Considering Resource Management in Agent-Based Virtual Organization*

Grzegorz Frąckowiak[1], Maria Ganzha[1], Maciej Gawinecki[1], Marcin Paprzycki[1], Michał Szymczak[1], Myon Woong Park[2], and Yo-Sub Han[2]

[1] Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland
{maria.ganzha,marcin.paprzycki}@ibspan.waw.pl
[2] Korea Institute of Science and Technology, Seoul, Korea
{myon, emmous}@kist.re.kr

Abstract. In this chapter we discuss a system designed to support workers in a virtual organization. The proposed approach is based on utilization of software agents and ontologies. In the system all *Users* are represented by their *Personal Agents* that help them in fulfilling their specific roles. At the same time all entities that the organization is comprised off (human and non-human) are represented as instances of resources in an ontology of the organization. Furthermore, each resource is associated with one or more profiles and these profiles are adapted to represent changes in resources (e.g. new experience/knowledge gained by a human resource, or approval of a duty trip application). The aim of this chapter is to describe basic functions of our system with special attention paid to software agents, their roles and interactions as well as utilization of ontologies in support of worker needs.

1 Introduction

Let us consider an organization in which teams of researchers are engaged in R&D projects and share a common virtual work-space (regardless if they are geographically distributed or not). Obviously, team work requires cooperation between members and support of collaborative research has to go beyond, even most sophisticated forms of, document versioning and flow of resources in the hierarchical structure of the organization. What needs to be taken into account is: (1) representation of domain specific knowledge (e.g. geological sciences); to provide context for management of resources pertinent to running projects (e.g. establishing a specific "location" of a resource within the domain knowledge allows for resource indexing, clustering; it also allows to establish *who* within the organization should receive a notification that a new resource—such as a book—has been acquired); (2) representation of structure of interactions and flow of resources in the project (and, more generally, within the organization); to route resources, based on project needs and responsibilities of team members

*Work was partially sponsored by the KIST-SRI PAS "Agent Technology for Adaptive Information Provisioning" grant.

(e.g. who should receive a report that a given task is completed, or to whom an application for a business trip should be routed); (3) representation of user profiles (situated within the domain knowledge and the structure of the project); to specify *interests*, *needs* and *skills* of individual workers (e.g. to establish who needs to be proactively trained in view of an upcoming project); (4) adaptability of the system; to deal with the fact that as the time passes the scope of the project may expand, contract or shift; functional interrelationships between team members (or within the whole organization) can change; their interests, needs and skills evolve; and, team members may be added, removed or replaced.

It is relatively easy to see that these four points can be generalized beyond the initial collaborative research scenario. Let us assume that for the second point we utilize a notion of a virtual organization (VO) [16–21], which allows us to define roles, interdependencies and interactions of participants. Here, it is important to note that while most conceptualizations of a VO stress the importance of its workers being spatially distributed, in the proposed approach "virtualization" involves realization of an actual organization as an e-organization. Therefore, it does not matter if the organization itself is actually geographically distributed or not. In such an organization its members need access to resources to complete their individual tasks and to facilitate completion of projects. In our approach, any entity within the organization, human and non-human, is considered to be a resource. Obviously, access of resources to resources should be, among others, adaptive (change with the task) and personalized (each team member—human resource—requires access to different resources; furthermore access is likely to be restricted by the organization policy/structure). The aim of our work is to develop a software infrastructure for such a virtual organization. The basic assumption underlying our approach is that emergent technologies such as software agents [42] and ontologies [36] should be utilized as a foundation around which the proposed system should be conceptualized. Let us stress that we do realize that these assumptions are not uncontroversial. However, our aim is to develop a system on their basis, and *in this way* to add to the discussion of viability of this approach (instead of getting involved in theoretical discussions). This being the case we assume that: (i) the organizational structure, consisting of "roles" played by various entities within the organization and interactions between them, should be represented by software agents and their interactions (i.e. the complete structure of an actual organization is mapped into the structure of an agent-based virtual organization), and (ii) domain knowledge, organization structure, resource profiles and resource matching should be based on ontologies and reasoning machinery associated with them. For instance, in a company that installs and services satellite TV antennas, the domain specific knowledge consists of a complete body of knowledge concerning such antennas. The structure of the company involves, among others, antenna installing teams, their equipment, the way that work orders are delivered to them, and the reporting upon task completion. Software agents represent each worker and support them in completing assigned tasks (e.g. managing a team of installers). Finally, human resource profiles describe skills of each member of service team, while the adapt-

ability involves situation when a new antenna is to be introduced to the market and installation crews have to be trained in its features.

The aim of this chapter is to summarize main results obtained thus far within the project and is based on [37, 8, 9, 38]. To this effect we proceed as follows. In the next section we present a general description of the the system. Then, we discuss the issues concerning interactions between software agents of human workers. Following the discussion concerning agents in the system, we concentrate our attention on ontologies and ontological demarcation of resources. We start with the generic ontology of the virtual organization, and follow with description of its extensions to the areas facilitated by applications proposed by an Institute of Science and Technology. Finally, we discuss processes involved in ontological matchmaking proposed in the system.

2 System Overview—Introducing Project Into the System

Before discussing the main features of the system let us first stress that in the proposed approach each worker in the organization is represented by her/his *Personal Agent* (*PA*). This agent plays two roles: (a) it is the interface between the *User* and the system, and (b) it supports its owner in all *roles* that (s)he is to play within the organization. Let us now present birds-eye view of the system, by discussing processes involved in introducing and running a project. To focus our discussion, in Figure 1 we present the use case diagram of the system. Note that the following discussion is written in terms of *entities with specific roles*, and such units can consist of one (or more) humans, agents, or "teams" consisting of humans and agents. We will return to the issue of interactions between humans and agents later in the chapter.

When a service/project is requested from an organization (which can be anything from a one-person business to a 50,000+ employees corporation) a *Project Manager* (*PM*) is associated with it. The *PM* is a *role* that is associated, for instance, with a person who in the *VO* is represented by the *Personal Agent*, which will support that person in fulfilling the role of the *PM*. *PM*s first task is to make sure that the request is thoroughly analyzed and on the basis of such analysis to make a decision if the job should be accepted. This task is delegated to the *Analysis Manager* (*AM*). At the same time a *Task Monitor Agent* (*TMA*) is created to oversee the task performed by the *AM* (for more details about role of the *TMA*, see below). It should be noted that the structure of the *AM* can be either very complicated and consist of a number of humans and agents (e.g. in the case of a corporation that is evaluating a multi-million euro construction project) or very simple (e.g. in case of a small business assessing acceptance of a brake pads replacement job). Finally, it is even possible that the *PM* can play the role of the *AM* (e.g. in the case of a very small business or self-employment). Regardless of the specific situation, the most important deliverable prepared by the *AM* is a set of reports that support the decision to accept or reject the

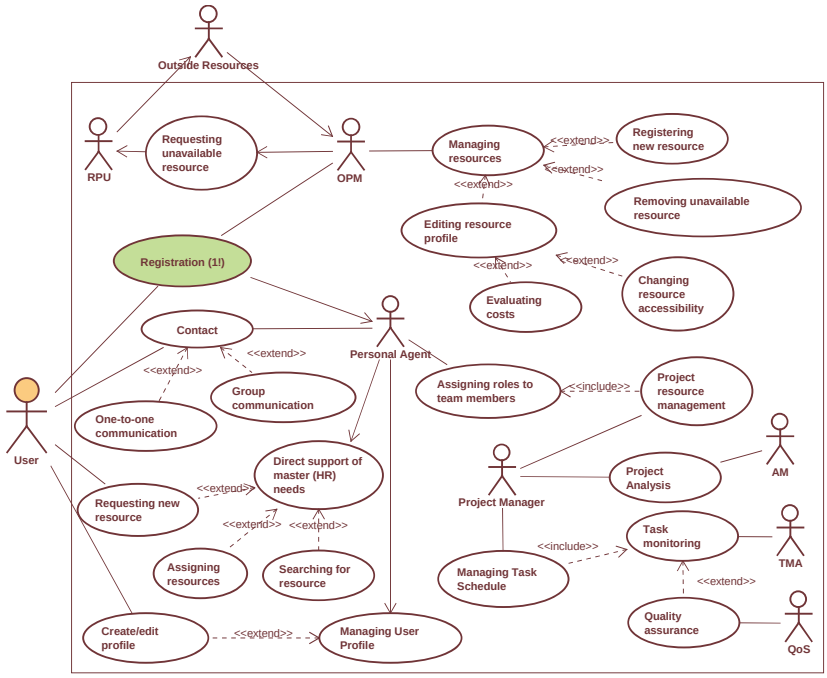

Fig. 1. Use case of the system

requested project. The report(s) prepared by the *AM* is(are) backed up, among others, by the cost, resource and income analysis.

