AmbieAgents: A Scalable Infrastructure for Mobile and Context-Aware Information Services

Till C. Lech
CognIT a.s
Meltzersgt. 4
N-0257 Oslo, Norway
+47 22540520
Till.Lech@cognit.no

Leendert W. M. Wienhofen
CognIT a.s
Meltzersgt. 4
N-0257 Oslo, Norway
+47 22540520
Leendert.Wienhofen@cognit.no

ABSTRACT
Context-aware information systems for mobile users have to cope with a variety of requirements in order to be able to provide an added value that goes beyond simply location-based services. The EU-IST AmbieSense project has developed a reference architecture and implemented demonstrators that meet a number of these requirements. In this paper we present AmbieAgents, an agent-based infrastructure for context-based information delivery for mobile users. We describe the overall AmbieSense system architecture and particularly the agent-system and its main features. As the AmbieSense system was demonstrated in a real world scenario at the Oslo international airport, we also give an account of our experiences from user tests executed during the demonstration.

Categories and Subject Descriptors
H.3.4 [**Information Storage and Retrieval**]: Systems and Software – *Distributed systems*

General Terms
Design, Experimentation

Keywords
Multi-Agent Systems, Pervasive Computing

1. INTRODUCTION
Ambient information systems have to cope with a variety of challenges in order to fit the needs of mobile users. Different connections to content sources with different bandwidth have to be considered; content from the different sources must be gathered and presented to the user in a coherent way. Furthermore, the system must allow for the easy integration of new content sources in order to be attractive for new content service providers.

In order to meet these requirements, the EU AmbieSense project (IST 2001-34244, http://www.ambiesense.net) has developed a reference architecture for ambient, personalized and context-aware distribution of information to mobile users. The cornerstones of the AmbieSense reference architecture include wireless context tags, content provision platforms and mobile devices. These three platforms require an infrastructure for the communication of user contexts and content retrieval that provides the necessary flexibility, robustness and scalability. In order to meet these challenges, an agent-based approach was chosen when developing the infrastructure. Intelligent Agents and Multi-Agent Systems have become a predominant computing paradigm in the field of pervasive computing. Among other reasons, this can be justified this by the fact that agents can reduce the complexity of this application domain by delegation of tasks. Furthermore, agents are especially well-suited for distributed environments.

There are numerous related approaches to context-aware information services that utilise agent-based technology. The MyCampus project at CMU [9] has developed an environment that is related to the AmbieSense domain, however, in AmbieSense, a wider context model is being used in order to cover a broad range of possible applications. More recently, the BerlinTainment [13] project has demonstrated a framework for the provision of activity recommendations based on mobile agents, covering a similar information domain as AmbieSense.

In this paper, we describe the features and architecture of AmbieAgents, the AmbieSense Multi-Agent System. AmbieAgents, which is based on JADE/LEAP serves as infrastructure between the AmbieSense corner stones. AmbieAgents can run in a variety of configurations in distributed environments according to the hardware and functional requirements of the service application employing the system. The AmbieSense architecture has been implemented, tested and demonstrated in a real world setting at the Oslo international airport. After a brief overview of the overall system architecture of the AmbieSense system, we give a detailed account of AmbieAgents and its use by example of the AmbieSense airport information application.

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands

2. THE AMBIESENSE SYSTEM

2.1 Overall Architecture of the AmbieSense System

The goal of the AmbieSense project is to develop a platform that satisfies the information needs of travelers and mobile users by providing ambient and personalised information in a context-aware manner. In the course of the project a reference architecture for context-aware information services has been designed. The corner stones in the AmbieSense reference architecture comprise the following main parts:

- Wireless Context Tags
- Mobile Users
- Content Service Providers (CSP)

Context tags are miniaturized, wireless computers placed at strategic points in the user's environment that can relay content from the CSP to the mobile device. Context Tags hold information about their environment and can be accessed by the mobile devices via Bluetooth in order to pinpoint the user's location and to enrich user contexts with ambient information such as location, available services and uplinks etc. Together with agents, which process the enriched contexts, Context Tags create an infrastructure of ambient intelligence beyond simply location-based services. Rich user contexts can be used in order to retrieve relevant and personalised content that is provided either directly from the content provider (e.g. via WLAN or GPRS) or downloaded from the context tag.

