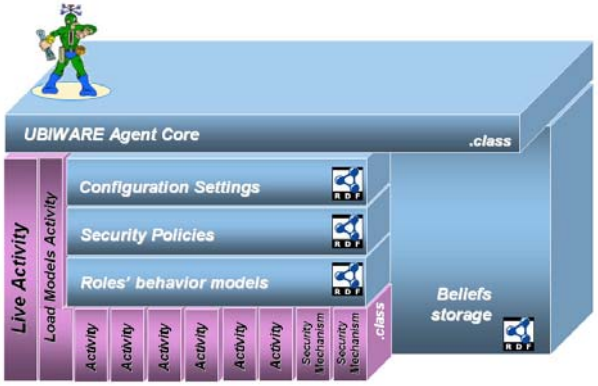
UBIWARE TEKES PROJECT – ANNUAL REPORT (2009-2010)

TEKES

UBIWARE Project

Year 2009-2010



Annual Report

Industrial Ontologies Group

Agora Center, University of Jyväskylä

metso automation shared services

FINGRID

innow

ECURAS

University of Jyväskylä

December, 2010
Jyväskylä, Finland

Table of Contents

1 Introduction 3
2 Project Background Concepts 5
3 Project Results (Year 2009-2010) 10
3.1 The central principles and tools of UBIWARE – Deliverable 3.1 10
3.2 Progress statuses of the industrial cases – Deliverable 3.2 16
3.3 UBIWARE Platform Prototype v.3.0 – Deliverable 3.3 21
3.4 UBIWARE Platform Prototype v.3.1 – Deliverable 3.4 26
4 Achievements of the year 27
4.1 International and local cooperation 27
4.2 Awarded degrees 29
5 UBIWARE Project Publications (up to the end of December 2010) 29
6 References 32

Project's webpage: *http://www.cs.jyu.fi/ai/OntoGroup/UBIWARE_details.htm*

Group's website: *http://www.cs.jyu.fi/ai/OntoGroup/*

Head of UBIWARE Industrial Consortium
(Steering Committee Head) **Dr. Jouni Pyötsiä**,
Metso Automation Oy.

Jouni.Pyotsia@metso.com , Tel.: 040-548-3544

UBIWARE Contact Person **Prof. Timo Tiihonen**,
Vice-Rector, University of Jyväskylä

tiihonen@it.jyu.fi , Tel.: 014-260-2741

UBIWARE Project Leader **Prof. Vagan Terziyan**,
Agora Center, University of Jyväskylä

vagan@it.jyu.fi , Tel.: 014-260-4618

Project's webpage:

http://www.cs.jyu.fi/ai/OntoGroup/UBIWARE_details.htm

1 Introduction

Project Motivation

Recent advances in networking, sensor and RFID technologies allow connecting various physical world objects to the IT infrastructure, which could, ultimately, enable realization of the "Internet of Things" and the Ubiquitous Computing visions. Also, this opens new horizons for industrial automation, i.e. automated monitoring, control, maintenance planning, etc, of industrial resources and processes. A much larger, than in present, number of resources (machines, infrastructure elements, materials, products) can get connected to the IT systems, thus be automatically monitored and potentially controlled. Such development will also necessarily create demand for a much wider integration with various external resources, such as data storages, information services, and algorithms, which can be found in other units of the same organization, in other organizations, or on the Internet.

Such interconnectivity of computing and physical systems could, however, become the "nightmare of ubiquitous computing" (Kephart and Chess, 2003) in which human operators will be unable to *manage* the complexity of interactions, neither even architects will be able to *anticipate* and *design* that complexity. It is widely acknowledged that as the networks, systems and services of modern IT and communication infrastructures become increasingly complex, traditional solutions to manage and control them seem to have reached their limits. The IBM vision of autonomic computing (e.g. Kephart and Chess, 2003) proclaims the need for computing systems capable of "running themselves" with minimal human management which would be mainly limited to definition of some higher-level policies rather than direct administration. The computing systems will therefore be *self-managed*, which, according to the IBM vision, includes self-configuration, self-optimization, self-protection, and self-healing.

The vision of autonomic computing emphasizes that the *run-time* self-manageability of a complex system requires its components to be to a certain degree autonomous themselves. Following this, we envision that the software agent technologies will play an important part in building such complex systems. Agent-based approach to software engineering is also considered to be facilitating the *design* of complex systems.

A major problem is inherent *heterogeneity* in ubiquitous computing systems, with respect to the nature of components, standards, data formats, protocols, etc, which creates significant obstacles for interoperability among the components of such systems. Semantic Web technologies are viewed today as a key technology to resolve the problems of interoperability and integration within heterogeneous world of ubiquitously interconnected objects and systems. The Internet of Things should become in fact the *Semantic Web of Things*[1]. Our vision for this project subscribes to this view. Moreover, we believe that Semantic Web technologies can facilitate not only the discovery of heterogeneous components and data integration, but also the behavioral control and coordination of those components.

[1] David Brock and Ed Schuster (MIT Data Center) at *Semantic Days 2006*, Norway, April 26, 2006, http://www.olf.no/english/news/?30357

Self-management of systems is one of the central themes in the EU 7-th Framework ICT Programme (2007-2013). The Objective "Service and Software Architectures" of the Challenge 1 "Network and Service Infrastructures" includes the need for strategies and technologies enabling mastery of complexity, dependability and behavioral stability, and also the need for integrated solutions supporting the networked enterprise. Also, the Objective "The network of the future" of this Challenge includes the need for re-configurability, self-organization and self-management for optimized control, management and flexibility of the future network infrastructure. In addition, the whole Challenge 2 "Cognition, Interaction, Robotics" has as its motivation the need for creating "artificial systems that can achieve general goals in a largely unsupervised way, and persevere under adverse or uncertain conditions; adapt, within reasonable constraints, to changing service and performance requirements, without the need for external re-programming, re-configuring, or re-adjusting". It is noticeable that the systems (stand-alone or networked) monitoring and controlling material or informational processes is one of the three focus areas of this Challenge.

Project Goals

This project intends to bring the Semantic Web, Distributed AI and Human-Centric Computing technologies to the Ubiquitous Computing domain, especially its industrial cluster. It aims at designing a new generation middleware platform (UBIWARE) which will allow creation of *self-managed* complex industrial systems consisting of mobile, distributed, heterogeneous, shared and reusable components of *different* nature. Those components can be smart machines and devices, sensors, actuators, RFIDs, communication systems and networks, web-services, software, information systems, humans, models, processes, organizations, etc. Such middleware will enable various components to automatically discover each other and to configure a system with complex functionality based on the atomic functionalities of the components.

We believe that tasks of automatic integration, orchestration and composition of such complex systems will be impossible with centralized control due to the scalability issue. Therefore, the components should be to a certain degree autonomous, proactive, and goal-driven. In other words, utilization of the agent technologies is needed to enable flexible communication and coordination of the components. Interoperability among the components requires use of metadata and ontologies. As the amount of components can grow dramatically, without their ontological classification and (semi- or fully-automated) semantic annotation processes, the automatic discovery will be impossible.

Project Stages

Research and development within this project follows the following main directions (workpackages):

1. Core Distributed AI platform design (UbiCore);
2. Managing Distributed Resource Histories (UbiBlog);
3. Smart Ubiquitous Resource Privacy and Security (SURPAS);
4. Self-Management, Configurability and Integration (COIN);
5. Smart Interfaces: Context-aware GUI for Integrated Data (4i technology);
6. Middleware for Peer-to-Peer Discovery (MP2P);
7. Industrial cases and prototypes.

UBIWARE requires the reliable core platform to enable semantics-based proactivity and coordination of the components (workpackage WP1) and tools for managing and integrating distributed histories of the components (workpackage WP2). It also requires essentially new solutions towards security, service provisioning and information integration. Ubiquitous computing environment demands ubiquitous, yet flexible, security with respect to all kinds of interactions, based on well-defined and machine-readable policies (workpackage WP3). In UBIWARE, there is a need for new solutions towards self-management, configuration and integration of the components (workpackage WP4) and flexible semantic interfaces to deliver the integrated data to different human users having different backgrounds (workpackage WP5). UBIWARE should allow resource discovery not only based on centralized registries but also on the peer-to-peer basis (workpackage WP6). Finally, the middleware should be tried on real industrial cases to evaluate the scientific concepts behind it and facilitate its further utilization (workpackage WP7).

Workpackages 1 through 6 include both research and development tasks. The above tasks will be approached by combining various research methods with agile software development processes. This means that software prototypes will be iteratively developed during the whole project lifecycle based on real data, real needs and changing requirements of industrial partners. The result will be both the basic software tools for the UBIWARE platform and several industrial cases prototyped based on these tools. Prototypes of UBIWARE, integrating the work in the workpackages at different levels of their readiness, are developed during each project year, as UBIWARE 1.0, UBIWARE 2.0, UBIWARE 3.0 and UBIWARE 3.1, and reported through deliverables D1.3, D2.3, D3.3 and D3.4, correspondingly. The status reports for the industrial cases are collected in separate deliverables (one per year) D1.2, D2.2, and D3.2.

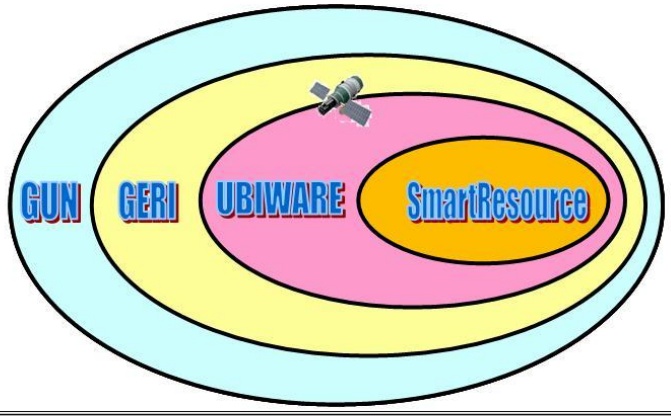
2 Project Background Concepts

This project is a next step of our research group towards the *Global Understanding Environment (GUN)* (Terziyan, 2003, 2005; Kaykova *et al.*, 2005a). The first step was done in the SmartResource project (2004-2006). Figure 1 depicts our research roadmap. A very general view on GUN is presented in Figure 2.

When applying Semantic Web in the domain of ubiquitous computing, it should be obvious that Semantic Web has to be able to describe resources not only as passive functional or non-functional entities, but also to describe their behavior (proactivity, communication, and coordination). In this sense, the word “global” in GUN has a double meaning. First, it implies that resources are able to communicate and cooperate globally, i.e. across the whole organization and beyond. Second, it implies a “global understanding”. This means that a resource A can understand all of (1) the properties and the state of a resource B, (2) the potential and actual behaviors of B, and (3) the business processes in which A and B, and maybe other resources, are jointly involved.

Global Understanding Environment (GUN) aims at making heterogeneous resources (physical, digital, and humans) web-accessible, proactive and cooperative. Three fundamentals of such platform are *Interoperability*, *Automation* and *Integration*. Interoperability in GUN requires

utilization of Semantic Web standards, RDF-based metadata and ontologies and semantic adapters for the resources. Automation in GUN requires proactivity of resources based on applying the agent technologies. Integration in GUN requires ontology-based business process modeling and integration and multi-agent technologies for coordination of business processes over resources.