Since we assume that data processed in the system is based on appropriate (domain and organization) ontologies, one of crucial tasks of the *AM* is to "translate" the common language requirements originating from the user (project proposer) into a set of requirements specified utilizing ontologies employed in a given organization. To fulfill this need, at the beginning of its work, the *AM* instantiates a new resource called the *Project Request* (which has its own profile). This resource is used when the *AM* performs initial analysis of the proposed task and creates the first version of the *System Requirements Specification* (*SRS*) which, again, is a resource with its own profile. During its work the *AM*, among others, analyzes resources available in the organization (their profiles, availability and accessibility). For instance, in the case of a cable TV installation job, this step is going to be rather simple and involves steps like: (1) checking whether the customer who requests installation lives in the service-covered area, and (2) if there are resources available to install the cable TV at her address, within a specific time-frame. Note that such simple analysis could easily be performed by a software agent with a build-in expert system. On the other hand in the scenario of the project involving development of an intranet and a knowledge portal for a company, analysis would involve more elaborate actions, such as: checking tech-

nological requirements for the project, existing similar solutions, organization or customers experience etc. It can be conjectured that in this case the *AM* would most likely involve human resources as well as software agents helping them completing the tasks. Let us mention, that in the case of a large corporation the *AM* may not have permission to know about all resources available in the organization. In this case, the *AM* will specify resources that the project needs and which, according to its best judgment, are unavailable. It will be then the role of the *PM* (and possibly its supervisors in the organization) to assess if the resources are actually available, or if they need to be found outside of the organization, and what is the effect of such search on the viability of project acceptance. Note that this process may involve interaction with the *Organization Provisioning Manager*, which is aware of all resources available in the organization (see below).

If the *AM* recommends that the project is rejected, and the *PM* concurs (which may, or may not be the case; it is a well-known fact that there exist projects which are accepted even if they should not have been accepted, for instance for "political" reasons), the requester is informed about the decision, the *PM* is disassociated from the role, and this ends the process. Let us now assume that the *SRS* and other supporting documentation prepared by the *AM* suggests that the project should be accepted. As a result the *PM* prepares an initial *Project Schedule* and on its basis works to establish if the *Resource Reservation* can be completed (note that the fact that John, the Java coder, works for the organization does not mean that John is available starting from May 4th). To achieve this goal, the *PM* has to analyze available resources (its own and provided by the customer). It may involve, for instance, checking availability of programmers who have the required competence in PostgreSQL, object oriented programming and recent web technologies, as well as availability of resources such as: servers, (e-)learning materials for software to be used in the project, licenses and requirements for both test and final deployment environments etc. Again, the *PM* can analyze only these resources which it has access to (is allowed to know about, as established by the ontology of the organization). If resources that the *PM* knows about are not sufficient, the *PM* requests the *Organization Provisioning Manager* (*OPM*) to facilitate the missing resources (e.g. C# programmer(s), or a DB2 e-learning course). Note that such resources may be available in the organization, but the *PM* may not have access to this knowledge. *OPM*'s role is to provide resources for other resources which request them, as well as to deal with resources that are being delivered to the organization (e.g. books/papers/reports send to its digital library). Here, we assume that, to fulfill its role, the *OPM* has access to information about all resources available in the organization. Since the *OPM* can be queried by authorized (where the authorization is also ontologically specified) resources that play various roles in the organization, it has to analyze available resources using various patterns of reasoning and possibly some expert systems. Note also that, again, the *OPM* can be either an agent, a human represented by its *PA*, or a composite structure consisting of multiple agents and humans (e.g. it can have in its disposal a resource that indexes and routes incoming documents / books / journals, a search engine,

a library material acquisitor, etc.). Again, if the resource (a) is found, and (b) can be reserved (for a specified time), it can then be assigned to the requesting *PM*. Otherwise, the *OPM* triggers action of a *Resource Procurement Unit* (*RPU*), which is responsible for finding an appropriate resource. Assuming, for instance, that C# programmers and DB2 e-learning materials were not found within the organization the *OPM* may generate a (ontologically demarcated) request to the *RPU* to acquire specific resources. The *RPU* in turn will communicate it to the "world outside of the organization." For simplicity we omit situations which clearly have to involve human intervention. Let us assume that company needs construction workers to start a project in Lublin, Poland. If it does not employ large enough number of such workers it, most likely, will be the role of human managers to assess if they can be hired for the time of the project. Therefore, at this stage, we assume that the *RPU* can immediately provide information about availability and cost (estimate) of requested resources. As a result of these processes two outcomes are possible. First, it is established, that the initial *Project Schedule* cannot be supported with necessary resources (which may result in project schedule (re)negotiation(s) with the client, or project rejection). Second, the *Project Schedule* can be completed (with possible minor modifications) in such a way that the project can be accepted and a contract signed.

Let us now discuss processes that take place after the final version of the *Project Schedule* is created and contract signed. First, the *Project Schedule* is used by the *PM* to assign tasks to appropriately reserved (human or non-human) *Resources*. Note that each *Resource* can be either a "single resource," or a collection of resources treated as a single unit. For instance, team that is responsible for the back end of the portal may consist of 4 coders and a manager, while the team dealing with user interface could consist of 2 coders and an artist, etc. In the hierarchical structure of the organization, at one level, both teams will be treated as a single resource, with their own tasks, and a *Task Monitor Agent* (*TMA*) associated with it. At the same time, inside these composite resources an appropriate organizational structure (based on the same ontology of the organization) will be realized, and individual (sub)tasks and their *TMA*s instantiated.

The *PM* monitors status of all tasks (including their start and completion) by assigning to each task a *TMA* and by communicating with them. Each *TMA* monitors a specific task until its completion (then it is killed by the *PM*; currently, we assume that creation of a new *TMA* is easier to achieve than adapting a given *TMA* to manage a different task). While working on the task, *Resources* might be interrupted by unexpected circumstances which either can be dealt with "locally" (e.g. by finding tips on how to deal with a "heap memory exceeded" error in Java, or how to build a DB2 cluster) or ones that will probably influence other parts of the project (e.g. customer requested that a different data structure is to be interfaced with, or some additional unavailable resources turn out to be needed, or a particular employee has to immediately take a family leave of absence, etc.). These circumstances are expected to involve *PM*'s reaction and should be tagged appropriately by the *Task Monitor Agent*. Let us stress that not

every interruption requires an immediate *PM*'s intervention. Across the system we assume that resources can interact with each other (which resources can communicate directly is specified by the organization and represented in the organizational ontology), among others, to solve basic problems occurring during task execution. For instance, to find a manual for software used in the project given resource can contact other members of its group. Finally, each resource might generate multiple interrupts, but as long as these do not require the *PM* to react (e.g. the schedule of the project is not affected) they are going to be tackled locally.

Obviously, at a certain moment each (sub)task comes to an end. Upon completion of a task, the task-specific *Quality of Service* (*QoS*) module analyzes the work. A *Os* module might consists of a team of humans, or be instantiated as an expert system. Here, consider testing functionality of the company portal, or a test of a completed unit of a Python code, or checking quality of the TV signal after the TV is installed. Each of these quality tests requires different testing and different resources to complete the quality assessment. Unless the quality of the work is not satisfactory and further improvements are needed, the *PM* is informed about completion of the (sub)task. If the result of the task does not satisfy the requirements, there is a necessity to repeat some part of, or even the whole task. This can take more time and resources than it was specified in the *Project Schedule*. However, only conflicts with the schedule should result in the *PM* being "alarmed." Note that a "major interrupt" that results in changes in the *Project Schedule* may need to be propagated within the structure of the team that works on the project.