A detailed account of the AmbieSense Reference architecture is provided in [8]; in this paper, we focus on AmbieAgents, the infrastructure for communication of context and distribution of content in AmbieSense.

2.2 User Context in AmbieSense

Before examining the AmbieAgents infrastructure, it is worthwhile to take a closer look at the concept of context. Context-awareness has become a more and more important issue in pervasive and ubiquitous computing. Capturing and utilising user contexts is a major challenge in order to provide value-added services that go beyond simply location-based approaches. The AmbieSense Reference Information Model [AmbieSense 2003] describes context as follows:

"A context describes aspects of a situation seen from a particular actor's point of view. (…) In this way context is actually defined as something separate from the situation it self. A context in AmbieSense is a representation inside the computer. It represents aspects of a situation in the real world. (…) AmbieSense chose to implement this structure by developing Java-classes integrated with XML technology."

AmbieSense employs a user context model introduced by [7]. Here, a user context consists of five main parts:

- Social Context
- Task Context
- Personal Context
- Environmental Context
- Spatio-Temporal Context

These five sub-contexts constitute the main classes of the AmbieSense ontology that is used by AmbieAgents in order to communicate. Inside these five classes, application-specific

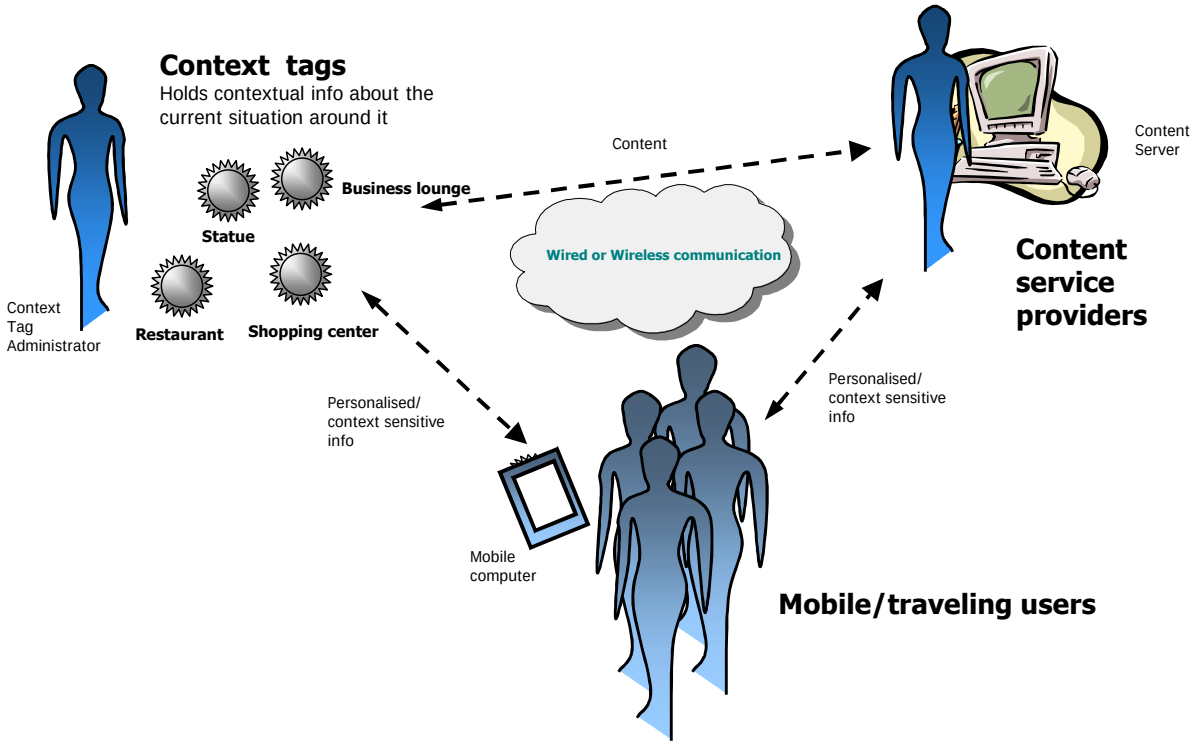

Figure 1. The AmbieSense overall reference architecture (from [8])

concepts and attributes can be added to the overall AmbieSense ontology. In order to enable persistent context storage, AmbieAgents can utilise a means for context management, such as the AmbieSense Context Middleware [6]. How context can be used by AmbieAgents in order to serve an information service will be described by example of the Oslo Airport application in Chapter 4.

3. AMBIEAGENTS - ARCHITECTURE

The AmbieAgents are built upon the JADE (Java Agent DEvelopment) Framework [2]. JADE supports the implementation of multi-agent systems through a middleware that complies with the FIPA specifications.