GUN (Global Understanding Environment) – Proactive Self-Managed Semantic Web of Things - general concept and final destination

GERI (Global Enterprise Resource Integration) – GUN subset related to industrial domains

UBIWARE – middleware for GERI

SmartResource – semantic technology, pilot tools and standards for UBIWARE

***Figure 1** - The research roadmap towards GUN.*

Main layers of GUN can be seen in Figure 2. Various resources can be linked to the Semantic Web-based environment via adapters (or interfaces), which include (if necessary) sensors with digital output, data structuring (e.g. XML) and semantic adapter components (XML to Semantic Web). Software agents are to be assigned to each resource and are assumed to be able to monitor data coming from the adapter about the state of the resource, make decisions on the behalf on the resource, and to discover, request and utilize external help if needed. Agent technologies within GUN allow mobility of service components between various platforms, decentralized service discovery, FIPA communication protocols utilization, and multi-agent integration/composition of services.

When applying the GUN vision, each traditional system component becomes an agent-driven "smart resource", i.e. proactive and self-managing. This can also be recursive. For example, an interface of a system component can become a smart resource itself, i.e. it can have its own responsible agent, semantically adapted sensors and actuators, history, commitments with other resources, and self-monitoring, self-diagnostics and self-maintenance activities. This could guarantee high level of dynamism and flexibility of the interface. Such approach definitely has certain advantages when compared to other software technologies, which are integral parts of it, e.g. OOSE, SOA, Component-based SE, Agent-based SE, and Semantic SE. This approach is also applicable to various conceptual domain models. For example, a domain ontology can be considered as a smart resource, what would allow having multiple ontologies in the designed system and would enable their interoperability, on-the-fly mapping and maintenance, due to communication between corresponding agents.

***Figure 2** - The concept of Global Understanding Environment.*

The SmartResource project, in its research and development efforts, has made some steps towards Global Understanding Environment by decomposing it into three main parts, and analyzing each.

The first is the *General Adaptation Framework (GAF)* for semantic interoperability. GAF provides means for semantic description of industrial resources, including dynamic and context-

sensitive information. The central part is GAF is played by the Resource State/Condition Description Framework (RscDF). An implementation of GAF for a specific domain is supposed to include also an appropriate RscDF-based domain ontology, an appropriate RscDF Engine and the family of so called “Semantic Adapters for Resource” to provide an opportunity to transform data from a variety of possible resource data representation standards and formats to RscDF and back. For more details about RscDF and GAF see (Kaykova *et al.,* 2005b) and (Kaykova *et al.,* 2005a).

The second is the *General Proactivity Framework (GPF)* for automation and proactivity. GPF provides means for semantic description of individual behaviors by defining the Resource Goal/Behavior Description Framework (RgbDF). An implementation of GPF is supposed to include also an appropriate RgbDF-based domain ontology, an appropriate RgbDF engine and a family of “Semantic Adapters for Behavior” to provide an opportunity to transform data from a variety of possible behavior representation standards and formats to RgbDF and back. See more on RgbDF in (Kaykova *et al.,* 2005c).

The third is the *General Networking Framework (GNF)* for coordination and integration. GNF provides means for description of a group behavior within a business process. It specifies the Resource Process/Integration Description Framework (RpiDF), and an implementation of GNF is supposed to include also an appropriate RpiDF-based domain ontology, an appropriate RpiDF engine and a family of “Semantic Adapters for Business Process” to provide opportunity to transform data from a variety of business process representation standards and formats to RpiDF and back.

Finally, GUN ontologies will include various available models for describing all GAF-, GPF- and GNF- related domains. The basis for interoperability among RscDF, RgbDF and RpiDF is a universal triplet-based model provided by RDF and two additional properties of a triplet *(true_in_context* and *false_in_context*). See more about contextual extension of RDF in (Khriyenko and Terziyan, 2006).

As said above, the UBIWARE project is intended to continue our work towards GUN. The SmartResource project analyzed the central GUN concepts and resulted in some, more or less separated, pilot tools and solutions. In contrast, the UBIWARE project will result in a complete and self-sufficient middleware platform. For this, UBIWARE will integrate SmartResource ideas, elaborate them, and extend with related solutions in supporting but mandatory areas such as security, human interfaces and other.

In this project, we will naturally integrate the Ubiquitous Computing domain with such domains as Semantic Web, Proactive Computing, Autonomous Computing, Human-Centric Computing, Distributed AI, Service-Oriented Architecture, Security and Privacy, and Enterprise Application Integration. We will finish with a real prototype of the UBIWARE for industrial needs as a key toolset for future "Global Enterprise Resource Integration" (GERI) Platform. UBIWARE should bring the following features to industrial partners: Openness, Intelligence, Dynamics, Self-Organization, Seamless Services and Interconnectivity, Flexibility and Reconfigurability, Context-Awareness, Semantics, Proactivity, Interoperability, Adaptation and Personalization, Integration, Automation, Security, Privacy and Trust.

In one sense, our intention to apply the concepts of automatic discovery, selection, composition, orchestration, integration, invocation, execution monitoring, coordination, communication, negotiation, context awareness, etc (which were, so far, mostly related only to the Semantic Web-Services domain) to a more general "Semantic Web of Things" domain. Also we want to expand this list by adding automatic self-management including (self-*)organization, diagnostics, forecasting, control, configuration, adaptation, tuning, maintenance, and learning.

According to a more global view to the Ubiquitous Computing technology:

- UBIWARE will classify and register various ubiquitous devices and link them with web resources, services, software and humans as business processes' components;
- UBIWARE will consider sensors, sensor networks, embedded systems, alarm detectors, actuators, communication infrastructure, etc. as "smart objects" and will provide similar care to them as to other resources.

Utilization of the Semantic Web technology should allow:

- Reusable configuration patterns for ubiquitous resource adapters;
- Reusable semantic history blogs for all ubiquitous components;
- Reusable semantic behavior patterns for agents and processes descriptions;
- Reusable coordination, design, integration, composition and configuration patterns;
- Reusable decision-making patterns;
- Reusable interface patterns;
- Reusable security and privacy policies.

Utilization of the Distributed AI technology should allow:

- Proactivity and autonomic behavior;
- Communication, coordination, negotiation, contracting;
- Self-configuration and self-management;
- Learning based on liveblog histories;
- Distributed data mining and knowledge discovery;
- Dynamic integration;
- Automated diagnostics and prediction;
- Model exchange and sharing.

Utilization of the Human-Centric approach enables us to consider humans in four possible roles (*Figure 3*):

- *Human as UBIWARE user* will get unique access to integrated and adapted services and information;
- *Human as UBIWARE service provider* will get support in online service provisioning and benefit as a servicing component in various business processes (e.g. a maintenance expert);
- *Human as UBIWARE resource* will be able to get online care from integrated distributed resources and services (e.g. monitoring the health of an employee);
- *Human as UBIWARE administrator* will be able to launch and configure UBIWARE for a particular task.

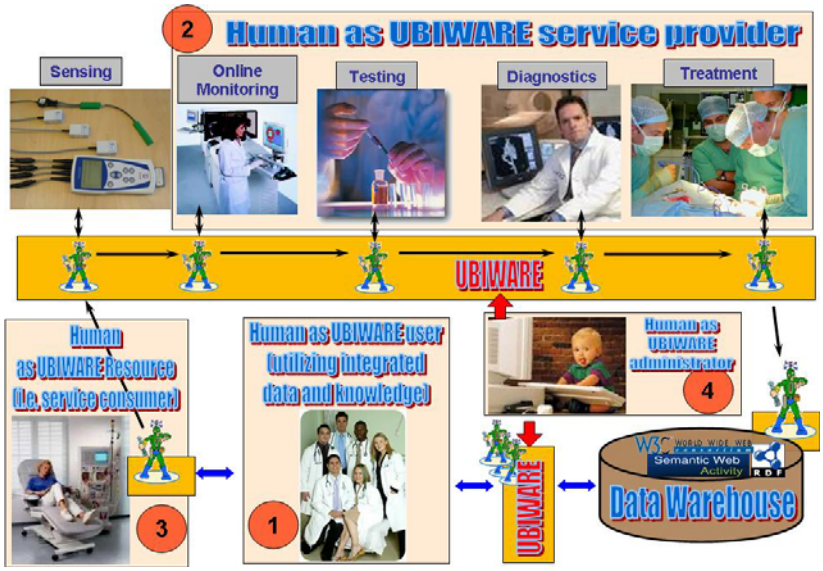


Figure 3 *- Human in different roles in UBIWARE.*

3 Project Results (Year 2009-2010)

The research results from the workpackages are reported through three integrating deliverables (one per project year):

1. *D3.1:* The central principles and tools of UBIWARE;
2. *D3.2:* Progress statuses of the industrial cases;
3. *D3.3:* UBIWARE Platform Prototype v.3.0.

Due to extension of the thrid last year of the project we had an additional deliverable *D3.4* (UBIWARE Platform Prototype v.3.1) that includs improvement of the plarform core functionality and edvanced use case presentation.

3.1 The central principles and tools of UBIWARE – Deliverable 3.1

WP1: UbiCore

During WP1's Year 3 (the *Coordination phase*), we investigated the approaches where behavioral S-APL models are used not only as prescriptive tool (i.e. loaded by agent to act based on them), but also as descriptive tool - accessed by other agents to e.g. understand what to expect from or how to interact with the agent in question. The mail research questions were:

- How to enable agents to flexibly discover each other, based both on the roles played and on particular capabilities possessed.
- What would be concrete benefits of and what mechanisms are needed for accessing and using a role's script by agents who are not playing that role but wish to coordinate or interact with an agent that does?

Service advertising and discovery are an intrinsic part of the process management scenarios. In the open environment the value of it should not be underestimated. With the adoption of automated contracting and legal mechanisms in the services world, the dynamic matchmaking and service discovery will become a cornerstone of business process management. We perceive that process management will be driven by autonomous entities with certain degree of freedom to take decisive actions. The foresight of the future leads us towards further directions the dynamic process management area.

In particular, within the framework defined in the introductory part of this Section we will define behavioral patterns and data structures that specify how the agent should (re-)act in order to support servicing agent interaction scenarios. The scenarios are domain independent and constitute three infrastructural layers: execution layer, process management layer and configuration layer.

The execution layer defines how one agent can request another agent to execute an action and receive action results.

Process management layer is built on top of the execution layer and allows agents to control composite actions in runtime, e.g. before the actual execution starts, the process handler agent can request for availability of candidate agents, if they confirm participation in the process or not. This layer also involves contracting.

The configuration layer specifies a meta-level for agent-to-agent interactions that allows agents to negotiate about the processes they run and cooperatively agree on changes.

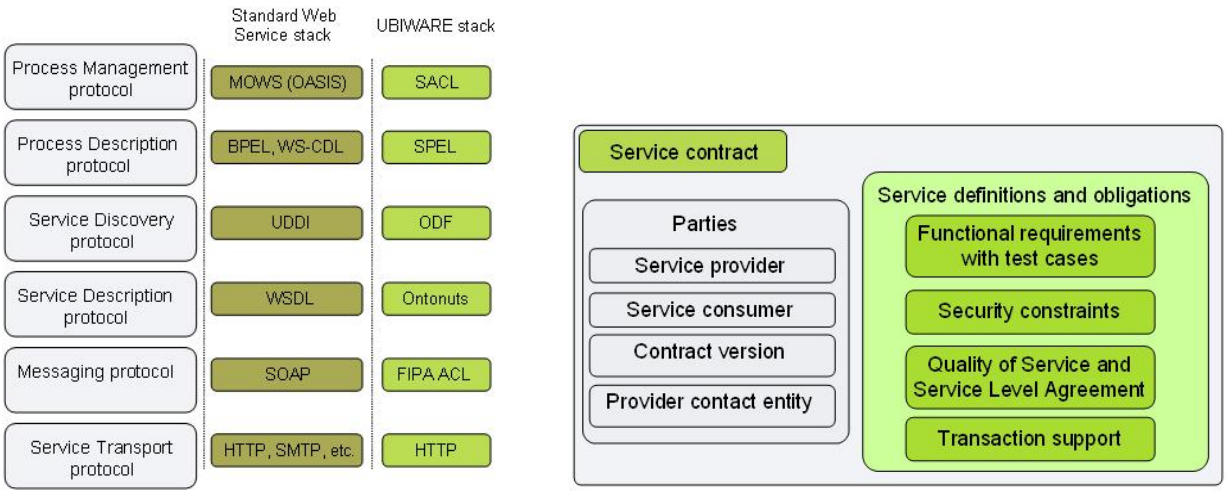


Figure 4 – *Web Services stack vs. UBIWARE and service contract specification.*

WP2: UbiBlog

The third year phase of the WP2 (Mining phase) as it was specified before would rather be an application case of the Ontonuts technology than a research topic. With respect to this fact we have decided to concentrate our efforts towards the fusion of the WP1 and WP2 3rd year objectives. Shortly, the WP1 objective for the 3rd year is to elaborate a mechanism for advertising of agent capabilities and role scripts (complex capabilities) amongst agents, whereas WP2 research targets data mining capabilities only. Therefore, it is reasonable to integrate WP1 and WP2 under one umbrella of Ontonuts technology which is targeting even more ambitious

goal – to enable automated (re)planning and execution of semantically annotated agent actions including distributed data querying, data mining as well as distributed agent-to-agent servicing.