Obviously, completion of a (sub)task may trigger execution of another (sub)task specified in the workflow of a given project. Upon completion of all (sub)tasks specified in the *Project Schedule*, the project is completed. This means that the *Human Resource* that played the role of the *Project Manager*, will no longer play this role (for *that* project) and the functionality of its *Personal Agent* has to be appropriately adjusted. Similarly, functionalities and profiles of *all* agents involved in the project have to be adjusted (e.g. experience-related information).

3 Agents in the system

Thus far we have described processes that take place within the Virtual Organization, considered from the point of view of "roles" existing in the system and their interactions. In this context let us recall, that one of our assumptions is that each *Worker* will be represented by a *Personal Agent*, while a number of auxiliary agents may be instantiated as well. In this way, the proposed approach is grounded not only in general agent notions (see, for instance, [26]), but also in role-oriented agent system development methodologies (e.g. Gaia [43], or Prometheus [33]). Here, the problem space is initially defined in terms of (1) roles that are to be fulfilled, and (2) interactions between entities playing these roles. In the second step each identified role is functionalized by a single agent, or is further divided into a number of cooperating (sub)agents (see, also, [25]). However, we are well aware of the fact that not all roles can be fulfilled by soft-

ware agents alone. Therefore, let us consider roles that have been distinguished thus far: *PM*, *AM*, *RPU*, *OPM TMA*, and *QoS*. As noted above, in some cases these roles may be fulfilled by software agent(s), some of them are likely to be played by one or more humans (supported by their *Personal Agents*), while some are likely to be completed by teams consisting of software agents and humans. Note that while specific arrangements may depend on the particular organization (and its domain of operation), processes described above remain unchanged. In this context we have identified a few situations that are expected to trigger a necessary reaction of a human actor (however, this list is not exhaustive):

1. project requirements analysis
2. accepting a particular person to become a manager of a project
3. changes in customer requirements
4. the *OPM* not being capable of finding required resource(s) within the organization
5. negotiating and accepting the *Project Schedule*
6. accepting the *Resource Reservation* document
7. final task acceptance

Even though human intervention is likely to be required, it has to be stressed that our interest is in performing as many tasks as possible utilizing software agents alone and thus completing them in an autonomous fashion or to provide support for humans in fulfilling the above specified roles. In this context, the role-based approach allows us to specify sets of functions associated with each role and then select which of them can be fulfilled by software agents and which have to involve human participation. For instance, consider the *TMA* that makes sure that a Cable TV Box was successfully installed before 16:13 at a specific address and if this is not the case, raises an alarm, and a *Personal Agent* that helps the human *PM* managing a team of coders. The process is as follows: the autonomous agent, when created to fulfill a given role is provided with required modules to accomplish it, e.g. the *TMA* obtains information about the deadline it is to observe and what to do if it is, or is not met. The situation is somewhat more complicated in the case of the *PA*. First, let us recall that every worker is represented in the system by her/his own *PA*. Furthermore, upon joining the system (and thus the organization) the *PA* registers with the *OPM*—the resource manager—and becomes one of available resources. In Figure 1 we have depicted the *PA* and conceptualized it as an interface between the human and the remaining parts of the system; as well as a "helper" that supports user in fulfilling her role. Since role can change, the *PA* has to be able to support user in anyone of them. However, in Figure 1 we were able to identify core functions of the *PA*, which are used regardless of a specific role. To provide support for the user who is assigned a specific role, modules facilitating functions associated with that particular role are then loaded into the *PA*, extending its functionality (for more details see [10]). Note that in the, somewhat more complicated case, when a team of analysts (*AM*) estimates feasibility of a project, each team member will be represented by its *PA*. Therefore, a role-specific set of interactions between these *PA*'s and humans they represent will constitute fulfillment of the role *AM*.

We utilize an AML [2] diagram in figure 2 to illustrate this general schema. In this figure we can see the generic agent the *VOAgent* which utilizes appropriate modules stored in the module / profiles library to become a *Personal Agent* first, and then play any role in support of its *User*.

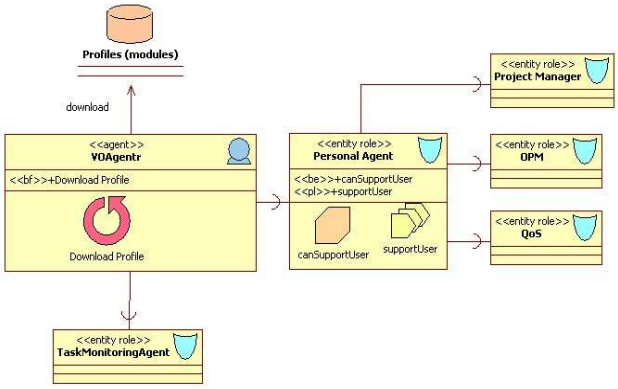

Fig. 2. AML Mental Diagram for the VOAgent

Observe that in this way, we have only "one basic type" of an agent in the system: the *Personal Agent* (supplemented by possibly some auxiliary agents). Furthermore, the *PA* can support user in playing *any* role identified in the organization. This in turn matches very nicely with the real world organization, where we also have only one main entity: human being that can play various roles identified in the organization.

4 Ontologies in the system

Thus far we have focused our attention on specific roles and their interactions involving various entities (human and non-human resources) in the system, and associating these entities with software agents alone and with agent-human teams. The other important developmental decision that was made was to utilize ontologies to represent (1) the domain of interest, (2) the structure of the organization, and (3) resource profiles. Let us now focus our attention on ontologies that are to be used in the system and start with a brief analysis of related work.

4.1 Toronto Virtual Enterprise (TOVE)

TOVE project run at the Enterprise Integration Laboratory of the University of Toronto. Its main goal was to establish generic, reusable enterprise data model that was to have the following characteristics [3, 1]:

- to provide a shared terminology for the enterprise that each entity within it can jointly understand and use,

- to define the meaning of each term in a precise and unambiguous manner,
- to implement the semantics in a set of axioms, to enable *TOVE* to automatically deduce the answer to "common sense" questions about the enterprise,
- to define a set of symbols for depicting a term or concept constructed thereof in a graphical context.

According to documents found within the project WWW site, ontology developed by the project included terms such as: resource, requirement, time, state or activity; and was created in Prolog. We thought about relying on the *TOVE* project and utilizing data model constructed there. Especially, since *TOVE* was based on extensive research and considered work of an enterprise from the design and operations perspectives [27]. Unfortunately, inability to find actual ontologies (except of conference papers), a long list of important features to be added found at the project web site, and the fact that the last update of that site was made on February 18, 2002, let us to believe that the TOVE project has died sometime in 2002. Therefore, we have decided to utilize only the theoretical part of *TOVE*.

4.2 OntoWeb

The *OntoWeb Network* is an initiative aiming at building a bridge between academics and the industry in order to promoting the Semantic Web [44]. The *OntoWeb Portal* of the *OntoWeb Network* project allows to insert and retrieve information about academic and industry employees, projects and documents [46]. Within the project the *OntoWeb* ontology was developed and made available at [45]. Unfortunately the *OntoWeb* ontology has also important drawbacks:

- The *OntoWeb* ontology is created in RDF Schema, which does not have rich enough semantics. Our experience with the RDF Schema shows that it is undeniably well suited for building conceptualizations [39]. However, in the case of a more complex software system, richer semantics and guaranteed computational completeness are desired. In particular, semantics of the RDFS which lacks quantifiers is hardly suitable for defining a data model (which involves defining cardinalities of entity relations) of the system. Therefore, reusing the *OntoWeb* ontology as the system core ontology would result in restricting types of reasoning available in the system.
- The *OntoWeb* ontology does not support resource profiles and information access restrictions, while they are necessary for the proposed system [9, 11].

Summarizing, we dropped the idea of reusing the *OntoWeb* ontology due to the limited expressivity of the RDF Schema and lack of necessary concepts. Instead, we followed guidelines and results obtained within both *TOVE* and *OntoWeb* projects and developed an ontology matching our project's needs. Let us therefore look into ontologies that have been developed within our system.