The agent platform can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required. The synergy between the JADE platform and the LEAP libraries allows to obtain a FIPA-compliant agent platform with reduced footprint and compatibility with mobile Java environments, as necessary in the AmbieAgent and AmbieSense context.

The AmbieAgents system was designed to be scalable in terms of hardware requirements as well as the complexity of the applications. It has been tested with agents in different settings, showing the flexibility of the design (depicted in Figure 3). Common for all configurations is the Context Agent on the mobile device, distributing the user context to the retrieval agents. The agents responsible for the content retrieval can be run on the mobile device, too, or they can be hosted by the content service provider. It is possible to run all agents and even the content (for example, in case there are downloadable information packages) on the mobile device (given that the implemented agents are not all too knowledge-intensive, due to limited processing capacity on the mobile device). Or, in case the content agent is very knowledge-intensive, it can also be run on a server.

3.1 The Context Agent

Context in AmbieSense is represented as a structured set of attribute-value pairs that capture relevant aspects of a user situation, where the term relevance depends on the service application domain (to a map service, for example, the user location will be more important than to a news service).

The Context Agent is the heart of the AmbieAgents framework and the most autonomous agent in the system. All agent communication is channeled through the Context Agent, as it is the one agent that administers the access to the user's context space. Whenever the current user context (attributes in the Social Context, Task Context, Personal Context, Environmental Context or Spatio-Temporal Context) changes, the Context Agent prepares the context by using the Context Ontology in order to transform the user context into an ACL message. The ACL message is then sent to the available Content Agents, Recommender Agent or other relevant agents.