We foresee that model player services will be a successful business case for the emerging paradigm of cloud computing. Pay-per-use principles combined with high computational capacities of cloud and standardized DM-models will be definitely an alternative to expensive business intelligence and statistics toolkits.

Another niche of data mining services in cloud computing can be model construction services. Such systems will drive innovations in data mining methods as well as applied data mining in certain domains. Such service will compete by introducing know-how and innovative tools and algorithms that bring add-values in e.g. predictive diagnostics or computational error estimation. This direction will lead to so-called "web of intelligence"[2].

The role of UBIWARE in cloud computing emerges as a cross-cutting management and configuration glue for interoperability of future intelligent cloud services (see Figure 5).

The main burden of UBIWARE will be management of consistency across different domain conceptualizations (Ontologies) and cross-domain middleware components. Fine-grained ontology modeling is still a challenge for research community and we predict that in the nearest future the domain modeling will be task-driven, i.e. the domain model engineers may incorporate some standardized and accepted conceptualizations, whereas the whole ontology for solution will be tailor made. Tailored ontologies will require subsequent mapping mechanisms and additional efforts. Nevertheless, we have to cope with it because building one centralized world ontology has been reasonably criticized as utopia.

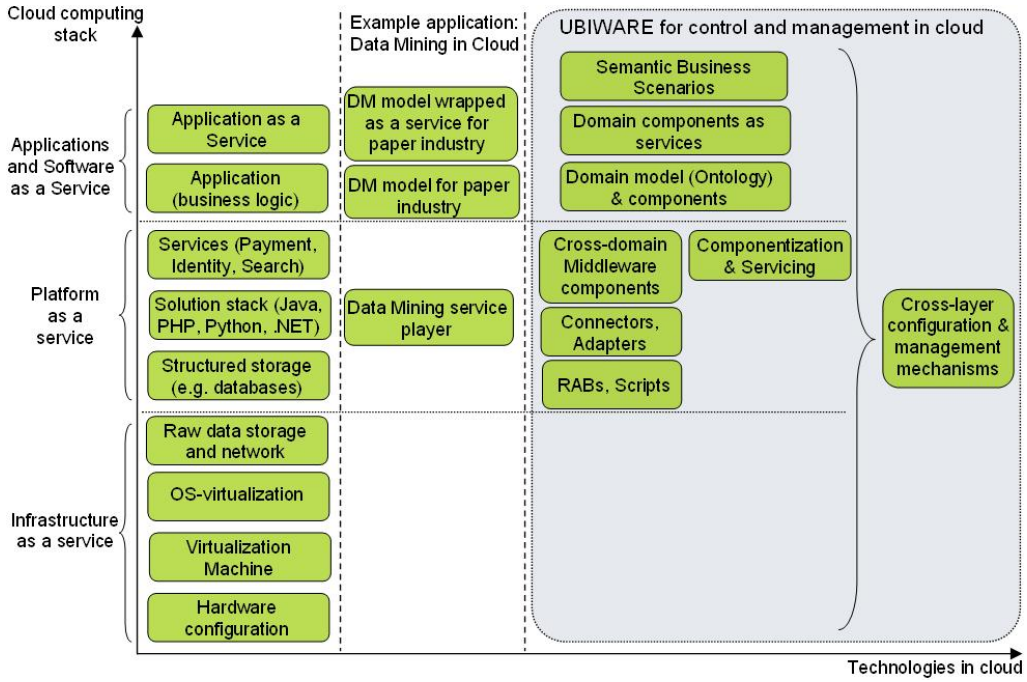


Figure 5 – *Role of UBIWARE in cloud computing.*

[2] Terziyan, 2009, www.cs.jyu.fi/ai/ICNS-2009.ppt

Self-management and configurability in UBIWARE can be seen from two points of view:

- Initial self-configuration
- Runtime self-configuration

Initial self-configuration is understood as the ability of the system to interconnect and configure its components based on a certain goal or policy specified by the user. After specifying the goal of the system, the system itself should be able to automatically choose proper agents and delegate proper roles to them. The result should be a system that is performing the task specified by the goal. The user does not have to provide the system with any code, only the goal is needed. The system will find the best configuration based on the goal specified and a domain specific ontology.

Runtime self-configuration is the ability of the system to adapt to the environment. Thanks to this ability the system is able to perform its task even if the circumstances change. The process of runtime self-configuration should be context-aware, ontology-driven and policy-based.

Based on the text above, we specify the following research questions. First two are related to initial self-configuration and the third question is related to runtime self-configuration.

- How can we find suitable agents to perform the task based on the goal specified by the user?
- How can we transform a goal into a set of scripts to be executed by these agents?
- Once the system is configured, how can we maintain this state even if the circumstances of the system change?

In this workpackage we discussed issues involving initial process configuration and runtime self-management. In the beginning we defined terms like abstract process, executable process, abstract step, binding, etc. We proposed architecture for initial configuration involving Abstract process repository, director agent, ODF and several Ontonut implementations. We described the initial process configuration phase, in which an executable process is built based on an abstract process. This phase involves Ontonut binding and variable binding.

Later, we discussed the issue of self-management in runtime. We used a three layer self-management architecture proposed by Kramer and Magee and adapted it to suit our needs. The first layer of the architecture is implemented in agents running particular Ontonuts. They monitor the Ontonut execution and if there is an exceptional situation that the Ontonut cannot handle, it is reported to the second layer. The second layer is implemented in the director agent and its goal is to deal with situations that were not handled by the first layer. Depending on the user context, the second layer will try to use an alternate Ontonut or contact the user to ask for additional instructions. As an open philosophical question we leave the issue of replacing a single Ontonut with prepared partial plans. Another open question is the issue of binding Ontonuts whose precondition is broader than the precondition of the Ontonut being replaced. The third layer is mostly dealing with planning. In case the second layer fails to resolve the problem, the third layer will have to replan and reach the original goal using another set of Ontonuts. At the same time we discussed related work in the field of self-management and its impact on Ubiware.

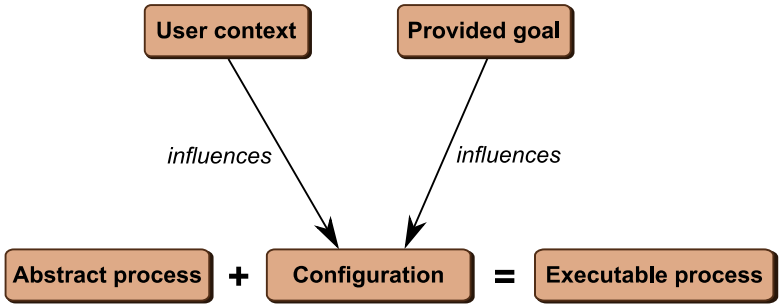


Figure 6 - *The relationship between an abstract process and an executable process.*

WP5: 4I (FOR EYE) technology

This workpackage studies dynamic context-aware Agent-to-Human interaction in UBIWARE, and elaborates on a technology which we refer to as 4i (FOR EYE technology). From the UBIWARE point of view, a human interface is just a special case of a resource adapter. We believe, however, that it is unreasonable to embed all the data acquisition, filtering and visualization logic into such an adapter. Instead, external services and application should be effectively utilized. Therefore, the intelligence of a smart interface will be a result of collaboration of multiple agents: the human's agent, the agents representing resources of interest (those to be monitored or/and controlled), and the agents of various visualization services. This approach makes human interfaces different from other resource adapters and indicates a need for devoted research. 4i technology will enable creation of such smart human interfaces through flexible collaboration of an Intelligent GUI Shell, various visualization modules, which we refer to as MetaProvider-services, and the resources of interest.

According to the discussion during the last 2nd project year steering group meeting, we agreed to concentrate 3rd year project research on business issues, commercialization steps of the results. After the 2nd year we had clear vision of the idea, had the elaborated prototype of an initial idea, we became ready for the next valuable step during WP5's Year 3. This step consist of two parts: elaboration of the general architecture of the product (necessary components, tools and utilization models) and commercialization part (business and market analysis, business models, promotion, distribution and etc.). During WP5's Year 3 (the *Commercialization phase*), therefore, the following research questions were to be answered:

- What should be the general architecture of the product – 4I(FOR EYE) tool package so that it will be possible to build and further extend a different services based on the product? What are the requirements for the product, for the product components, what are the necessary modifications and the use cases of the product utilization?
- What are the commercialization and marketing steps?

To become a product, prototype should pass through a number of stages that bring generalization to the prototype (ability to be used in different domains and arias), provide common information infrastructure and user/programming interfaces of the system to allow extension and configuration of it. One of the challenges is ontology creation task. No doubts that the best way is to develop common ontology. But, practically, more realistic way is to start from building local domain and task-specific ontologies and further apply adaptation and bridging technologies

to allow interoperability between systems based on different ontologies. Current prototype does not have separate ontology as such, and uses some limited (task specific) imbedded one. Thus, ontology/vocabulary creation and manipulation mechanism is one of the challenges that has to be taken into account.

As was mentioned before, ability of the Browser to connect/import external information repositories is very important functionality. GUI-Shell should provide user interface to select external repository to be imported via appropriate adaptation module from a list of available in Browser as well as interface for search of new adaptation modules and their imbedding. Regarding to the plan for Inno-W industrial case, we are going to elaborate adaptation module to convert their RDF repository into internal format. It will be just one adaptation module, but we are going to implement general functionality of the Browser according to modular approach of source data adaptation.

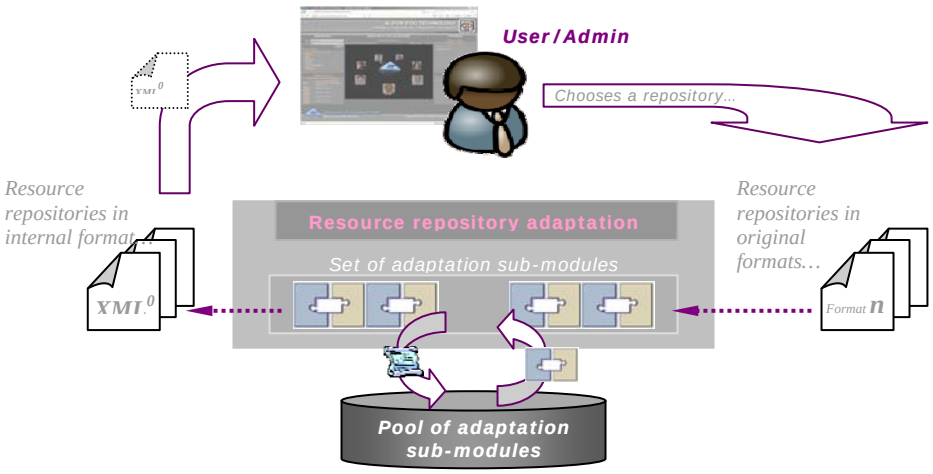


Figure 7 *– Source data adaptation.*

The same modular approach also can by apply for resource visualization context definition and creation. Following such approach, definition of new context will be accompanied not only by correspondent description, but also by JavaScript code, correspondent functional server part and appropriate html, media and other files. Then extension of the Browser with new context will be conducted via installation of add-on package. Extension of the Browser with new MetaProvider could be much simple and be conducted via extension of repository of MetaProvider's descriptions with a new one.