4.3 Generic Top-level Ontology of the Organization

Before we start let us note that delivering a comprehensive ontology for modeling an organization is beyond the *current* scope of our project. Our main aim is to deliver a framework for adaptive resource management (information provisioning in particular). Hence, the proposed ontology may not include all the necessary features to design model of any organization. However, we believe that ontology requirements considered at this stage have been specified to support currently-necessary functions of the system. Furthermore, they are flexible enough to support its future extension in order to support comprehensive organization modeling. Keeping this in mind, let us look into main ontologies of the proposed system.

We have decided to use OWL-DL as the ontology demarcation language, as it guarantees computational completeness and rich semantics—utilizing the Description Logic [4]. As mentioned above, one of main ideas of our approach is to model *every* entity within the organization (including humans) as a *resource*. Furthermore, each resource will have a *profile* and, depending on its type and role, may appear in a context of multiple profiles. For instance, knowledge about a person may be described with any of the following (and not limited to these) profiles: professional experience, education, personal, accommodation preference or dining preference. In Figure 3 we depict the generic resource and the generic profile concepts.

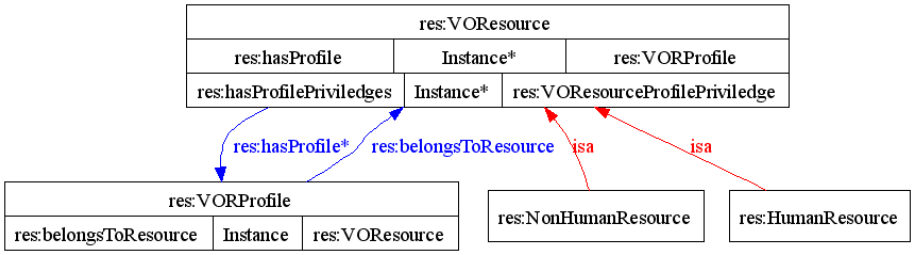

Fig. 3. Generic resource and generic profile concepts

A *resource profile* provides detailed information about any resource (human or non-human). It is composed of a resource specific data and "opinions" about other ontology concepts or ontologically demarcated objects [22]. Classes *VOResource* and *VOProfile* are designed to be extended by any organization specific resources and their profiles (assuring that the concept is robust and flexible). Deriving these core concepts in domain ontologies allows to define more organization-specific resource, such as: an *employee* of the cable TV installation company or academic institution; a *book* in a library; *requirements specification* (*SRS*) document in an IT company; or a *Duty Trip Report* in an organization that requires its employees to deliver such reports.

Note that some resource profiles may consist of private or classified information (e.g. personal data) therefore it is necessary to build an infrastructure which

can restrict access to the information. This is also important since accessibility to certain documents depends on employees "position" within an organization (e.g. annual evaluation of a worker should be visible only to that worker and her supervisors, but not her co-workers). A *VO Resource Profile Privilege* is a class which describes restrictions established for a profile. It binds a profile with a restriction type which is applied to all resources from a particular *Organization Unit (OU)*—whenever information is requested by, or matched with, resources. The binding of the *OU* and a particular *Profile Privilege Type* is realized by the *Profile Privilege* class.

The *Profile Privilege Type* is an enumerable type specifying supported access privileges: *Read*, *Write* and *Admin*. Names of the first two are self-explanatory, while the third (*Admin*) type represents an administrative privilege which allows to modify access restrictions of the profile. Here, for instance, the *HR Department* is expected to have *Write* privileges for worker profiles, while the *PA* is going to have *Read* privileges for information provided by the *OPM* (see Figure 1). The design of the *Profile Privilege* is depicted in Figure 4.

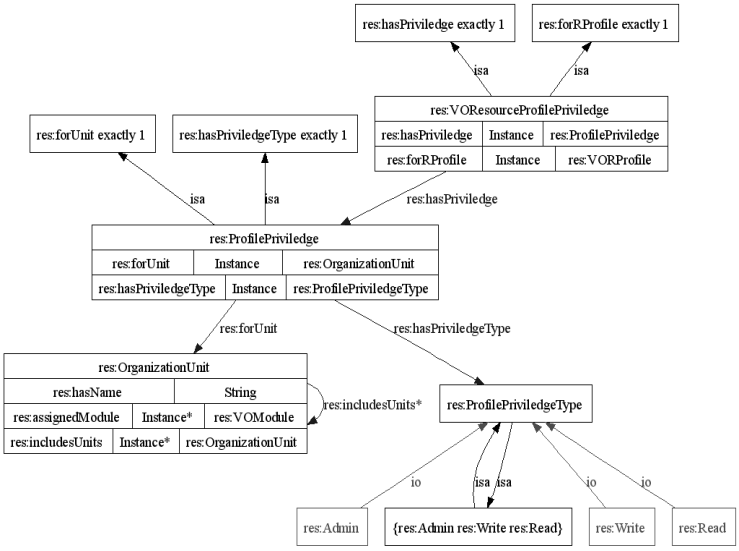

Fig. 4. Profile Privilege design

4.4 Demonstrator Applications

To extend our discussion of ontologies to be used in the system, let us introduce two applications depicted in Figure 5. In the *Grant Assistant System* (*GAS*), the *OPM* of a university (or a research institute) receives grant announcements and its role is to deliver them to these and only these *PA*s that represent *Users* that may be interested in them. In other words, the *OPM* (see figure 1) has to decide who (which *PA*(s)) should receive a given announcement, based on

ontologically demarcated profiles describing faculty in the university (researchers in the institute) and profiles of grant announcements. Here, we assume that the announcement is a resource that has already an assigned profile based on the internal domain ontology (note that specifying entity inside, or outside, of the system that performs profile demarcation is of no importance here).

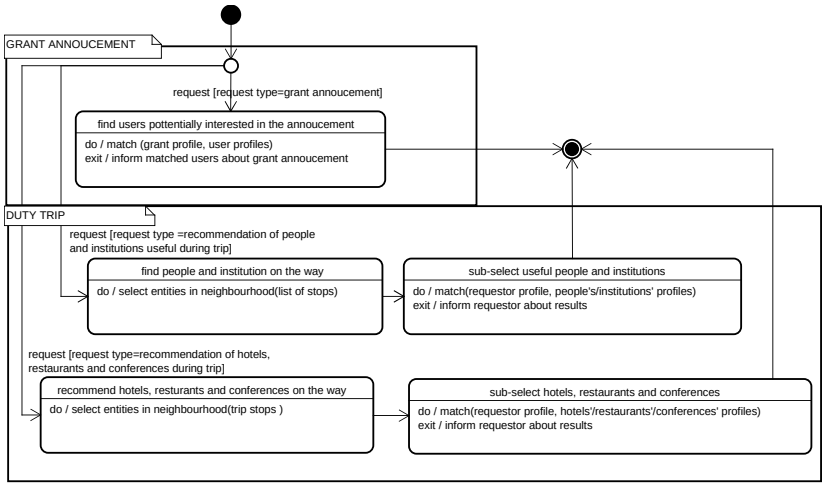

Fig. 5. Matching scenarios in the proposed system

The *Duty Trip Support* (*DTS*) scenario (specific to a Research Institute in East Asia, but easily generalizable) is more involved. Here, workers use the intranet to apply for a *Duty Trip* and to submit trip report. Our aim is to utilize results obtained in our project to provide them with additional functionalities. First, note that for the Institute in question, cost of air travel (to most destinations outside of East Asia) is much higher—in a relative sense—than costs of a stay extended by a few days. Thus, an employee traveling to a given city (e.g. in Europe or America), may visit also near-by-located institutions (e.g. universities or companies), or persons that her institute has contacts with. Second, a recommender where to stay and eat could be of value (e.g. consider Indonesian researchers confronted with typical Irish food). In addition to personalized information delivery, the system is expected to help researchers in all phases of duty trip participation; from the preparation of the initial application until filing the final report. Note that the *Trip Assistant* is actually a role played by the *OPM*, which provides the requested personalized input to the *PA* (see function *Searching for resource* in Figure 1). In Figure 6 we present the activity diagram of the *Duty Trip Support*. In this diagram we can see two moments when the *PA* communicates with the *Trip Assistant* (*OPM*), first when the application for the trip is prepared (and institutions/people to visit are sought), second, when actual details of the trip (e.g. hotels) are to be selected.