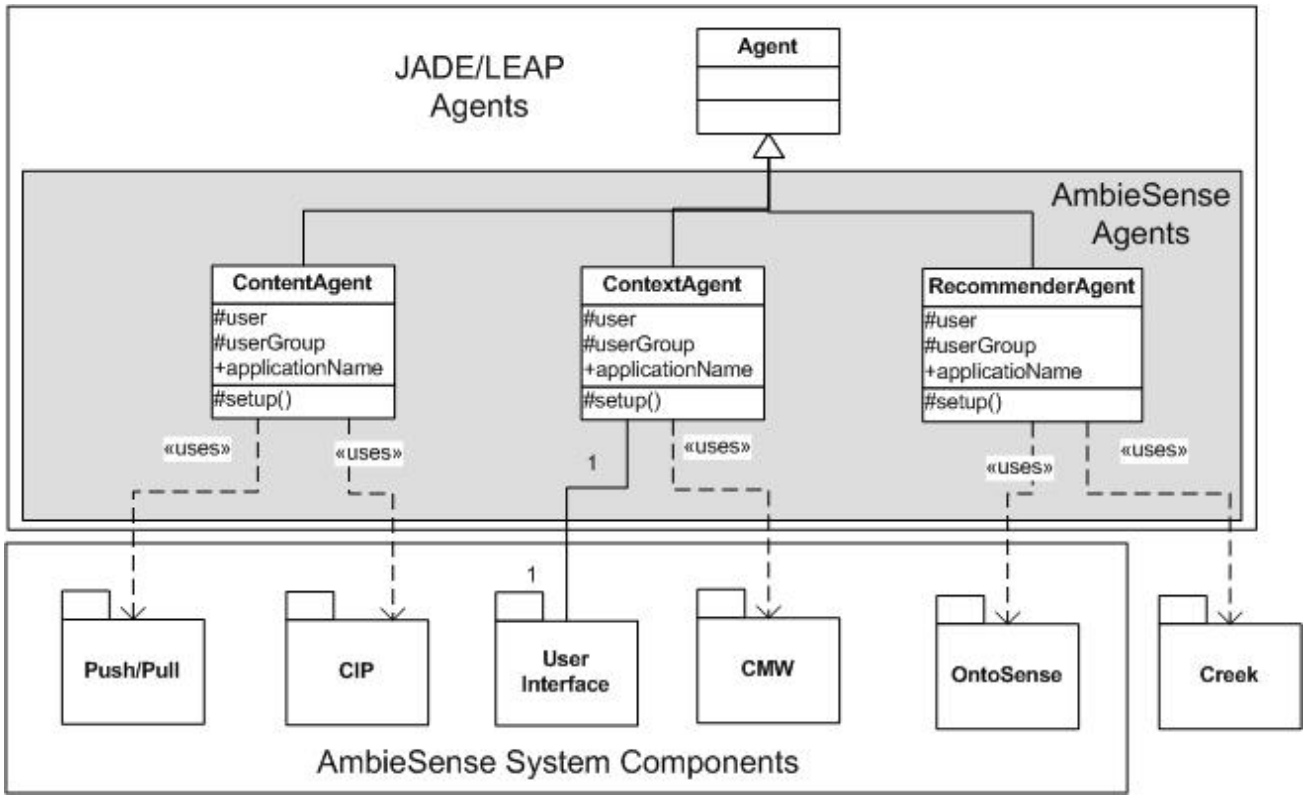

Figure 2. AmbieAgents – UML representation

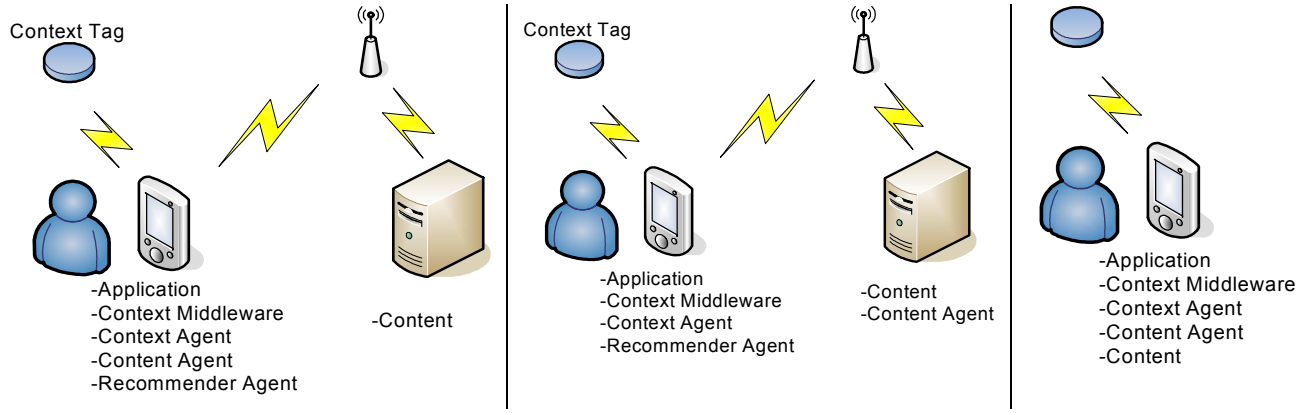

Figure 3. AmbieAgents Configurations

The user contexts used by the content agents are anonymous. The context middleware on the mobile device is the only component where the user's context history and his personal data are linked together. This is the reason why the user's personal Context Agent is the only entity that has access to the context space, thus controlling the information flow to and from the user – even though this may seem a bottleneck in the system.

When relevant content is found by one of the agents, the results (the resources' URLs) are returned to the Context Agent, which accumulates them and passes them on to the overlaying application.

3.2 Content Agents

The Content Agent enables the delivery of content to mobile users in the AmbieAgents system, linked to the user's context in a meaningful way. As described above, the Context Agent sends the user's context as an ACL message whenever the context changes and there's a need for content. The Content Agent uses information retrieval applications, such as CORPORUM OntoExtract technology for indexing content developed by CognIT a.s. The CORPORUM toolbox consists of a set of NLP mechanisms that enable high-precision content classification and retrieval based on a linguistic analysis and neuro-fuzzy methods, as described in [3]. A major part of the functionality in OntoExtract was developed within the OnToKnowledge project (IST1999-10132) [4]. The Content Agent uses this technology to retrieve relevant information, based on the users' preferences, from a pre-indexed source such as the shopping area web site of OSL. A detailed account of how the Content Agent uses OntoExtract is given by example of the OSL Gardermoen case in Section 4.

3.3 Recommender Agent

The Recommender Agent employs CREEK [1], a case-based reasoning (CBR) engine in order to determine the user's situation and triggers more specific content queries to the Content Agent. A detailed account of the use of CBR in AmbieSense is provided in [5]. Recommender agents in the AmbieAgents system can employ different techniques to recommend relevant information and can even use proprietary systems as long as they provide a JAVA compatible API.

3.4 Ontology