Concerning the commercialization steps of the Browser, we have considered general model of 4I Environment where we tried to define main players and roles. We described two business scenarios of Browser utilization: *Global Use of the Browser* and *Local Corporate Use of the Browser*. So, as in a case of ontology creation we think that the easiest way to start with the second scenario, but this scenario can evolve to a Web of separated Corporate 4I Environments later. In this case we will face a problem of interoperability of the systems especially on the level of ontology specification, resource annotation and context creation. Thus, with further evolution of the scenario we will come to the first scenario with more complex structure. Still, it is more realistic scenario, but of course, the best scenario is the first one, even if it demands more efforts and investments.

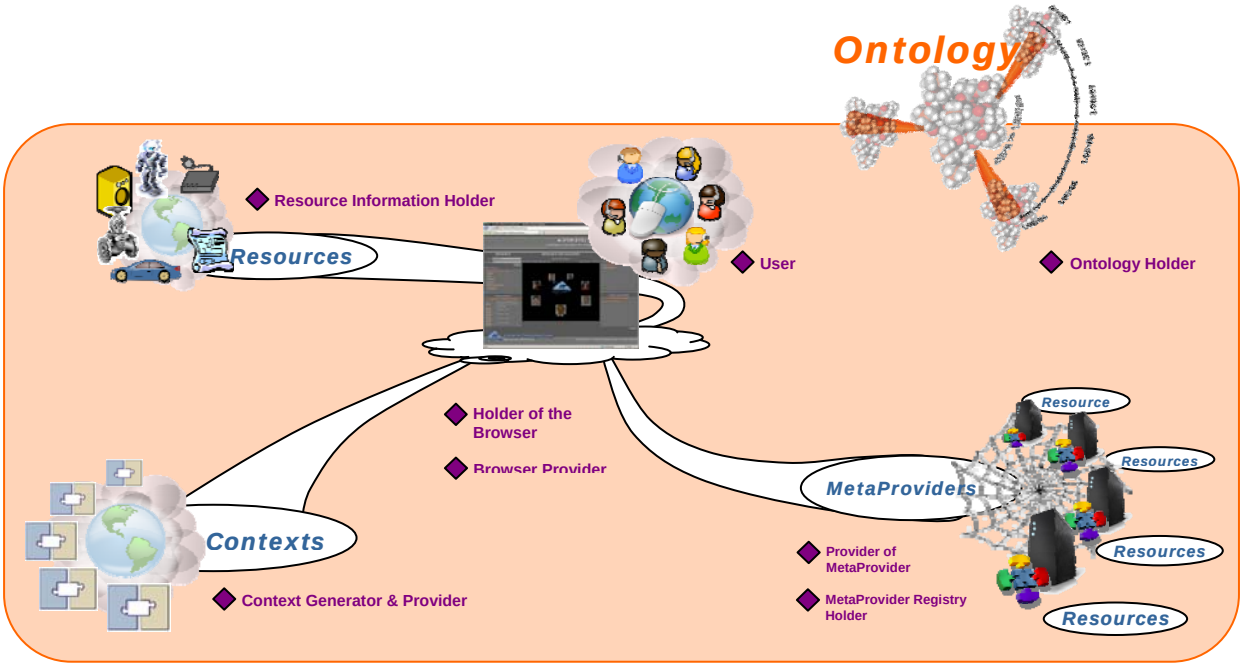


Figure 8 *– General model of 4I Environment.*

Finally, we have presented intelligent way of automatic/semiautomatic context recognition and personalized visualization invocation as a next valuable enhancement of the Browser.

3.2 Progress statuses of the industrial cases – Deliverable 3.2

WP7:

The objective of this workpackage is to trial UBIWARE on real industrial cases. This has two major goals for such case studies. The first goal is to evaluate the scientific concepts behind UBIWARE and to find problems and issues in UBIWARE that would otherwise be overlooked. The second goal is to facilitate the further utilization of UBIWARE in the industry. Several specific cases, proposed by the industrial partners, are analyzed, designed and prototyped based on the UBIWARE platform. The reasons for prototyping are the same: to identify issues in UBIWARE that would get overlooked if the work was only theoretical and thus abstract, and to demonstrate the benefits of UBIWARE in a tangible way so to facilitate future industrial adoption.

In the 3rd project year we deliver three industrial cases, those of Fingrid, Inno-W and Metso Automation.

During the Year 3, with respect to all three cases the task is the following:

Task T3.1_w7: Developing a full prototype application: connecting to additional relevant resources and extending the interactions between them towards a sufficiently elaborated application.

Fingrid case:

With respect to the UBIWARE approach and platform, Fingrid's main area of interest is in organizing smart data management related to the events/alarms which company gets from their control systems. Existing systems do not provide many possibilities for managing this data beyond storing it to a time-series log, and browsing it with some filtering possibilities. A wish is to that the data should get flexibly accessible, integrated with other related data, and possibilities should be provided for producing generalizing reports to the power system operation and asset management persons.

Fingrid has the following two databases that were so far the objects of interest in the UBIWARE's Fingrid case:

- **Event History database**: **Eventlog** (Oracle) in the office environment, to which data is automatically replicated from SCADA's event history database. A record in this database contains such information as the time of the event, event class, access area, substation ID, device ID, the state of the object, and some other.
- **Elnet database** (Oracle) that stores information about all the equipment, including circuit-breakers, disconnectors, transformers, capacitors, and other. A record in this database contains such information as device group, device ID, ownership (Fingrid or external), and other.

The unique device ID present in both databases enables join queries.

This industrial case was developed by Industrial Ontologies Group for Fingrid Oyj as a part of UBIWARE project (2007-2010). We provided a User's Guide that primarily describes the improvements developed within the third year of the project. It includes only minimal documentation related to work performed in the first and second year of the project.

In the beginning of the development phase, Fingrid specified these proposals:

Proposal code	Description
P1	Operation counts and operation time of compressors in compressed-air plants (Paineilmalaitosten kompressorien toimintakerrat ja käyntiaika)
P2	Operation time of compensation equipments (capacitors, reactors) and transformers (Kompensointilaitteiden ja muuntajien käyntiaika)
P3	Operation counts of circuit-breakers and disconnectors owned by Fingrid after the last maintenance (Katkaisijoiden ja erottimien toimintakerrat viime huollon jälkeen)
P4	Adding of new filtering conditions for equipment alarms of the job responsibility areas (R1 alarms) (Työaluekohtaisten laitehälytysten suodatusehtojen lisäys)
P5	Protection alarms to the experts of protection by email (Suojaushälytykset suojausasiantuntijoille)
P6	Tripping alarms will be sent more often than once a day, for example once a hour (Laukaisutiedot tiheämmällä lähetysvälillä)
P7	Developing of the user interface for management of filtering conditions (Käyttöliittymän kehittäminen suodatusehtojen hallintaa varten)

Table 1 – *Fingrid proposals.*

Proposals P1 and P2 were being solved together and the result is available through page Operation_time_counts.html. Proposal P3 is in the waiting state due to establishment of testing environment. Proposals P4-P7 were solved together and the result is available through the administrator's interface, which will be described later.

User's Guide includes general description of the package structure, hardware and software requirements, installation and execution, administrative interface.

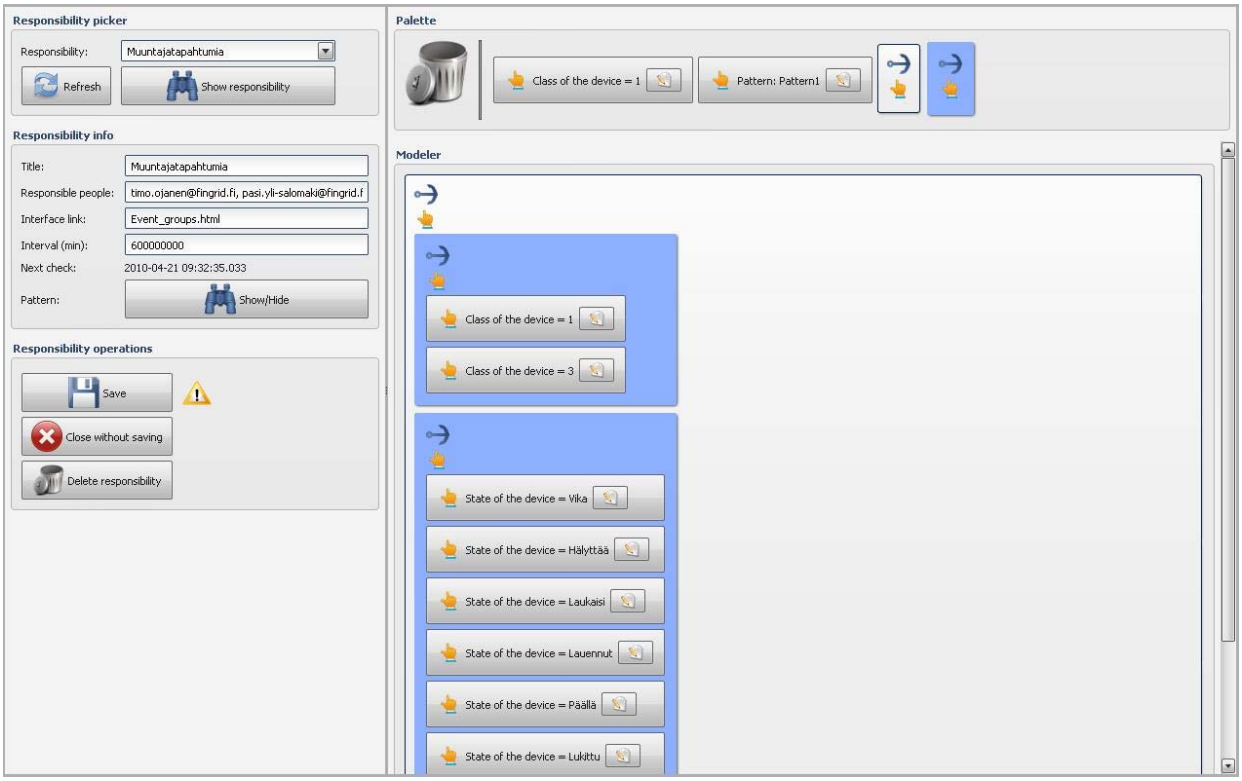


Figure 9 – *User interface with modeler.*

At the same time we provided Developer's Guide for those who wish to understand this industrial case. Its main audience are programmers, there it will be more technical and written in a slightly less formal way. It is also a sort of a diary that can be used in the future to extend the industrial case with additional functionality. This documentation is not comprehensive, because the source code itself contains comments. This document is supposed to give the programmer an overall view of the problem. It is recommended to read the user's guide first (at least till installation), because it contains information about proposals and other issues related to the case. It includes technologies used, operation counts and times, and administrative interface.

Inno-W case:

The plan for the implementation of industrial cases for Inno-W Company was to elaborate adapter for a real RDF data storage of proposals to be browsed through 4I (FOR EYE) Browser in the context of close/similar resources. This case is a test bed for a research and development within the WP5.