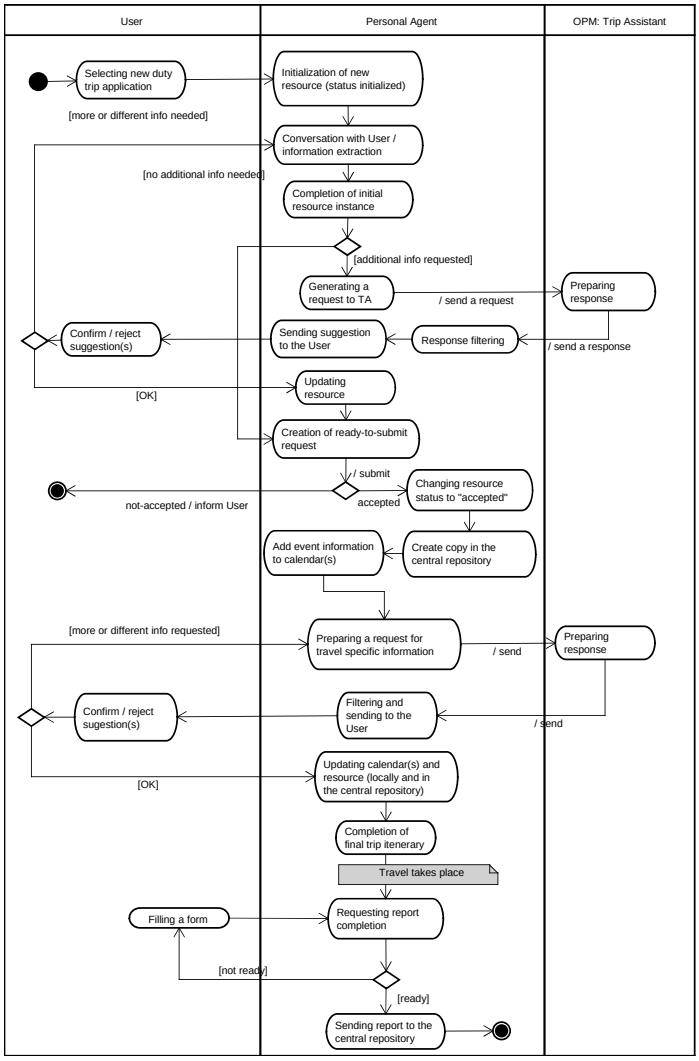

Fig. 6. Activity Diagram of the *Duty Trip Support* subsystem

4.5 Ontologies for the *Institute of Science and Technology* (*IST*)

To illustrate how the proposed ontology can be utilized in a specific organizational setting, let us discuss briefly its application to selected features of an ontological model of an Institute of Science and Technology (the *IST* ontology). In the architecture of our system, the *Domain Ontology* is an extension of the *Generic Ontology* outlined in Figure 3. Here, human resources are modeled in a way that is specific to the *East Asian Institute of Science and Technology*, though similarities with general human resource descriptions can be seen. Let us start from the *ISTPerson*, which is a class describing all employees of the Institute. Figure 7 illustrates the *ISTPerson* concept.

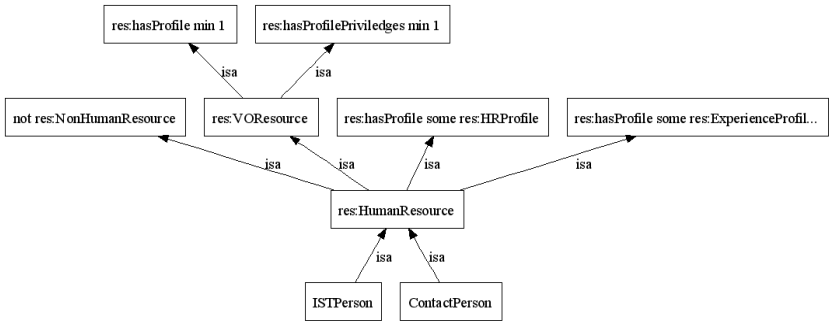

Fig. 7. Ontological description of the person on the Virtual Organization

While human resources have (multiple) general profiles, according the ontology, currently the following profiles can be assigned to employees of the Institute:

- *FIST Experience Profile.*
- *FIST Person Profile.*
- *Organization Profile.*
- *Dining Preference Profile.*
- *Accommodation Preference Profile.*

Here, the *FIST Experience Profile* allows to describe both educational and professional experience of the employee (and is depicted in figure 9). Professional experience is represented as a "project history" in which a given worker participated, while working in the organization. The educational experience lists academic degrees of the employee.

Additionally, specification of (multiple) research field(s) further describes employees competences (research fields used here are based on the South Asian RFCD [28]). Note that, as described in the last section, it is also possible to assign level of competence for each research field [12].

The *Personal Profile*, presented in figure 8, is a set of data typically stored by the *HR Department*. It represents personal data of an employee.

The *Organization Profile* specifies, for instance, a division in which the employee works; it can be also used to establish who is the supervisor of an em-