To allow for any reasoning within the AmbieAgents framework a common agreement on the semiotics used are required. The AmbieSense ontology represents the OSL Gardermoen ontology used by the Recommender Agent.

The FIPA provides ontologies for interoperability of applications. However, no useful ontology for the OSL Gardermoen case, representing among others user context, could be found. Therefore, an ontology was created that reflects the AmbieSense user context model. The general knowledge and contextual model in AmbieSense is encapsulated in OntoSense, the ontology for the AmbieSense domain. OntoSense reflects the general knowledge and contextual model in AmbieSense.

The ontology is structured as a tree, with three layers, where a common frame of reference constitutes the root, the AmbieSense specific concepts are defined in the middle layer, and the upper part, is specific for each of the specific sub-domains. For example, each of the restaurants at Oslo Airport has the possibility to build their own ontology concerning their actual offers. This ontology is then hooked up to the OntoSense ontology, which primarily defines the contextual concepts.

4. EXAMPLE APPLICATION: INFORMATION SERVICES AT THE OSLO AIRPORT

The implemented AmbieSense system was tested at the Oslo International airport (OSL) at Gardermoen in August 2004. The purpose of the demonstrator was to show how user context can be utilized in order to offer value-added information services to travelers at an international airport. As mentioned in Section 2.2, it is the application that defines the context model – within the framework of the overall AmbieSense context model. The owners of the application – in this case the responsible staff at OSL, as they are the domain experts – define what a user context should consist of in order to be able to distribute relevant context in an optimized way. The user context for the OSL information service application focused on the following main parts:

- Personal context: The user's shopping and dining preferences that are stored on the user's device

- Spatio-temporal context: The user's location in the building, typically provided by the Context Tag in the vicinity of the user.
- Environmental context: Status of the user's flight
- Task context: What the user is about to do. Either provided explicitly by the user or inferred from the user's location.