According to the general architecture of 4I (FOR EYE) Browser, adaptation of data sources is done via importing the source to the Browser and converting the data to the internal format. At the same time with converting the original data format to internal one, adapter builds all necessary visualization context descriptions and provides visualization module (MetaProvider) binding information.

Current adapter works with both input formats: RDF with XML serialization and RDF in N-triple format. The data type of the source storage provided by Inno-W Company for the current industrial case is N-triple RDF. Inno-W Company provided us access to their portal where we can create sample set of proposals. Following URL *"http://mvi.inno-w.com/triplify"* is used to retrieve the RDF source from the database.

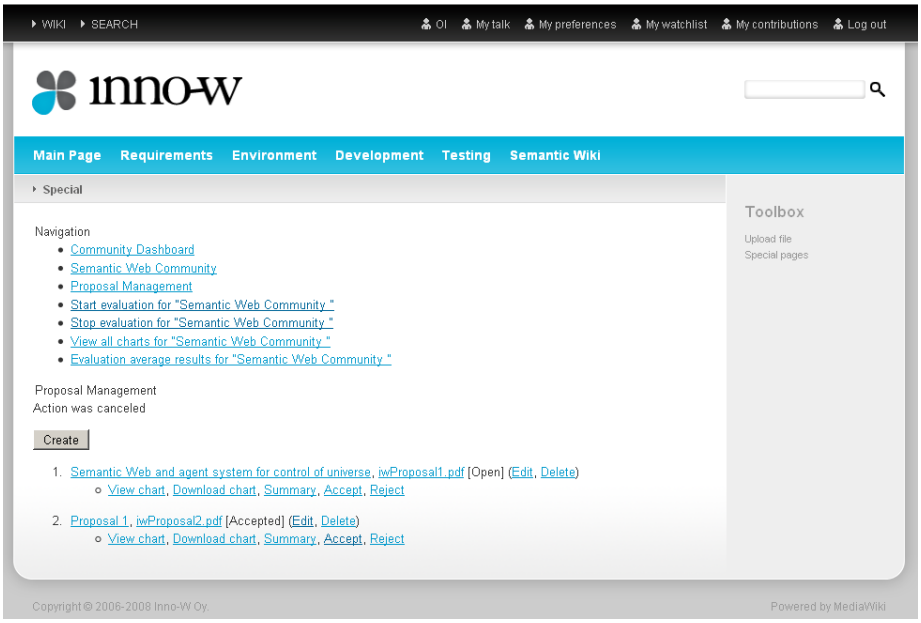


Figure 10 – *Inno-W portal –editing of the proposals.*

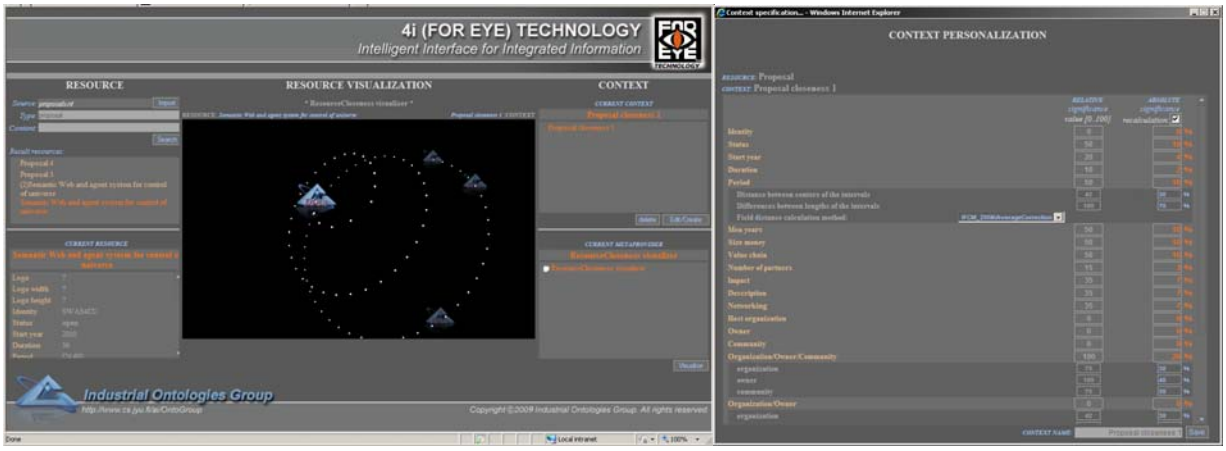


Figure 11 – *4I (FOR EYE) Browser and Resource closeness visualization context editor.*

Metso case:

Metso industrial case was selected to be a test bed for a research and development within the WP2 (Distributed Querying and Integration) because of its "distributed nature" and big amounts of data in storages, that cannot be collected in one place.

In the previous version of the industrial prototype we have integrated event flow data (events from monitoring and diagnostic systems) together with the structural and design data to provide a convenient assistant tool for an expert in diagnostics. In other words, ease the access to the relevant information needed for decision making. We have integrated the most significant parts (derived from the use case) of the data samples provided. Those significant parts were semantically adapted. "Semantically adapted" means development of components, that represent the actual data sources as a virtual memory. I.e. the data from data sources was not fully transformed into S-APL, but it was annotated to answer the semantic queries instead. The description/annotation of the components refers to the domain ontology, which makes the specification explicit. At the same time, the descriptions themselves are not enough to run queries; they are interpreted by the Ontonuts engine and then used in planning and execution. In the new version of the prototype we have made a manager for the components/Ontonuts developed. The manager provides functionality for editing and making test runs for the newly updated component descriptions.

The "Agent Component Manager" is developed as a web-based user interface that interacts with the UBIWARE platform. The interface is constructed with HTML and JavaScript technologies, using Qooxdoo[3] open-source JavaScript framework. The UI communicates with the UBIWARE platform via HTTP, in particular with the agent whose components are being managed.

The User Interface has three tabs. In the first tab, named "Browse/Edit", user is able to search and configure/update the existing components. At the moment the interface is tailored to certain type of components – Ontonuts. Or, being more precise, to the Donut type of the Ontonut, which is a semantic database connector. The user can edit the component in a nutshell, e.g. update the database query that is going to be executed, change the database URL, port number or username and password. In second tab, named "Component Player", the UI allows the user to specify a call to the component by adding and removing conditions/constraints. The user can execute the component call and see the results.

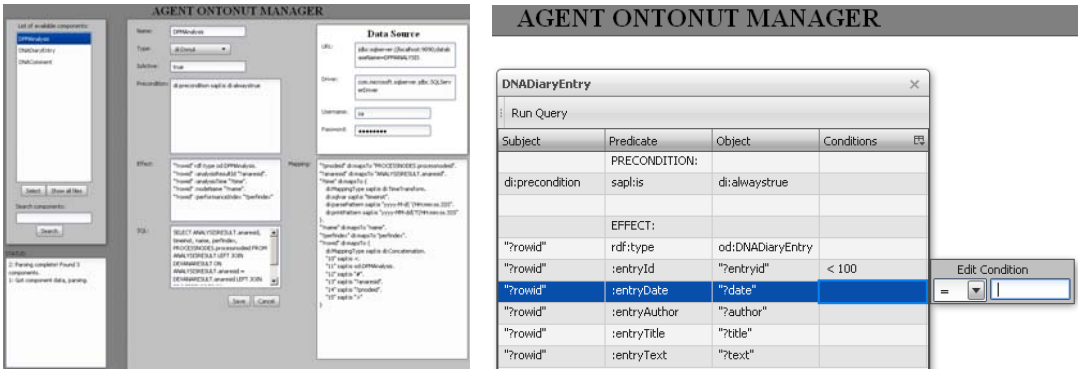


Figure 12 – *The Browse/Edit and the Component player tabs.*

[3] www.qooxdoo.org

The third tab is reserved for the Settings of the editor itself and will be developed in the future versions of the manager.

The ACM tool has been developed using Metso industrial maintenance data sources as test samples. In general, the tool can be applied to other problem domains as well. The main development direction will continue towards extending component types supported and then the component process chains building.

3.3 UBIWARE Platform Prototype v.3.0 – Deliverable 3.3

Main platform development:

In the year 3 of the platform evolution we have put our main effort to the platform usability issues. To make the platform attractive as a middleware solution, we have to offer a set of platform features that are comparable with other software development middleware available on the market today. Furthermore, to be able to demonstrate the benefits of the platform, we have to show a clear add value the platform may offer.

This deliverable has brought the UBIWARE platform to the qualitatively new level of the middleware solution – the platform now combines the features of the application server, the semantic web platform and the agent-driven platform, where agent-driven semantic applications can serve end customers with the high quality web-based GUIs, enhanced user-friendliness and responsiveness. The platform has become an application-independent runtime environment, where special infrastructure agents take care of the platform itself, not of the applications being run on it. At the same time, we introduce personal user agents, thus making the platform user-oriented.

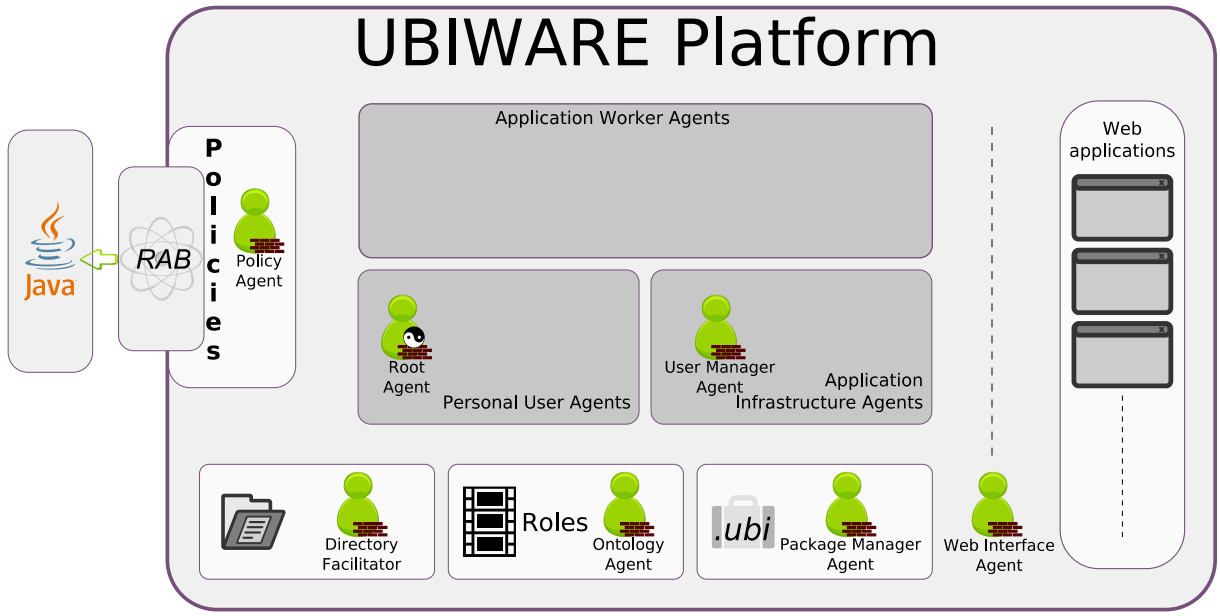


Figure 13 *– UBIWARE 3.0 platform architecture.*

We design the UBIWARE platform infrastructure for creation of various kinds of applications. Those applications have a freedom to use a web front-end, on-the-platform user management and other infrastructure or define their own platform components depending on the needs of the application. In the UBIWARE 3.0 architecture we identify two groups of agents. The first group includes the agents which are application-specific, whereas the second group gathers infrastructure agents providing services to those application-specific ones.

