic for ontologies that constantly change, because well-established optimization techniques such as tabling (used in various rule-based inference engine) cannot be applied. As a consequence of the high worst-case complexity of expressive DLs, such as $\mathcal{SHOIN}(\mathcal{D})$ underlying OWL DL, modern DL reasoners implement a suite of optimization techniques to achieve acceptable performance. The efficiency of implementations on concrete cases depends therefore on the applicability of optimizations, which varies with the language features in use. For example, the use of domain and range restrictions can lead to cycles in a Tbox for which termination of the tableaux algorithm can only be ensured by blocking. However, known blocking strategies for $\mathcal{SHOIN}$ are less effective if inverse roles are involved. On the other hand, if nominals do not occur in an ontology blocking can be realized more efficiently (Haarslev *et al.*, 2005). Therefore we avoid the use of standard ontologies, such as the $\mathcal{SHOIF}(D)$ entry sub-ontology of time (Pan & Hobbs, 2004). It has to be seen how the recently suggested techniques for optimizing DL reasoning in the presence of nominals (Sirin *et al.*, 2006a) perform in practice.

We optimized our initial ontologies by removing nominals and most of the domain and range restrictions. Furthermore, we reduced the number of loaded axioms and objects (especially Abox individuals) and axioms by splitting the ontology in small components and by separating ontologies in A- and Tboxes to cope with the limits of the OWL import statement. This step resulted in a performance gain of up to 1.5 seconds per request. Furthermore, we compared different retraction strategies using Racer. The simplest form of retraction is reloading of ontologies and can be accelerated by either loading from a pre-classified image or by cloning an ontology in memory. For small Aboxes cloning outperformed true retraction realized with forget statements. However, the strategy performed best was to keep situation individuals up to a certain number (about 20 in our case) in the Abox before cloning a fresh pre-loaded Abox. Of course, keeping individuals and axioms in the Abox is only possible if they do not influence later classifications.

The time to compute our comparable simple reasoning problems is dominated by the communication overhead caused by the reasoner interface. Accessing Racer via its native API using TCP is about 1.5 times faster than the access via HTTP/DIG and even 2 times faster than the access realized with the triple-oriented framework Jena2 (Carrol *et al.*, 2004). Naturally, we achieved

[11] http://dl.kr.org/dig/interface.html

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

the best performance by using the Pellet reasoner running in the same Java virtual machine and this way completely avoiding any external communication.

Because existing performance results of DL reasoners are often limited to static Tbox classification (Weithöner *et al*., 2006), we plan to perform a detailed analysis of the influence of different retraction strategies for dynamic assertional reasoning, to compare the performance of interfaces and to test the effect of the ontology size and complexity on realistic reasoning tasks. By that we hope to gain inside on how to further optimize our situation engine.

Our current prototype has only a limited support for automatic context acquisition. We plan to advance the prototype towards the use of more actual context information from the real world. Planned extensions will combine GPS-based location information with the RFID-based context tags we use currently for location tracking, as well as short distance wireless communication technologies such as Bluetooth to detect people in proximity (Luther *et al*., 2005a; Böhm *et al*., 2006).

Acknowledgements

The authors would like to thank Sebastian Böhm, Kunihiro Fujii, Thorsten Liebig, Takefumi Naganuma, Bertrand Souville and Timo Weithöner for their valuable contributions to this work.

References

Baader, F., Calvanese, D., McGuinness, D., Nardi, D. and Patel-Schneider, P. 2003 *The Description Logic Handbook—Theory, Implementation and Applications*. Cambridge: Cambridge University Press.

Bechhofer, S., Möller, R. and Crowther, P. 2003 The DIG description logic interface. In Calvanese, D., De Giacomo, G. and Franconi, E. (eds.), *Proceedings of the International Workshop on Description Logics (DL'03), Rome, Italy*, vol. **81** of *CEUR Workshop Proceedings*. CEUR-WS.org, pp. 152–159.

Böhm, S., Luther, M., Koolwaaij, J. and Wagner, M. 2006 ContextWatcher—connecting to places, people, and the world. In *Proceedings of the Poster and Demo Track of the 5th International SemanticWeb Conference (ISWC'06), Athens, GA, USA*.

de Bruin, J., Polleres, A., Lara, R. and Fensel, D. 2005 *OWL—WSML Working Draft 05–15–2005 D20.1v0.2, DERI*.

Carrol, J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A. and Wilkinson, K. 2004 *Jena: implementing the Semantic Web recommendations*. In Feldman, *et al*. 2004, pp. 74–83.

Chen, H., Finin, T. and Joshi, A. 2005. The SOUPA ontology for pervasive computing. In Tamma, V., Cranefield, S., Finin, T. and Willmott, S. (eds.), *Ontologies for Agents: Theory and Experiences*, Berlin: Springer Verlag. pp. 233–258.