The test was executed on PDAs (Compaq iPaq, equipped with Bluetooth and WLAN connectivity), running Familiar Linux (The Familiar Project, http://familiar.handhelds.org/) for handheld devices. Since the test was executed by actual travelers found at the airport, no contextual history of their preferences could be used. Therefore, the users had the possibility to configure their own profile of dining and shopping preferences (see Figure 4, left).

Figure 4. User profiling in the OSL application

In addition, users had the opportunity to enter their flight number (see Figure 4, right). The OSL application was connected to the OSL flight database that updated the status of the flight as well as information on check-in islands and gate number every two minutes. Information from the flight database, along with the user preferences and location is included into the user context model.

At the airport, both the location and what a traveler is about to do is highly relevant for the distribution of information. Tax-free offers have no value if you are not traveling abroad since you will not be allowed to buy these items. The current location of a user plays a very important role, as there are two 'points-of-no-return' at the OSL airport; the security check and the document control that has to be passed when entering the area for international departures. Once one of these points is passed, the traveler is not allowed back to the previous area, meaning that all services and facilities before that point are now irrelevant as the user is not able to access these areas.

In the OSL demonstrator the Context Agent gets notified about context changes through the context middleware. The updated context representation is then passed as an ACL Message to the content agent, using the OSL context ontology as common vocabulary. The content agent matches the updated context against the content representation in order to retrieve content items relevant to the user (see Figure 5).

The content used for the OSL demo consists of a set of over 200 HTML-pages, partially taken from the OSL web site (http://www.osl.no). These pages were optimized for displaying them on a PDA. The content pages cover a range from general information on check-in facilities, security etc. to information on shops, restaurants as well as special offers or information on destinations. In order to make the content accessible in a context-

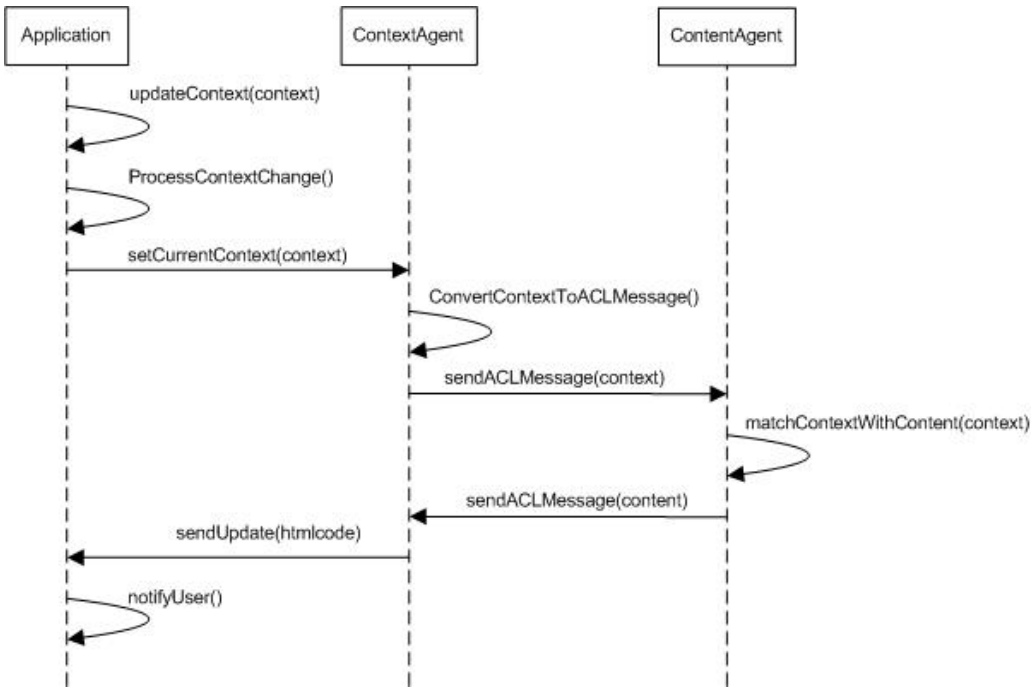

Figure 5. Data flow sketch of the OSL application

aware manner, meta-information for each html-page had to be extracted into an XML index file.