In the previous version of the UBIWARE platform the only way for agents to offer the communication with external systems was the AgentServer reusable atomic behavior and ServerEvent class. AgentServer and ServerEvent provided a way of sending and receiving messages through TCP sockets. This was convenient for machine to machine interaction and for simple user interfaces, but was lacking features, such as thread pooling, caching and support for existing tools, - those needed for building more sophisticated web based applications. The new web application architecture based on embedded Jetty HTTP server is designed to take UBIWARE to the modern web.

The main idea behind the new web architecture is to allow web developers to use their old and reliable and/or new and shiny web development tools and frameworks for the UI development while providing easy and unobtrusive integration to the agent platform. The high level general architecture for UBIWARE web application is depicted in Figure 14.

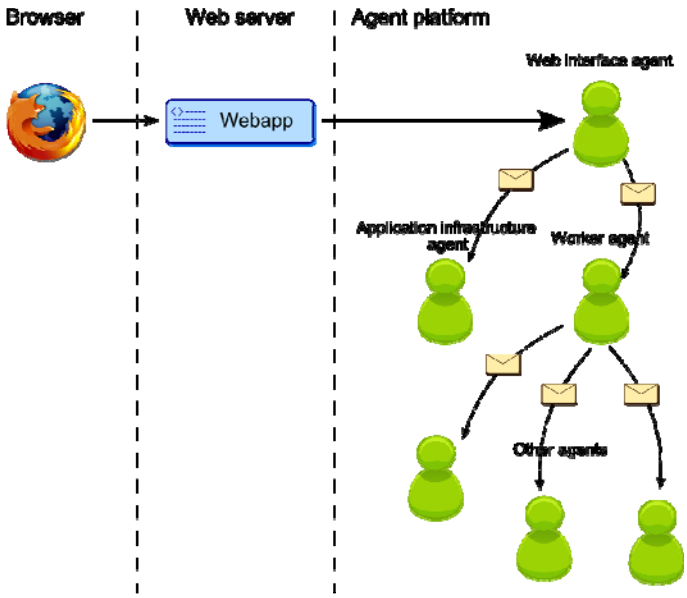


***Figure 14**- UBIWARE Web Application Architecture.*

One more improvement of the platform was UbiwareDesktop. Idea of it was borrowed from the web desktop environments. Web desktop is a desktop environment embedded in the browser. Web desktops like eyeOS (http://eyeos.org/) offer many of the functionalities and applications available on basic Windows, OSX or Linux desktop environments, such as productivity suites and file management. Some of the benefits of moving desktop to the web are high availability, server-side session management and centralized software management.

UbiwareDesktop is a web application that is distributed with the UBIWARE platform. In its current version the UI acts as simple launcher for other web applications deployed as part of the desktop environment. Applications can be opened as windows inside the desktop or in a new browser window. Figure 15 shows the desktop with two application windows open inside single browser window.

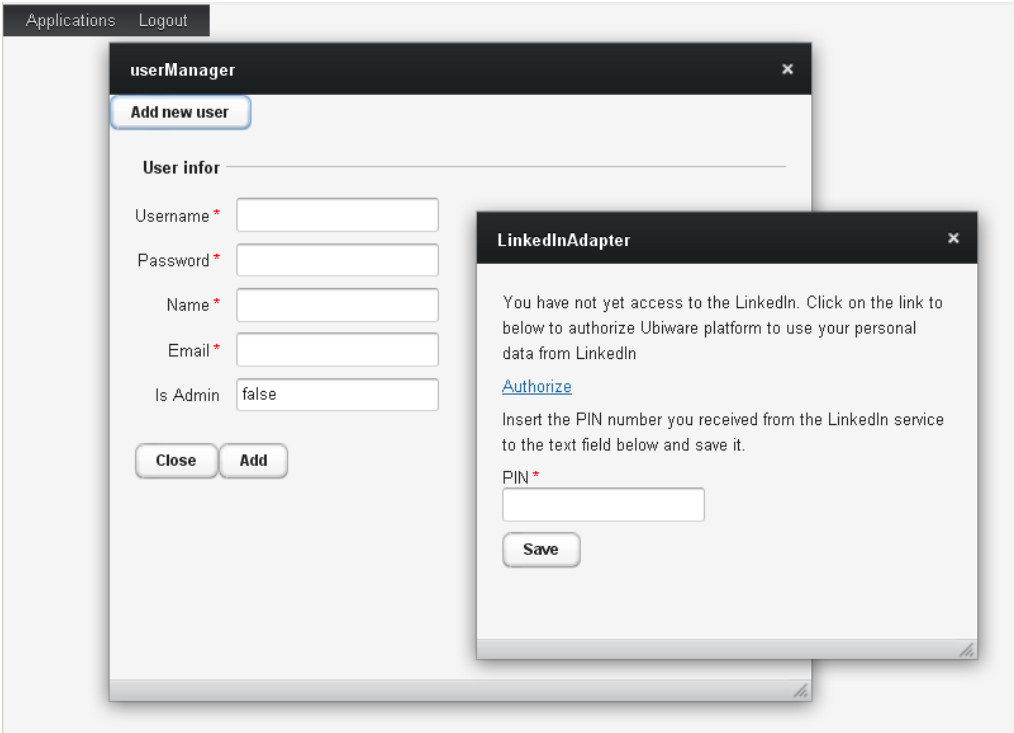


***Figure 15** – Ubiware Desktop with running applications.*

Currently the main benefit of using UbiwareDesktop as the deployment target for applications is the user management and authentication services provided by the desktop. As in any multi-user desktop environment, UbiwareDesktop requires users to login. After the user has successfully logged in, the UbiwareDesktop creates a ticket for the session, which is used to authorize the subsequent requests. Ticket can be used as kind of single-sign-on system, since other web applications can use the same ticket as a way to authenticate users. Users can be managed using another web application that is automatically available for all the administrative users.

In the future releases of the UBIWARE platform, desktop is envisioned to facilitate semantic drag-and-drop between applications. In order to make that possible without browser plugins, UbiwareDesktop could work as an intelligent, agent-driven mediator between source and target applications.

The core platform improvements are one of the most significant ones in the platform, although they are hardly visible or even presentable. In the 3.0 version we have resolved a lot of performance critical bottlenecks and have made several extensions that facilitate the application development on the platform.

Use Case: Mashupper – Agent-enabled Social Web:

The Mashupper application uses data from three prominent social network platforms: Facebook, LinkedIn and Twitter. As the largest player in the field with more than 500 million users, Facebook is becoming a virtual world of its own. It fulfills the need for general socializing in the web and covers both leisure and work. LinkedIn on the other hand is very much profiled to the business and professional side of the social networking. Then there is the Twitter, which has capitalized on the people's need to hear and to be heard, preferably in real-time. The social connections in Twitter are looser than in LinkedIn or Facebook. Everything you write to Twitter (or tweet) is public and can be read by anyone. In Twitter, user can start to follow his or her friends or actually anyone who seems interesting enough, in order to receive updates from those people in real-time.

Facebook, LinkedIn and Twitter is a good set of services to start with, but there are many other interesting social networking sites out there. Adding new information sources to Mashupper is relatively simple task, thanks to the agent-based architecture of UBIWARE platform, given of course that the social network service provides some kind of API for external services. The scope of the Mashupper is built on top of the concept of Personal User Network (PUN), which is defined as the combination of different kinds of human connections, which a particular user may have in the social web. PUN is used to link the different online identities or profiles into one and same connection, through which the user can observe the integrated presence of that connection in the current web2.0 landscape.

In order to make a successful UBIWARE-driven application we start with the common domain model construction and build a Social Ontlogy.

Mashupper is designed to run under UbiwareDesktop. It means that every user receives four worker agents, each responsible for one of the web applications. Normal infrastructure agents are naturally also running. Figure 2.2 shows all the agents involved and their roles.

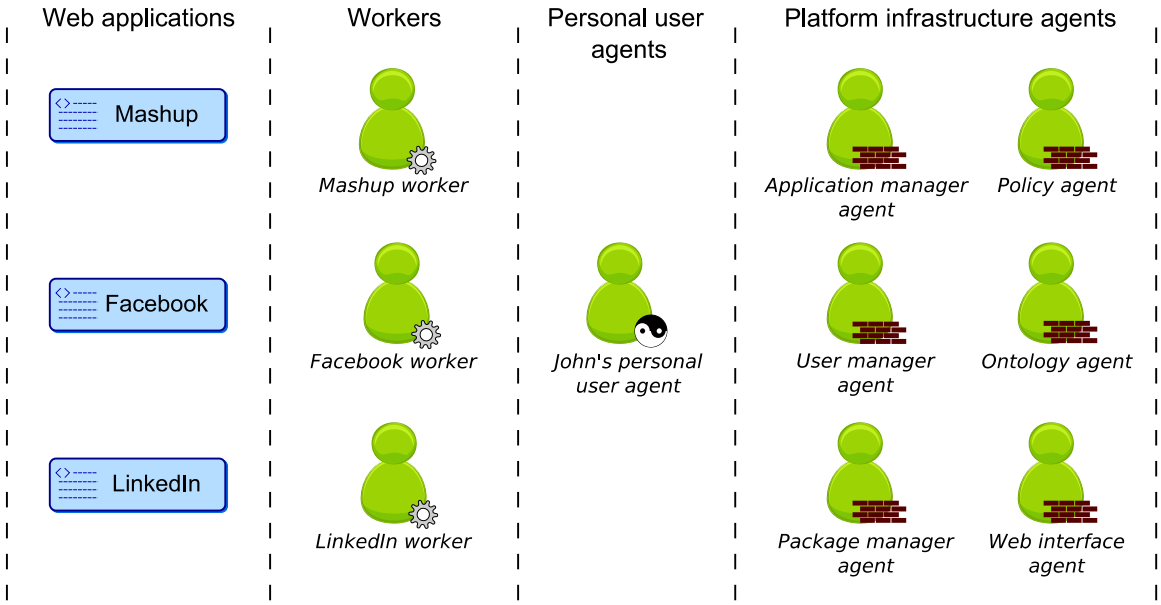


Figure 16*- Mashupper Application Agents.*

By developing the Mashupper application we tried to demonstrate the applicability of the UBIWARE 3.0 platform to the variety of tasks, not necessarily related to the heavy industry domain. The easy implementation of the Social Networking demo has proven that UBIWARE is genuinely middleware solution – with globally broad scope of application areas. Whatever we develop within the platform, we can reuse further in absolutely different application cases and scenarios. The social networking blocks and components are now available as reusable parts for any new application case.

General RDF adapter for 4I Browser:

According to the vision presented in Deliverable D3.1 this year, source data adaptation is one of the challenges that we have to solve to make 4I Browser more or less working system. 4I Browser is kind of engine that provide context-sensitive visualization of resources via MetaProviders, it provides interoperability between different resources and services and adds some additional functionality. Thus, repository of resource descriptions is an input data for the Browser. To make Browser able to work with any external repository, we have to elaborate general adapter that enable to convert data from any format to the required one. New data formats appear all the time and will require new adaptation modules. Thus, optimal way for such module elaboration is to make it extendible, be able to add new adaptation sub-module for new data format transformation. Regarding to the 4I (FOR EYE) Browser extension and with a purpose to allow user to import any repository (in RDF or N-triple format) him(she)self, we elaborated general adapter. Adapter is supplied with a graphical interface that helps user to configure adapter with a respect to supported data fields of internal format to be browsed through 4I Browser in the context of close/similar resources.