Duran-Limon, H. A., Blair, G. S., Friday, A., Grace, P., Samartizidis, G., Sivaharan, T. and Wu, M. 2004 *Context-Aware Middleware for Pervasive and Ad Hoc Environments*. Technical Report, Lancaster University.

Feldman, S. I., Uretsky, M., Najork, M. and Wills, C. E. (eds.) 2004 *Proceedings of the 13th International Conference on the World Wide Web (WWW'04)*, Manhattan, NY, USA: ACM Press.

Fikes, R., Hayes, P. and Horrocks, I. 2004. OWL-QL: A language for deductive query answering on the Semantic Web, *Journal of Web Semantics: Science, Services and Agents on the World Wide Web* **2**(1), 19–29.

Floréen, P., Przybilski, M., Nurmi, P. 2005 Towards a context management framework for MobiLife. In *Proceedings of the IST Mobile & Communications Summit*, Dresden, Germany.

Fukazawa, Y., Naganuma, T., Fujii, K., and Kurakake, S. 2006 Service navigation system enhanced by place- and task-based categorization. In *Proceedings of the 4th International Conference on Mobile Systems, Applications and Services (MobiSys'06)*, Upsala, Sweden.

Galinski, J., Kaya, A. and Möller, R. 2005 *Development of a server to support the formal Semantic Web query language OWL-QL*. In Horrocks *et al*. 2005.

Gil, Y., Motta, E., Benjamins, R. and Musen, M. (eds.) 2005 *Proceedings of the 4th International Semantic Web Conference (ISWC'05), Galway, Ireland*, vol. **3729** of *Lecture Notes in Computer Science*, Berlin: Springer Verlag.

Grau, B., Parsia, B. and Sirin, E. 2006 Combining OWL ontologies using E-connections, *Journal of Web Semantics: Science, Services and Agents on the World Wide Web* **4**(1), 40–59.

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

Grosof, B., Horrocks, I., Volz, R. and Decker, S. 2003 Description Logic programs: combining logic programs with Description Logic. In *Proceedings of the 12$_{th}$ International World Wide Web Conference (WWW'03)*, Budapest, Hungary: ACM Press.

Haarslev, V. and Möller, R. 2003 Racer: a core inference engine for the Semantic Web Ontology Language (OWL). In Sure, Y. and Corchó, O. (eds.), *Proceedings of the 2nd International Workshop on Evaluation of Ontology-Based Tools (EON'03), Sanibel Island, Florida, USA*, vol. **87** of *CEUR Workshop Proceedings*, CEUR-WS.org. pp. 27–36.

Haarslev, V., Möller, R. and Wessel, M. 2005 *Description Logic inference technology: lessions learned in the trenches*. In Horrocks, *et al*. 2005, pp. 160–167.

Hobbs, J. and Pan, F. 2004 An ontology of time for the Semantic Web, *ACM Transactions on Asian Language Information Processing* **3**(1), 66–85.

Horrocks, I. and Patel-Schneider, P. 2004 *A proposal for an OWL rules language*. In Feldman, *et al*. 2004, pp. 723–731.

Horrocks, I., Sattler, U. & Wolter, F. (eds.). 2005 *International Workshop on Description Logics (DL'05), Edinburgh, Scotland, UK*, vol. **147** of *CEUR Workshop Proceedings*, CEUR-WS.org.

Kernchen, R., Bonnefoy, D., Battestini, A., Mrohs, B., Wagner, M. and Klemettinen, M. (2006) Context-awareness in MobiLife. In *Proceedings of the 15th IST Mobile & Wireless Communication Summit*. Mykonos, Greece.

Khushraj, D. and Lassila, O. 2004 CALI: context awareness via logical inference. In *Proceedings of the ISWC'04 Workshop on Semantic Web Technology for Mobile and Ubiquitous Applications (SWMU'04)*, Hiroshima, Japan.

Klemettinen, M. (ed.). 2007 *Enabling Technologies for Mobile Services: The MobiLife Book*. England: John Wiley & Sons Ltd.

Lee, J., Park, I., Lee, D. and Hyun, S. (2005) Application-oriented context pre-fetch method for improving inference performance in ontology-based context management. In Lee, J. (ed.) *Proceedings of the 1st International Workshop on Contexts and Ontologies: Theory, Practice and Applications at AAAI'05,* Pittsburgh, Pennsylvania, USA. pp. 41–48.

Liebig, T., Luther, M., Noppens, O., Paolucci, M., Wagner, M. and von Henke, F. 2005 Building applications and tools for OWL. In Grau, B. C., Horrocks, I., Parsia, B. and Patel-Schneider, P. (eds.), *Proceedings of the 1st OWL Experiences and Directions Workshop (OWLED'05), Galway, Ireland*, vol. **188** of *CEUR Workshop Proceedings*,CEUR-WS.org. pp. 190–201.

Luther, M., Böhm, S., Wagner, M. and Koolwaaij, J. (2005a) *Enhanced presence tracking for mobile applications*. In Gil, *et al*. 2005, pp. 105–108.