The extracted information consists of:

- The category of the content: Shopping, Dining, Service or Destination Info
- The title of the Page
- The filename of the page
- The location of the described venue (or the location of where the content should be made accessible to the user)
- Keywords describing the content of the page.

Obviously, some of the meta-information (such as title and filename) could be extracted automatically; however the location had to be determined manually, since it referred to the position of the mounted Context Tags in the demo. For finding the keywords of each page, the OntoExtract tool for natural language analysis was used. Based on the linguistic analysis of the description of e.g. a electronics retailer, an interest model containing the key concepts of the resource were extracted. Utilizing the context representation, the Content Agent searches the index file for resources that match the user context. In addition, information about the location of the user can be used in order to provide task-specific content. When entering the airport building, information about the correct check-in island was given (based on flight number), when approaching the security check, information on what not to bring on a plane was presented to the user.

4.1 Results from User Interviews

During the OSL test, user interviews were conducted with travelers while and after using the system. As there only were 3 days for user tests and due to the time each test and interview takes (25-40 minutes), the results are of little or no statistic significance (about 50 interview transcriptions), however, some tendencies become obvious. In the following paragraphs, we focus on user comments having to do with the agent-based content delivery. Comments about speed, reliability of the WLAN or hardware have been analyzed in the AmbieSense Test and Evaluation Report.

In general, users seem to favor the idea of getting recommendations about relevant content based on their profile or their preferences compared to exploring the available content in a browsing manner. Basically, the users were satisfied with the quality and relevance of the content items presented by the agents, despite they did not really see the need for e.g. shopping information on a small scale airport like OSL. However, on bigger airports, they would find such a system useful. Most of the users would not see any problem to give some personal information about their preferences to the system, as long as privacy and security mechanisms prevent their data from being misused. The majority of the user interviews suggest that the combination of agent technology and user contexts is a viable approach to delivering information to mobile user in an airport setting. Especially, when using several content agents on different content services, the demonstrated technology will be able to serve the information needs of travelers.

5. CONCLUSION AND FURTHER WORK

We described the AmbieSense system that is the implementation of a reference architecture for context-aware information services for mobile users, consisting of wireless context tags, content provision platforms and mobile users. The infrastructure inbetween the architectural corner stones is a JADE/LEAP-bases multi agent system, AmbieAgents. The complete AmbieSense system was tested and evaluated in user tests at the Oslo international airport. In general, the majority of the user interviews conducted at OSL suggest that the combination of agent technology and user contexts is a viable approach to delivering information to mobile user in an airport setting.

Especially, when using several content agents on different content services, the demonstrated technology will be able to serve the information needs of travelers. As from the technical point of view, we found that JADE/LEAP was a reliable and stable framework for this type of application, confirming the similar experiences expressed by e.g. Pavel Vrbal [10] in his survey of agent development frameworks. Since the design of AmbieAgents is flexible, it can be used in many different scenarios, for example in a knowledge management/sharing scenario in the process industry as depicted in [11]. A follow-up project, Contigo, building on the results of AmbieAgents has been launched, financed by the Deutsche Telekom AG (T-Systems). This project aims at finding interesting persons in addition to information resources, thus transferring the popularity of online-communities into a mobile setting. The basic ideas for the Contigo project are documented in [12]. The first phase of the project is on its way, with a planned demonstrator at the CeBIT exhibition in 2006.