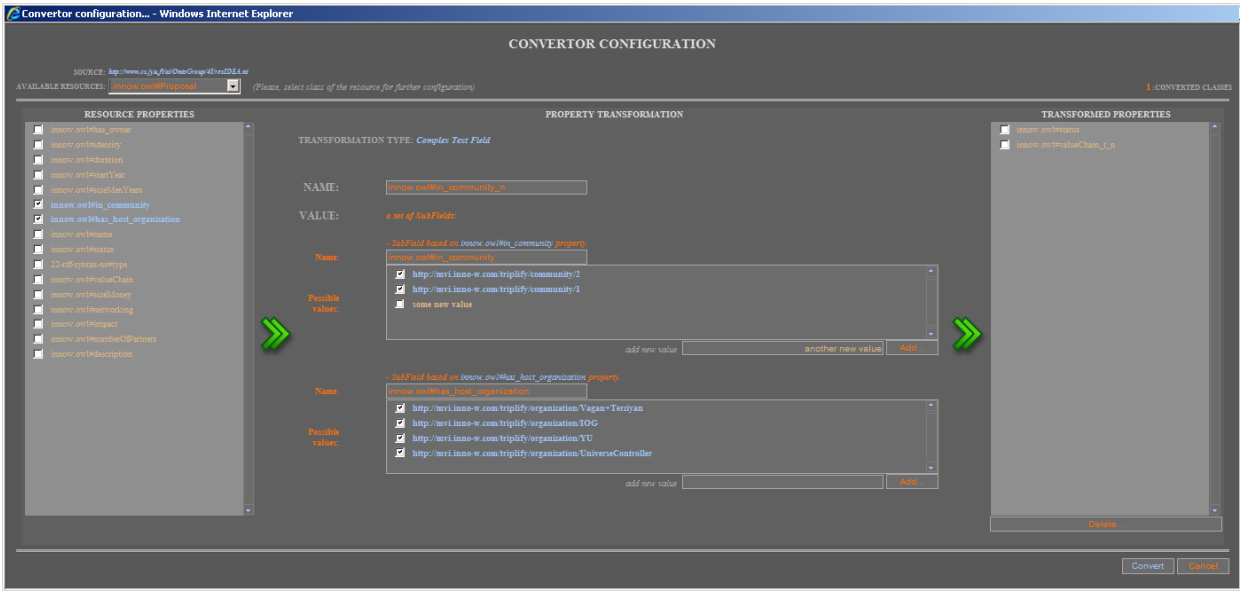


Figure 17 *– Configuration GUI of the adapter.*

3.4 UBIWARE Platform Prototype v.3.1 – Deliverable 3.4

UBIWARE deliverable D3.3 presented the integrated development results from all the work packages, i.e. the current state of the UBIWARE 3.0 platform prototype. Current deliverable D3.4 follows up and brings a platform update – UBIWARE 3.1. New version of the platform went through significant changes. The most important of them is the new architecture that follows cloud computing paradigm. Other changes include performance improvements and new features. This report can be understood as a "changelog" between platform version 3.0 and 3.1.

New version of the platform underwent several changes. Firstly, we bring a new agent classification system that simplifies the management and policy control of agents (see Figure 18). In the new version of the platform there are four types (classes) of agents. Each agent class is being started differently, has different rights (controlled by policies) and different tasks associated with it.

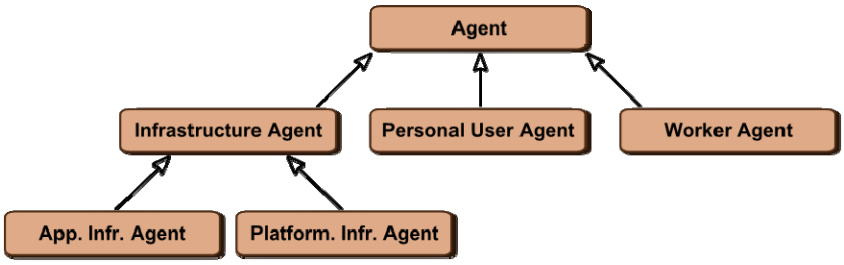


Figure 18 *- Agent classification schema*

Secondly, the new agent classification enabled new platform architecture. The platform infrastructure reported in deliverable D3.3 has been extended and improved. Now, the platform contains seven platform infrastructure agents, each supporting one type of platform functionality. Already existing platform infrastructure agents were modified as well. The platform supports application deployment in form of UBI packages through Package manager agent.

Moreover, we developed two new web applications supporting the platform administration (see Figures 19 and 20). First application is used for user management and the second one is used for management of UBIWARE applications.

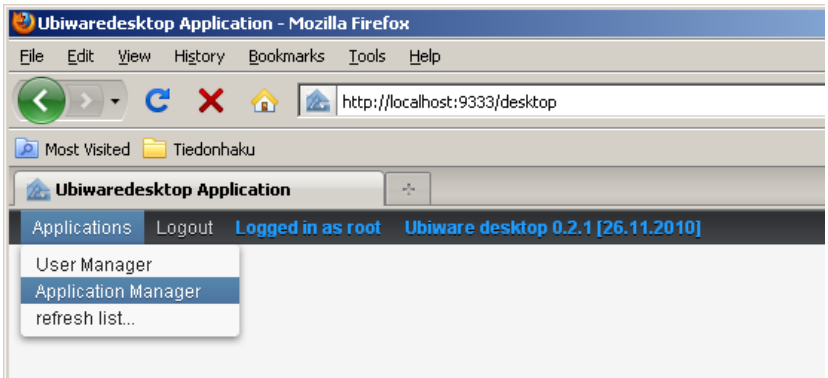


Figure 19 *- Ubiware desktop web application*

User management agent is responsible for user registration. Similar to desktop operating systems, every user has an account, his/her own list of applications and data. Users access the platform through Desktop application which follows the idea of web-based operating systems. User manager agent can be accessed through “User management” application which is available only to the administrator.

Application manager agent is responsible for application selection and deselection. Every user has access to “Application manager” web application where he/she can customize the list of available applications. Last of significant platform infrastructure agent changes includes a completely new directory facilitator called Ubiware Directory Facilitator. The underlying philosophy has changed accordingly.

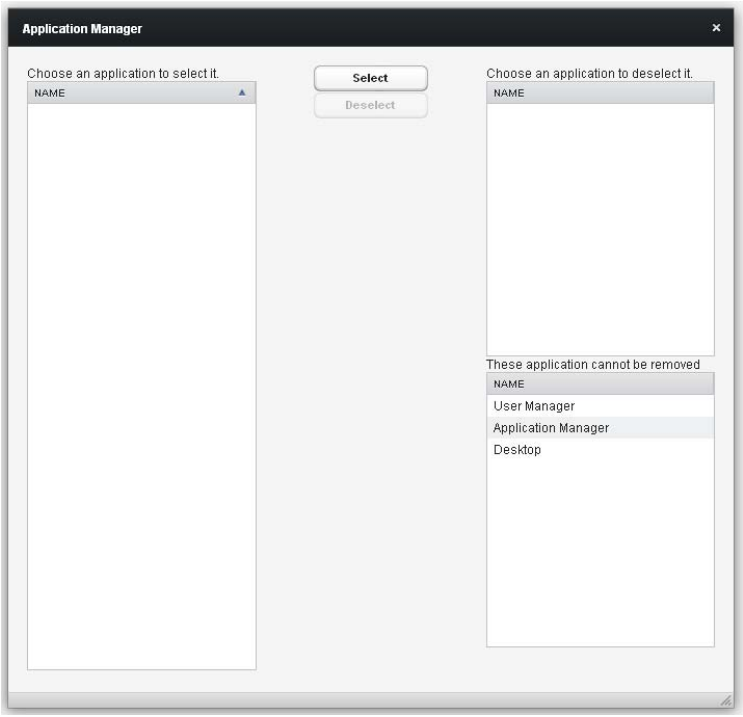


Figure 20 *- Application manager agent*

The rest of the platform infrastructure agents were changed as well to form a coherent platform. And finally, we create several new Reusable Atomic Behaviors (RABs) and several libraries that simplify UBIWARE application development.

4 Achievements of the year

4.1 International and local cooperation

As a representative of the University of Jyväskylä in TIVIT Cloud Software program (WP1: Technologies in the Cloud), IOG perform the tasks that are aimed at integration of Soprano RDF framework with UBIWARE platform. In this project the main work is done for the Nokia and we the main cooperation is done with Nomovok (Subcontractor of Nokia).

The cloud is the future of software systems. Therefore, the main objective of WP1 is to address the technologies for developing software-intensive applications for the cloud, and on the cloud. This requires a deep understanding of the nature of applications as well as the new business models enabled by the cloud. Thus, we are planning to continue this work also next year and show more benefits of UBIWARE platform for the tasks related to Cloud-based ecosystem development.

Another project that we are cooperating with is SCientific innOvation Product concEpt (SCOPE, 2009-2012) project run on the faculty of Information Technologies at the University of Jyväskylä. This project is supported by Nokia, EADS, NeedIt and other companies. Project is aimed to several aspects of mobile environment enhancements and new mobile application. Currently we are cooperation on one of them that concerns "Tactical platform to enhance public safety by situational awareness" (WP4). Within this cooperation we provide the information source integration and information matching solutions based on UBIWARE Platform.

We cooperate with Techila Company that provides cloud infrastructure solutions. The company has an agreement with a faculty of Information Technologies at the University of Jyväskylä and could provide a missing part (Infrastructure as a Service) for the Cloud Architecture based on UBIWARE Platform. We are preparing the project plan for Cloud Software program.

We started cooperation with the companies in the area of forest industry. We have prepared SOFIA (Seamless Operation of Forest Industry Applications) project proposal. The project aimed to bring together new ICT approaches and technologies (Semantic Web, Autonomic Computing, and Internet of Things) which are amongst mainstream trends in information integration and engineering of complex systems to boost adoption of new business models in forest industry. The practical expected outcome of the project was a SOFIA software platform for B2B mediation tailored to the forestry sector of Finland. We planned to have a strong national level network of forestry companies and organizations in order to get support and promote the new vision of flexible contracting for SMEs. Preliminarily we have discussed and received positive feedback from: organizations - Koneyrittäjät ry, Metsäalan kuljetusyrittäjät ry; forest industry companies and universities – Metsäteho, Metla, Joensuun yliopisto, VTT, Metsäkeskus Tapio, Metsäteollisuus ry, Energiateollisuus ry.

On international level, as a representative from University of Jyväskylä (Finland), together with IT'IS Foundation (Switzerland), TECHILA (Finland), SPEAG AG (Switzerland), University of Freiburg (Germany), EADS (Germany) IOG is going to participate in "i-Cloud" EU project, FP7/ICT – Cloud Computing.

IOG has been collaborating with finish academic partners such as University of Helsinki and Tampere University of Technology. Together with these partners we submitted "cCloud: Collaborative clouds" project proposal to Academy of Finland as a result of this collaboration. The cCloud project objective is to develop technologies and models for creation of open marketplaces and methodologies for the provision of service-ecosystem infrastructure elements. Such infrastructures provide service matching, eContracting, and ecosystem evolution services that can be utilized in coherent, SOA-based, and potentially agent-driven ways. Modules for automated, context- and content-sensitive interoperability solutions form part of this frame. As a basement for such collaborative Ecosystem environment, we consider a network of the platforms

that provide such main functionalities as cross-platform communication, interoperability of heterogeneous resources (components) and a toolbox for component composition on both levels. Due to heterogeneity of provided services and supported components, UBIWARE is based on integration of several technologies: Semantic Web, Distributed Artificial Intelligence and Agent Technologies, Ubiquitous Computing, SOA (Service-Oriented Architecture), Web X.0, and related concepts. Thus, we are going to use the platform as a basis and extend its functionality towards the needs of Cloud Ecosystem elaboration. We consider this cooperation as a great opportunity for further utilization of the project results and dissemination of them on a Finnish academic level.