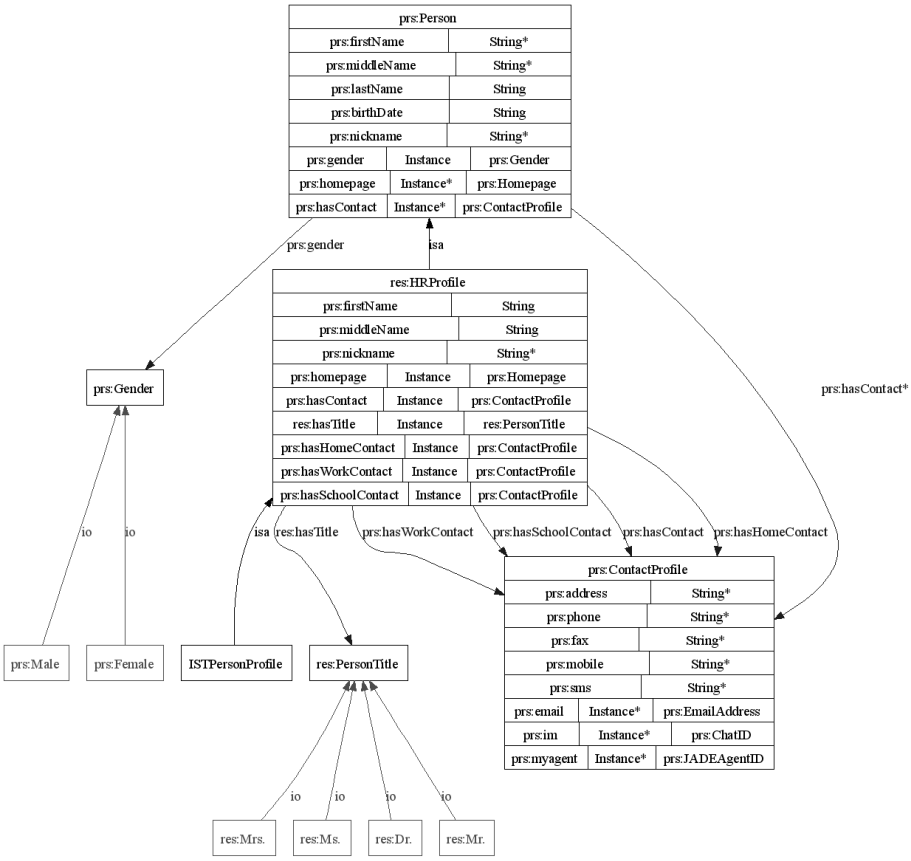

Fig. 8. Ontological Description of the Personal Profile

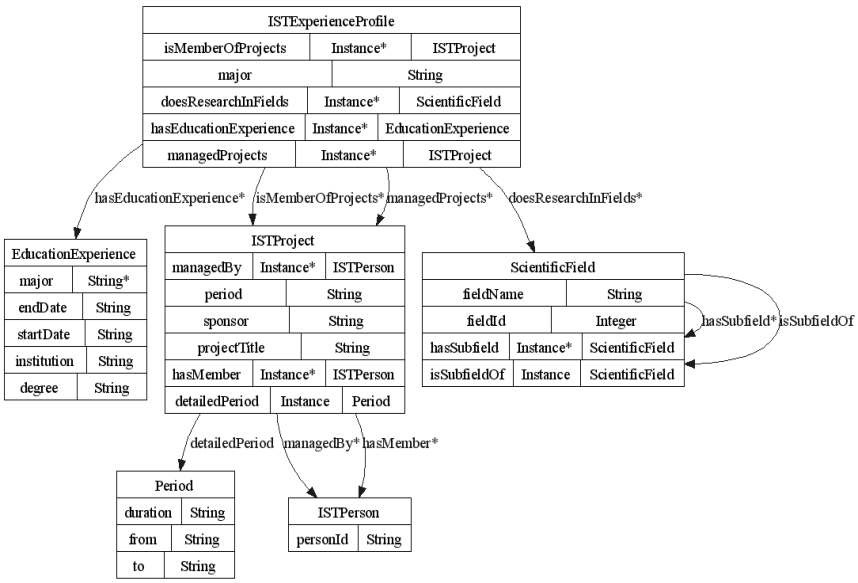

Fig. 9. Ontological Description of Employees Competences

ployee. Finally, *Accommodation Preference* and *Dining Preference* profiles represent ones attitude toward restaurants and hotels visited thus far. These concepts establish a link between the *Travel Support System* ontology ([14]) and the *IST* ontology. They utilize the *Hotel* and the *Restaurant* classes which are defined in the *TSS* ontology. While the *ISTPerson* is an example of a human resource in the domain of the *IST*, the *ISTDutyTrip* (*DTR*) is an example of a non-human resource (see figure, 10). This class represents a duty trip description from the *DTS* scenario and, as a resource child class, all its instances may have various profile instances assigned. However, the *DTR* concept is restricted to have all assigned profile instances of no other class than the *ISTDutyTripProfile*, which is designed to describe a potential duty trip in possibly big detail (for more details on the *Duty Trip* and this profile type, see below). The discussed profile may carry also information about accommodation and dining preferences which refer to the *Hotel* and the *Restaurant* class instances. It is easy to notice that, similarly to the case of *Accommodation Preference* and *Dining Preference* profiles, this is another link between the *IST* ontology and the "travel objects" of the *TSS* ontology. Additionally, the OWL class range of the *ISTDutyTripProfile destination* property, which is defined in the *IST* ontology, refers to the *PlaceOnEarth* class which derives the *SpatialThing* class of the *TSS* ontology.

Concerning the *TSS* ontology, let us note that we are currently using only its minimalistic OWL-DL version. In the near future, for description of "travel objects," we intend to utilize the full version of the *TSS* ontology. However, since it was created in the RDF Schema [5], this will require an adequate migration

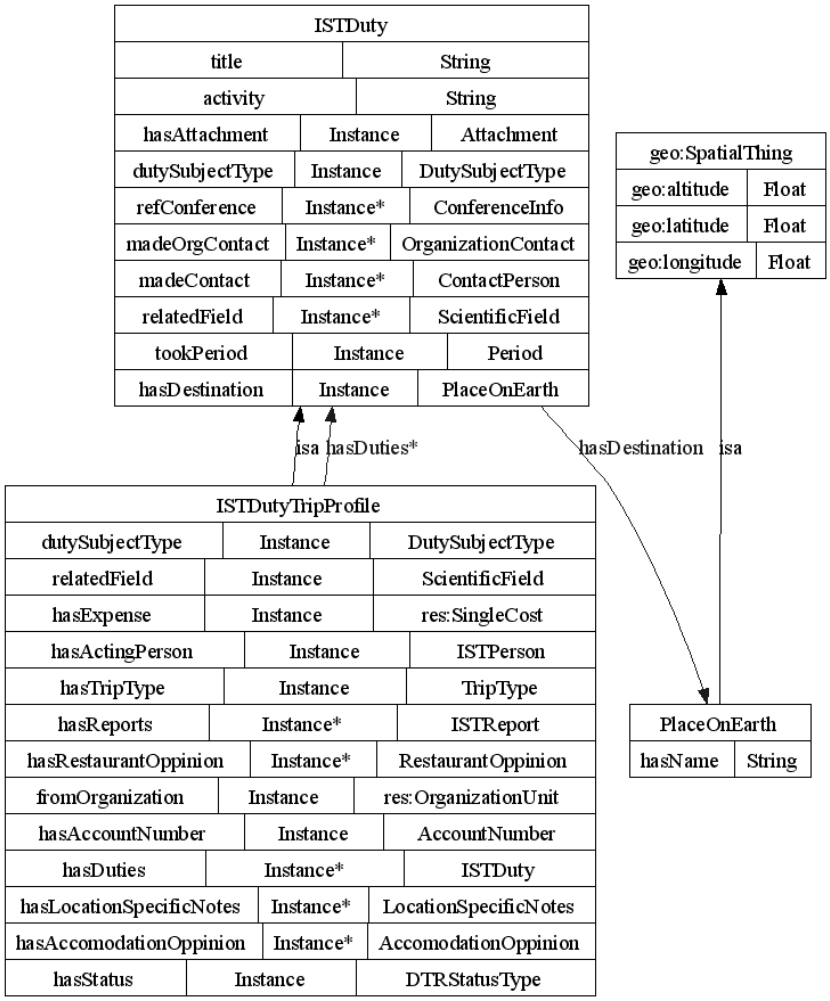

Fig. 10. Duty Trip Report Profile

to OWL-DL; for the sake of compatibility with the *Duty Trip Support* and the *Grant Announcement* subsystems. In figure 11 we present the simplified version of the *TSS* ontology used in the system today.

Another example of a non-human resource subclass is the *ISTAnnouncement* which represents grant opportunities in the *GSA* scenario. This class has a property *refScientificField* which refers to an instance of the *ScientificField* class. Please note that instances of the same class are also referenced by instances of the *ExperienceProfile*. Hence, a relation between instances of the *ISTAnnouncement* and the *ISTPerson* who has her *ExperienceProfile* defined may be established. This issue will appear again in detail when we discuss ontological matching applied in the system.

4.6 Using Proposed Ontology to Demarcate Sample Resources

Let us now present a collection of samples of demarcating specific resources in the *Virtual Organization*, using concepts introduced thus far. Note that due to the limited space, we can only point to a few aspects and we hope that the reader will be able to follow the example and find more features. The initial context is provided by a *Duty Trip* to a conference in Oulu, Finland, where Mr. Jackie Chan (who comes from Hong-Kong) will stay in a Radisson SAS Hotel (and visit also Mikka Korteleinen in Rovaniemi). We start by illustrating (1) how the geolocation will be demarcated (following the travel ontology proposed in [14]), and (2) the direct connection between the travel ontology ([14]) and the organization ontology as the cities *geo:OuluCity*, *geo:HongKongCity* and *geo:RovaniemiCity* are instances of travel ontology element: *SpatialThing* and organization ontology class *City*.

```
geo:FinlandCountry a onto:Country;
        onto:name "Finland"^^xsd:string.
geo:ChinaCountry a onto:Country;
        onto:name "China"^^xsd:string.
geo:OuluArea a onto:Area;
        onto:name "Oulu"^^xsd:string;
        onto:isInCountry :FinlandCountry;
        onto:adjacentArea :LappiArea.
geo:LappiArea a onto:Area;
        onto:name "Lappi"^^xsd:string;
        onto:isInCountry :FinlandCountry;
        onto:adjacentArea :OuluArea.
geo:HongKongArea a onto:Area;
        onto:name "Hong␣Kong"^^xsd:string;
        onto:isInCountry :ChinaCountry.
geo:OuluCity a onto:City;
        onto:name "Oulu"^^xsd:string;
        onto:long "25,467"^^xsd:float;
        onto:lat "65,017"^^xsd:float;
        onto:isInArea :OuluArea.
geo:RovaniemiCity a onto:City;
        onto:name "Rovaniemi"^^xsd:string;
        onto:long "25,8"^^xsd:float;
        onto:lat "66,567"^^xsd:float;
        onto:isInArea :LappiArea.
geo:AberdeenCity a onto:City;
        onto:name "Aberdeen"^^xsd:string;
        onto:long "114,15"^^xsd:float;
        onto:lat "22,25"^^xsd:float;
```

Fig. 11. Minimalistic version of the *TSS* ontology used in the system

```
onto:isInArea :HongKongArea.
```

Here we have defined the *FinlandCountry* and the *ChinaCountry*. The presented snippet includes also instances of *OuluCity*, *RovaniemiCity* and *AberdeenCity* cities, and their region related data: *OuluArea*, *RovaniemiArea* and *HongKongArea*. The first two instances represent countries in which these cities and regions are located: China and Finland. Country, Region and City are three levels of administrative land division that we initially intend to support in the system. The issue of populating database with real life geospatial and administrative information will be discussed in the future as the most suitable methods for this purpose are still being researched. Let us note that we do not claim the above proposed representation of geospatial information is the most efficient solution to the problem, but we assume that it is sufficient enough for the purpose of our information provisioning system and the *Duty Trip Support* application.