6. ACKNOWLEDGEMENTS

The research described in this paper was part of the EU-funded IST project AmbieSense. Contributions to the design of the AmbieSense Multi-Agent System were made by several partners of the AmbieSense consortium; especially Hans Inge Myrhaug and Marius Mikalsen, at SINTEF, Norway, as well as Anders Kofod-Pedersen at The Norwegian University of Science and Technology (NTNU) in Trondheim.

7. REFERENCES

[1] Aamodt, Agnar and Enric Plaza (1994): *Case-based reasoning: Foundational issues, methodological variations, and system approaches*. AI Communications, 7(1):39 59, 1994.

[2] Bellifemine, F. Caire, G. Poggi, A. & Rimassa, G. (Sept. 2003), *JADE – a white paper, p.12-16.* Available at: http://sharon.cselt.it/projects/jade/papers/WhitePaperJADEEXP.pdf

[3] Bernt A. Bremdal and Fred Johansen: *CORPORUM technology and applications.* White Paper, CognIT a.s. 2000.

[4] Robert Engels and Till Christopher Lech: *Generating Ontologies for the Semantic Web*. J. Davies, D. Fensel, F. van

Harmelen (eds): Towards the Semantic Web: Ontology-driven Knowledge Management. Wiley, 2002.

[5] Anders Kofod-Pedersen and Agnar Aamodt: *Case-Based Situation Assessment in a Mobile Context-Aware System.* Artificial Intelligence in Mobile System (AIMS 2003). Seattle.

http://ai-gate.cs.uni-sb.de/%7Ekrueger/aims2003/camera-ready/kofod-petersen-6.pdf

[6] Marius Mikalsen and Anders Kofod-Petersen: *Representing and Reasoning about Context in a Mobile Environment*. Modeling and Restrieval of Context (MRC 2004). Ulm, Germany 2004.

[7] Hans Inge Myrhaug: *Towards Life-Long and Personal Context Spaces.* Workshop on User Modelling for Context-Aware Applications. Sonthofen, Germany (July 2001). Available at:

http://orgwis.gmd.de/~gross/um2001ws/papers/myrhaug.pdf

[8] Hans Myrhaug, Nik Whitehead, Ayse Göker, Tor Erlend Faegri and Till Christopher Lech: *AmbieSense – A System and Reference Architecture for Personalised Context-Sensitive Information Services for Mobile Users*. Proceedings of the Second European Symposium on Ambient Intelligence (EUSAI 2004). Eindhoven 2004

[9] Norman M. Sadeh, Enoch Chan, Linh Van: MyCampus: *An Agent-Based Environment for Context-Aware Mobile Services*. Proceedings of Workshop on Ubiquitous Agents on embedded, wearable and mobile devices. (ubiagents 2002) Bologna 2002. Available at:

http://autonomousagents.org/ubiquitousagents/2002/papers/papers/29.pdf

[10] Pavel Vrbal: *Holonic and Multi-Agent Systems for Manufacturing*. In Lecture Notes in Computer Science, Volume 2744 / 2004, pp. 47-58. Heidelberg: 2004

[11] Leendert W. M. Wienhofen: *Mobile Access To Process Knowledge: An Agent Based Approach.* In: proceedings of the 6th IFIP International Conference on Information Technology for Balanced Automation Systems in Manufacturing and Services, Manufacturing in a knowledge-driven society (BASYS 2004), Vienna, 2004.

[12] Will, Matthias O.; Lech, Till Christopher; Klein, Bertin: *Pervasive Knowledge Discovery: Continuous lifelong learning by matching needs, requirements and resources.* In: Proceedings of I-KNOW ’04. Graz: 2004.

[13] Jens Wohltorf, Richard Cissée, Andreas Rieger and Heiko Scheunemann: *An Agent-based Serviceware Framework for Ubiquitous Context-Aware Services*. AAMAS 2004. Available at: http://www.dai-labor.de/fileadmin/files/publications/AAMAS%20Workshop%20Paper.pdf