4.2 Awarded degrees

During this year one of the IOG members *Sergiy Nikitin* have finished his Doctoral Thesis. The dissertation "Dynamic Aspects of Industrial Middleware Architectures" is going to be defended at its public examination in January 2011. Another team member *Viljo Pilli-Sihvola* will get his Master Degree in December 2010. The title of his thesis is “Intelligence as a Service”.

5 UBIWARE Project Publications (up to the end of December 2010)*

[1] Nikitin S., Dynamic Aspects of Industrial Middleware Architectures, In: Jyvaskyla Studies in Computing, PhD Thesis, Jyvaskyla University Printing House (will be published in January 2011).

[2] Pilli-Sihvola V., Intelligence as a Service, MSc. Thesis, Department of Mathematical Information Technology, University of Jyväskylä, December 2010.

[3] Nikitin S., Terziyan V., Nagy M., Mastering Intelligent Clouds: Engineering Intelligent Data Processing Services in the Cloud, In: Proceedings of the 7th International Conference on Informatics in Control, Automation and Robotics (ICINCO-2010), 15-18 June, 2010, Funchal, Madeira - Portugal, 8 pp.

[4] Khriyenko O., Nikitin S., Terziyan V., Context-Policy-Configuration: Paradigm of Intelligent Autonomous System Creation, In: Joaquim Filipe and Jose Cordeiro (Eds.), Proceedings of the 12th International Conference on Enterprise Information Systems (ICEIS-2010), 8-12 June, 2010, Funchal, Madeira - Portugal, ISBN: 978-989-8425-05-8, pp. 198-205.

[5] Nikitin S., Terziyan V., Lappalainen M., SOFIA: Agent Scenario for Forest Industry, In: Proceedings of the 12th International Conference on Enterprise Information Systems (ICEIS-2010), 8-12 June, 2010, Funchal, Madeira - Portugal, 8 pp.

[6] Terziyan V., Kaykova O., Zhovtobryukh D., UbiRoad: Semantic Middleware for Context-Aware Smart Road Environments, In: Proceedings of the Fifth International Conference on Internet and Web Applications and Services (ICIW-2010), May 9-15, 2010, Barcelona, Spain, IEEE CS Press, 8 pp.

* Papers are downloadable from *http://www.cs.jyu.fi/ai/OntoGroup/UBIWARE_details.htm*

[7] Katasonov A., Terziyan V., Using Semantic Technology to Enable Behavioural Coordination of Heterogeneous Systems, In: Gang Wu (ed.), Semantic Web, IN-TECH Publishing, Vienna, Austria, January 2010, ISBN 978-953-7619-54-1, pp. 135-156. (Chapter VIII).

[8] Nagy M., Katasonov A., Khriyenko O. Nikitin S., Szydlowski M., Terziyan V., Challenges of Middleware for the Internet of Things, In: A. Lazinica (ed.), Robotics, Automation and Control, IN-TECH Publishing, 2009, ISBN: 978-953-7619-39-8, 24 pp. (Book Chapter).

[9] Khriyenko O., Terziyan V., Similarity/Closeness-Based Resource Browser, In: Proceedings of the Ninth IASTED International Conference on Visualization, Imaging and Image Processing (VIIP-2009), July 13-15, 2009, Cambridge, UK, 7 pp.

[10] Kesäniemi J., Katasonov A., Terziyan V., An Observation Framework for Multi-Agent Systems, In: Proceedings of the Fifth International Conference on Autonomic and Autonomous Systems (ICAS 2009), April 21-25, 2009, Valencia, Spain, IEEE CS Press, 6 pp.

[11] Katasonov A., Terziyan V., Semantic Approach to Dynamic Coordination in Autonomous Systems, In: Proceedings of the Fifth International Conference on Autonomic and Autonomous Systems (ICAS 2009), April 21-25, 2009, Valencia, Spain, IEEE CS Press, 9 pp.

[12] Terziyan V., Zhovtobryukh D., Katasonov A., Proactive Future Internet: Smart Semantic Middleware for Overlay Architecture, In: Proceedings of the Fifth International Conference on Networking and Services (ICNS-2009), April 21-25, 2009, Valencia, Spain, IEEE CS Press, 6 pp.

[13] Nikitin S., Katasonov A., Terziyan V., Ontonuts: Reusable Semantic Components for Multi-Agent Systems, In: Proceedings of the Fifth International Conference on Autonomic and Autonomous Systems (ICAS 2009), April 21-25, 2009, Valencia, Spain, IEEE CS Press, 8 pp.

[14] Khriyenko O., Adaptive Semantic Web based Environment for Web Resources, In: Jyvaskyla Studies in Computing, PhD Thesis, Volume 97, Jyvaskyla University Printing House, 192 pp., December 13, 2008.

[15] Bleier A., A Framework for Market-Based Coordination in Multi-Agent Systems, MSc Thesis, University of Osnabrück, September 30, 2008.

[16] Terziyan V., Semantic Web Services for Smart Devices Based on Mobile Agents, In: D. Taniar (Ed.), Mobile Computing: Concepts, Methodologies, Tools, and Applications (6 volumes), IGI Global, November 2008, ISBN: 978-1-60566-054-7, Vol. II, Chapter 2.22, pp. 630-641.

[17] Terziyan V. and Katasonov A. (2008) Global Understanding Environment: Applying Semantic and Agent Technologies to Industrial Automation, In: Lytras, M. and Ordonez De Pablos, P. (eds) Emerging Topics and Technologies in Information Systems, IGI Global , 2009, ISBN: 978-1-60566-222-0, pp. 55-87 (Chapter III).

[18] Katasonov A. and Terziyan V. (2008) Semantic Agent Programming Language (S-APL): A Middleware Platform for the Semantic Web, In: Proc. 2nd IEEE Conference on Semantic Computing, August 4-7, 2008, Santa Clara, CA, USA, pp.504-511.

[19] Khriyenko O., Context-sensitive Visual Resource Browser, In: Proceedings of the IADIS International Conference on Computer Graphics and Visualization (CGV-2008), Amsterdam, The Netherlands, 24-26 July 2008.

[20] Katasonov A., Kaykova O., Khriyenko O., Nikitin S., Terziyan V., Smart Semantic Middleware for the Internet of Things, In: Proceedings of the 5-th International Conference on Informatics in Control, Automation and Robotics, 11-15 May, 2008, Funchal, Madeira, Portugal, ISBN: 978-989-8111-30-2, Volume ICSO, pp. 169-178.

[21] Terziyan V., SmartResource - Proactive Self-Maintained Resources in Semantic Web: Lessons learned, In: International Journal of Smart Home, Special Issue on Future Generation Smart Space, Vol.2, No. 2, April 2008, SERSC Publisher, ISSN: 1975-4094, pp. 33-57.

[22] Katasonov A. and Terziyan V. (2007) SmartResource Platform and Semantic Agent Programming Language (S-APL), In: Proceedings of the 5th Conference on Multi-Agent Technologies (MATES'07), September 24-26, 2007, Leipzig, Germany, LNAI 4687, pp.25-36.

[23] Terziyan V., Predictive and Contextual Feature Separation for Bayesian Metanetworks, In: B. Apolloni et al. (Eds.), Proceedings of KES-2007 / WIRN-2007, Vietri sul Mare, Italy, September 12-14, Vol. III, Springer, LNAI 4694, 2007, pp. 634–644.

[24] Khriyenko O., Context-sensitive Multidimensional Resource Visualization, In: Proceedings of the 7th IASTED International Conference on Visualization, Imaging, and Image Processing (VIIP 2007), Palma de Mallorca, Spain, 29-31 August 2007.

[25] Khriyenko O., 4I (FOR EYE) Multimedia: Intelligent semantically enhanced and context-aware multimedia browsing, In: Proceedings of the International Conference on Signal Processing and Multimedia Applications (SIGMAP-2007), Barcelona, Spain, 28-31 July 2007.

[26] Khriyenko O., 4I (FOR EYE) Technology: Intelligent Interface for Integrated Information, In: Proceedings of the 9th International Conference on Enterprise Information Systems (ICEIS-2007), Funchal, Madeira – Portugal, 12-16 June 2007.

[27] Salmenjoki K., Tsaruk Y., Terziyan V., Viitala M., Agent-Based Approach for Electricity Distribution Systems, In: Proceedings of the 9-th International Conference on Enterprise Information Systems, 12-16, June 2007, Funchal, Madeira, Portugal, ISBN: 978-972-8865-89-4, pp. 382-389.

[28] Nikitin S., Terziyan V., Pyotsia J., Data Integration Solution for Paper Industry - A Semantic Storing, Browsing and Annotation Mechanism for Online Fault Data, In: Proceedings of the 4th International Conference on Informatics in Control, Automation and Robotics (ICINCO), May 9-12, 2007, Angers, France, INSTICC Press, ISBN: 978-972-8865-87-0, pp. 191-194.

[29] Naumenko, A., Srirama, S., Secure Communication and Access Control for Mobile Web Service Provisioning, In: Proceedings of International Conference on Security of Information and Networks (SIN2007), 8-10th May, 2007.

[30] Naumenko A., Semantics-Based Access Control in Business Networks, In: Jyvaskyla Studies in Computing, PhD Thesis, Volume 78, Jyvaskyla University Printing House, 215 pp., June 28, 2007.

[31] Naumenko A., Katasonov A., Terziyan V., A Security Framework for Smart Ubiquitous Industrial Resources, In: R. Gonzalves, J.P. Muller, K. Mertins and M. Zelm (Eds.), In: Enterprise Interoperability II: New challenges and Approaches, Proceedings of the 3rd International Conference on Interoperability for Enterprise Software and Applications (IESA-07), March 28-30, 2007, Madeira Island, Portugal, Springer, pp. 183-194.

[32] Naumenko A., SEMANTICS-BASED ACCESS CONTROL - Ontologies and Feasibility Study of Policy Enforcement Function, In: Proceedings of the 3rd International Conference on Web Information Systems and Technologies (WEBIST-07), Barcelona, Spain - March 3-6, 2007, Volume Internet Technologies, INSTICC Press, pp. 150-155.

6 References

Kaykova O., Khriyenko O., Kovtun D., Naumenko A., Terziyan V., and Zharko A. (2005a) General Adaption Framework: Enabling Interoperability for Industrial Web Resources, In: International Journal on Semantic Web and Information Systems, Idea Group, Vol. 1, No. 3, pp.31-63.

Kaykova O., Khriyenko O., Naumenko A., Terziyan V., and Zharko A. (2005b), RSCDF: A Dynamic and Context Sensitive Metadata Description Framework for Industrial Resources, In: Eastern-European Journal of Enterprise Technologies, Vol.3, No.2, pp. 55-78.

Kaykova O., Khriyenko O., Terziyan V., and Zharko A. (2005c) RGBDF: Resource Goal and Behaviour Description Framework, Proc. 1st International Conference on Industrial Applications of Semantic Web, Springer, IFIP, vol.188, pp. 83-99.

Khriyenko O., and Terziyan V. (2006), A Framework for Context-Sensitive Metadata Description, In: International Journal of Metadata, Semantics and Ontologies.

Kephart J. O. and Chess D. M. (2003), The vision of autonomic computing, IEEE Computer, Vol. 36, No. 1, pp. 41-50

Terziyan V. (2003) Semantic Web Services for Smart Devices in a "Global Understanding Environment", In: R. Meersman and Z. Tari (eds.), On the Move to Meaningful Internet Systems 2003: OTM 2003 Workshops, Lecture Notes in Computer Science, Vol. 2889, Springer-Verlag, pp.279-291.

Terziyan V. (2005) Semantic Web Services for Smart Devices Based on Mobile Agents, In: International Journal of Intelligent Information Technologies, Vol. 1, No. 2, Idea Group, pp. 43-55.