Luther, M., Mrohs, B., Wagner, M., Steglich, S. and Kellerer, W. (2005b) Situational reasoning—a practical OWL use case. In *Proceedings of the 7th International Symposium on Autonomous Decentralized Systems (ISADS'05)*, Chengdu, China. pp. 96–103.

Martin, D., Burstein, M., Hobbs, J. J. 2004 OWL-S: Semantic Markup for Web Services. W3C Member Submission, The OWL Services Coalition.

McGuinness, D. and van Harmelen, F. 2004 OWL Web Ontology Language overview. W3C Recommendation REC-owl-features-20040210, World Wide Web Consortium.

Moran, T.P. and Dourish, P. 2001 Introduction to special issue on context-aware computing, *Human-Computer Interaction (HCI)* **16**(2–3), 87–96.

Motik, B., Sattler, U. and Struder, R. 2005 Query answering for OWL-DL with rules, *Journal of Web Semantics: Science, Services and Agents on the World Wide Web* **3**(1), 41–60.

Mrohs, B., Luther, M., Vaidya, R., Wagner, M., Steglich, S., Kellerer, W. and Arbanowski, S. 2005 OWL-SF—a distributed semantic service framework. In *Proceedings of the 1st Workshop on Context Awareness for Proactive Systems (CAPS'05),* Helsinki, Finland: Helsinki University Press. pp. 67–78.

Naganuma, T. and Kurakake, S. 2005 Task knowledge based retrieval for service relevant to mobile user activity. In Gil, *et al*. 2005, pp. 959–973.

Naganuma, T., Luther, M., Wagner, M., Tomioka, A., Fujii, K., Fukazawa, Y. and Kuakake, S. 2006 Task-oriented mobile service recommendation enhanced by a situational reasoning engine. In Mizoguchi, R., Shi, Z. and Giunchiglia, F. (eds.), *Proceedings of the 1st Asian Semantic Web Conference (ASWC'06), Beijing, China*, vol. **4185** of *Lecture Notes in Computer Science*, Berlin: Springer Verlag. pp. 768–774.

Pan, F. and Hobbs, J. 2004 Time in OWL-S. In *Proceedings of the AAAI Spring Symposium on Semantic Web Services*, California, USA: Stanford University. pp. 29–36.

Parsia, B., Halaschek-Wiener, C. and Sirin, E. 2006 Towards incremental reasoning through updates in OWL-DL. In *WWW'06 Workshop on Reasoning on the Web (RoW'06)*, Edinburgh, UK.

Sirin, E., Grau, B. C. and Parsia, B. (2006) From wine to water: optimizing Description Logic reasoning for nominals. In Doherty, P., Mylopoulos, J. and Welty, C. A. (eds.), *Proceedings of the 10th International Conference on the Principles of Knowledge Representation and Reasoning (KR'06)*, Lake District, UK: AAAI Press. pp. 90–99.

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press

Sirin, E., Parsia, B., Grau, B. C. and Kalyanpur, A. 2007 Pellet: a practical OWL-DL reasoner, *Journal of Web Semantics*, **5**(1), pp. 51–53.

Tsarkov, D. and Horrocks, I. 2005 Ordering heuristics for Description Logic reasoning. In Kaelbling, L.P. and Saffiotti, A. (eds.), *Proceedings of the 19th International Conference on Artificial Intelligence (IJCAI'05)*, Edinburgh, Scotland, UK: Professional Book Center. pp. 609–614.

Turhan, A.-Y., Bechhofer, S., Kaplunova, A., Liebig, T., Luther, M., Möller, R., Noppens, O., Patel-Schneider, P., Suntisrivaraporn, B., Weithner, T. 2006 DIG2.0—towards a flexible interface for Description Logic reasoners. In Grau, B. C., Horrocks, I., Parsia, B. and Patel-Schneider, P. (eds.), *Proceedings of the OWL Experiences and Directions Workshop (OWLED'06) at the ISWC'06, Athens, Georgia, USA*, vol. **216** of *CEUR Workshop Proceedings*, CEUR-WS.org.

Wang, X., Zhang, Q., Gu, T. and Pung, H. K. 2004 Ontology-based context modeling and reasoning using OWL. In *Proceedings of the PerCom Workshop on Context Modeling and Reasoning (CoMoRea'04)*, Orlando, FL, USA. pp. 18–22.

Weithöner, T., Liebig, T., Luther, M., and Böhm, S. 2006 What's wrong with OWL benchmarks? In Wache, H., Stuckenschmidt, H., Parsia, B., Guo, Y., Finin, T. and Beckett, D. (eds.), *Proceedings of the 2nd International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS'06)*, Athens, GA, USA. pp. 101–114.

Yau, S. S., Karim, F., Wang, Y., Wang, B., and Gupta, S. K., 2002 Reconfigurable context-sensitive middleware for pervasive computing, *Pervasive Computing* **1**(3), 33–40.

https://doi.org/10.1017/S0269888907001300 Published online by Cambridge University Press