The listing that follows shows a simple instance of the Radisson SAS Hotel in Oulu, demarcated according to the simplified *TSS* schema. Note that the hotel feature *locatedAt* references instance of a *City* and *SpatialThing* classes—the *OuluCity*. It is the direct connection of the *TSS* ontology and *VO Ontology* which was discussed above (see also, 11).

```
hot:OuluRadisonSAS a tss:Hotel;
  onto:locatedAt geo:OuluCity.
```

ContactPerson#1 represents a human resource that is not employed at the Institute but is recognized because it has been introduced in the past to the system by one of the Institute's employees. According to the example beneath, Mikka Korteleinen is a researcher who specializes in Paleontology and is located in Rovaniemi, Finland.

```
:ContactPerson\#1 a onto:ContactPerson;
        onto:hasProfile :ContactPersonProfile\#1.
:ContactPersonProfile\#1 a onto:ContactPersonProfile;
        person:fullname ''Mikka Korteleinen''^^xsd:string;
        person:gender person:Male;
        person:birthday "1967-11-21T00:00:00"^^xsd:dateTime;
        onto:doesResearch science:Paleontology-13108;
        onto:locatedAt geo:RovaniemiCity;
        onto:belongsTo :ContactPerson\#1.
```

In the next snippet we introduce instances of the *ISTPerson* and *OrganizationUnit* classes. These instances represent Mr. Jackie Chan and Ms. Mi Lin who are employees of the Institute. The organization units to which Mr. Chan and Ms. Lin belong to reflect their positions in the organizational structure of the Institute.

```
:HROU a onto:OrganizationUnit;
  onto:name ''Human Resource Management Organization Unit''^^xsd:string.
:GOU a onto:OrganizationUnit;
  onto:name ''General Organization Unit-suitable
              for all employees''^^xsd:string.
:Employee\#1 a onto:ISTPerson;
        onto:id "1234567890"^^xsd:string;
        onto:hasProfile (:Employee\#1PProfile, :Employee\#1EProfile);
        onto:hasProfilePriviledges :ResProfPriv\#2.
        onto:belongsToOUs (:GOU).
:Employee\#2 a onto:ISTPerson;
             onto:id "011111111"^^xsd:string;
             onto:hasProfile (:Employee\#2PProfile);
             onto:belongsToOUs (:GOU, :HROU).
```

Detailed personal information of each of these employees is described in separate instances of the *ISTPersonalProfile* class. Such profiles could look as follows:

```
:Employee\#1PProfile a onto:ISTPersonalProfile;
  onto:belongsTo :Employee\#1;
  person:fullname "Jackie Chan"^^xsd:string;
  person:gender person:Male;
  person:birthday "1982-01-01T00:00:00"^^xsd:dateTime.
:Employee\#2PProfile a onto:ISTPersonalProfile;
  onto:belongsTo :Employee\#1;
  person:fullname "Mi Lin"^^xsd:string;
  person:gender person:Female;
  person:birthday "1981-02-01T00:00:00"^^xsd:dateTime.
:Employee\#1EProfile a onto:ISTExperienceProfile;
  onto:belongsTo :Employee\#1;
  onto:doesResearchInFields
    scienceNamespace:Volcanology-13105,
    scienceNamespace:Paleontology-13108,
    scienceNamespace:Geochronology-13204;
  onto:knowsFields
    [a onto:Knowledge;
    onto:knowledgeObject scienceNamespace:Volcanology-13105;
    onto:knowledgeLevel "0.75"^^xsd:float],
    [a onto:Knowledge;
    onto:knowledgeObject scienceNamespace:Paleontology-13108;
    onto:knowledgeLevel "0.40"^^xsd:float],
    [a onto:Knowledge;
    onto:knowledgeObject scienceNamespace:Geochronology-13204;
    onto:knowledgeLevel "0.90"^^xsd:float];
  onto:managesProjects (:Project1).
:Project1 a onto:ISTProject;
  onto:managedBy :Employee\#1;
  onto:period
    [a onto:Period;
    onto:from "2008-06-01T00:00:00"^^xsd:dateTime;
    onto:to "2009-05-31T00:00:00"^^xsd:dateTime];
 onto:fieldsRef scienceNamespace:Volcanology-13105;
 onto:projectTitle``Very Important Volcanology
        Scientific Project''^^xsd:string.
```

Note that from the snippet above we can establish that a person identified as *Employee#1* specializes in *Volcanology* and his *level of knowledge* is identified as 0.75 (for more info about assigning levels of skills, or more generally "temperature" to a feature, see [12, 14, 22]), *Paleontology* (level of knowledge identified as 0.4), and *Geochronology* (level of knowledge 0.9). Additionally, this person is scheduled to manage a project entitled: "Very Important Volcanology Scientific Project", which starts on June 1st, 2008 and ends on May 31st, 2009.

Obviously, scientific interests of a given employee (*onto:knowsFields* in the above example) can be replaced by professional skills describing worker in any discipline. For instance, they could as well be used to specify that a given programmer has knowledge of Smalltalk (level 0.7), Fortran (level 0.5), dBase (level 0.65), etc. In this way the proposed approach is both robust and flexible.

Having defined human resources, in what follows, we define an exemplary *Duty Trip Report* as a non-human resource.

```
:DTR\#1 a onto:ISTDutyTripReport;
   onto:hasProfile (:DTRProfile\#1);
   onto:hasProfilePriviledges :ResProfPriv\#1.
:DTRProfile\#1 a onto:ISTDutyTripReportProfile;
   onto:destination geo:OuluCity;
   onto:traveler :Employee\#1;
   onto:status dtStatusNamespace:Application;
```

```
    [a onto:Period;
    onto:from ''2008-06-07T00:00:00''^^xsd:dateTime;
    onto:to ''2008-06-19T00:00:00''^^xsd:dateTime.].
    onto:stayedAt hot:OuluRadisonSAS
    onto:expense [a onto:SingleCost;
          ''4000''^^xsd:float;
          onto:expenseCurrency ''USD''^^xsd:string.]
    onto:duty :AdditionalDuty\#1;
    onto:purpose ''Conference''^^xsd:string;
    onto:belongsTo :DTR\#1.
:AdditionalDuty\#1 a onto:ISTDuty;
          onto:destination geo:RovaniemiCity;
          onto:madeContact :ContactPerson\#1.
```

Here the *DTRProfile#1* is a profile of resource represented by the *DTR#1*. In our example the latter is a *Duty Trip Report* resource. The employee who this profile directly refers to, is represented by the *:Employee#1*. Hence, we can tell that a person represented in the system as *:Employee#1* applied for a duty trip (*DTR#1*). The current status of that Duty Trip Report is *Application* and the trips destination is Oulu, Finland. The researcher intends to stay there for twelve days (from June 7th till June 19th, 2008). Again, properties of the *ISTDutyTripProfile* refer to the system schemas (organization and domain ontologies) and fulfill the data model requirements set by the Duty Trip Support System which we develop (see also [38]). Please note that, in order not to overly complicate the example, the snippet above does not cover all properties of the *ISTDutyTripProfile* class which deprecated the *DTProfile* class defined in [38].

```
:ProfPriv\#1 a onto:ProfilePriviledge;
             onto:forUnit :HROU;
             onto:hasPriviledgeType priv:Admin.
:ProfPriv\#2 a onto:ProfilePriviledge;
             onto:forUnit :GOU;
             onto:hasPriviledgeType priv:Read.
:ResProfPriv\#1 a onto:VOResourceProfilePriviledge;
             onto:forRProfile :PersonalProfile\\#1;
             onto:hasPriviledge (<:ProfPriv\\#1>).
:ResProfPriv\#2 a onto:VOResourceProfilePriviledge;
             onto:forRProfile :DTRProfile\\#1;
             onto:hasPriviledge (<:ProfPriv\\#1> <:ProfPriv\\#2>).
```

In the snippet above we define a set of resource profile privileges which were presented in figure 4. Observe that the defined privileges allow members of the *HR* unit (in case of our example: Ms. Mi Lin) to *administer* selected profiles (e.g. *:PersonalProfile#1*), while members of the *General Organization Unit* are only allowed to *read* it (by default all access is forbidden). On the other hand, the *:DTRProfile#1* can be read by all employees of the *GOU*. In this way we assure control of access rights within the organization.

Finally, in order to discuss matchmaking that is going to take place in the system, let us introduce one more example of a non-human resource—an exemplary grant announcement. The main topic of this grant is: *Geochemistry*.

```
:SampleGrant a onto:ISTAnnouncement;
   onto:hasDescription ''Description of the exemplary
    grant announcement. It should be really interesting.''^^xsd:string;
   onto:refScientificFields
                        (<scienceNamespace:Geochemistry-13200>).
```

Note that the *SampleGrant* does not have its own profile and all its attributes are defined as its bare properties because there is not need to restrict access to information about *Grant Announcements* in the system with the use of *VOResourceProfilePriviledge* instances.

5 Matchmaking in the System

In the *Duty Trip Support* scenario, the *OPM* undertakes the role of a *Travel Assistant*, while in the *Grant Assistant* scenario it plays the role of a *Document Dispatcher* (in a real organization the first role may be played by a human supported by its *PA*, while the latter by a sfotware agent alone). Let us now describe how the desired results (finding personalized information or delivery of the document to the correct set of workers) are to be obtained. Before we proceed let us stress that we assume that all data *within* a given organization is demarcated utilizing a common (for that organization) ontology. Therefore, in what follows we do not have to deal with matching differing and potentially incompatible (external) ontologies. All that we are interested in is: how to establish "distances between resources" within a single ontology and how to use this information in the above described scenarios.

Let us now consider, introduced above, sample profile of a human resource—the *Employee#1* human resource and his profile; and demarcated non-human resources—the Duty Trip Report (*:DTRProfile#1*) and the *SampleGrant*. These profiles will allow us to introduce and briefly discuss matchmakings that are to take place in the system.

5.1 Calculating distances between resources

From the scenarios described above and summarized in figure 5, we can easily see the need for resource matching (finding distances between two or more resources). To focus our attention, let us present a few examples of types of resource matching operations that have to be implemented in our system (this list is not intended to be exhaustive, but rather to point to some classes of needed resource matching and/or distance calculations):

1. computing distance between two geographical locations; to be able to establish if a given location is close-enough to the place where the employee is to travel (so that she can attempt at visiting another institution and/or colleague),
2. matching a non-human resource (e.g. a grant, hotel, restaurant, conference) with a human-resource; to find if a person who is planning a trip could be interested in a given nearby located conference, or if an employee is potentially interested in a grant announcement,
3. matching two human resources to find out who are the researchers that a person planning a trip may be interested in visiting.

Upon further reflection it is easy to notice that the way the distance between resources should be calculated depends on types of objects which are arguments of calculations. For example, the distance between value of *onto:destination* property of the *DTRProfile#1* and the value of the *onto:locatedAt* of the *ContactPersonProfile#1* instance will be calculated in a different way than the distance between values of *onto:refScientificField* property of the *SampleGrant* and the *ExperienceProfile#1* instances. The following object types that appear in our work can be distinguished, based on different approach to calculate their distance (calculations specified here involve the above presented ontology snippets):

1. Objects which represent geographical locations—distance between the *onto:destination* property range of the *onto:ISTDutyTripProfile* class and the *onto:locatedAt* of the *onto:ContactPersonProfile* class.
2. Numeric objects—distance between *onto:long* property values.
3. Date objects—distance between *onto:from* and the *onto:from* (or the *onto:to*) property values.
4. Enumerable objects—distance between *onto:refScientificField* property values/range of the *onto:ISTAnnouncement* and the range of *onto:doesResearchInFields* property of the *onto:ISTExperienceProfile* class.

Let us now discuss possible approaches to distance calculations/resource matching for the four distinguished classes of properties.

Location based calculations *City*, *Country* and *Area* are classes designed to represent geographical locations which may be visited by the *User*. These classes have properties which allow to build a tree structure of countries, areas and cities. For instance, *FinlandCountry*, *LappiArea*, *OuluArea*, *ChinaCountry*, *HongKongArea*, *RovaniemiCity*, *OuluCity* and *AberdeenCity* were samples of geo-locations introduced above. They represent a part of an administrative division of Finland and China. First level in our structure is a country, the second level is an area and finally city is the third one. Available properties allow to query for neighbor (adjacent) instances of the same class. This approach requires access to administrative divisions of the world data, otherwise it may be of little value in terms of facilitating a location based advice. Apart from the administrative division tree, these classes allow to describe actual geo-coordinates of objects. Location based advising can be performed by calculating object's distances using the general formula (*long* - longitude, *lat* - latitude, *alt* - altitude):

$$\sqrt{(long_0 - long_1)^2 + (lat_0 - lat_1)^2 + (alt_0 - alt_1)^2}$$

Note that in most business travel scenarios the altitude (alt_0 and alt_1) is of little relevance and can be omitted. Obviously, similar calculations can be performed not only for conferences and/or institutions, but also for all other geo-objects (e.g. restaurants and hotels) as their coordinates are described in the same way as cities (hotel, restaurant and city are subclasses of the *Spatial Thing* class in our travel ontology; see [14, 39]). Therefore, the *DTS* system will be able to provide at least the following geo-info-based advice:

1. Location notes and tips (textual information about a location which was added to Duty Trip Reports - class in the ontology: *Location Specific Notes*),
2. Organizations and people that can be visited (objects of *Organization Contact* and *Contact Person* classes, these objects are created by the employees during the Duty Trip Report's creation),
3. Information about nearby conferences of possible interest (based on location of the trip and the conference as well as on the personal interest and conference topics),
4. Hotels and restaurants (based on the *Hotel* and *Restaurant* TSS ontology classes),
5. Car rental and golf courses (ontology extensions based on the OTA specification [31], also included in the *TSS* ontology).

5.2 Numeric and date object calculations

Computing distance between numeric and date object is rather obvious. The distance will be represented by the result of difference operation on these objects. In the first case, the result will be a number, in the latter case the result will be a time period (e.g. of a stay in a given place). Note that currently most major programming languages provide date calculation support hence we believe this issue should not be discussed in more detail (assuming there are no problems with date representation and deserialization).

5.3 Enumerable object calculations

In case of an ontology, enumerable values can be more complex than *enums* known from popular programming languages. In an ontology, class instances can also be enumerable values. In that case complex structures can be constructed, representing relations between objects. For instance, presented above *ScientificField* class falls under the OWL *oneOf* restriction, however each instance of that class has property values which refer to other instances of that class. This results in a graph-like structure of enumerable values.

To calculate distance between two object of enumerable type, let us note first that if the structure of *enum* values is flat (plain list with no relations between objects) it can be assumed that the distance is 0 if the values match, otherwise it equals to 1. An example of such simple enumerable is the *Gender* property, which is utilized in the human resource profile. Here we have two values: *Male* and *Female* and if they match the distance is 0, and 1 otherwise.

Let us now present a method for calculating distance between class instances which involve transitive, non-symmetric properties. Here, a path in a directed graph is calculated for all relations. Let us assume that R is such a transitive, not symmetric relation (*property* in the OWL notation). Then the distance between two vertices of a graph of relation R: v_0 and v_k ($dist_R(v_0, v_k)$) is calculated according to the following algorithm:

1. If there exists $path_R(v_0, v_k)$ in the graph of relation R, then the shortest one can be found and

$$dist_R(v_0, v_k) = length(shorthestPath_R(v_o, v_k));$$

   otherwise go to 2nd step.
2. Let $X = \{x : path_R(x, v_0)$ and $path_R(x, v_k)$ exist$\}$. Find such $y \in X$, that $length(path_R(y, v_0))$ is minimal among all vertices belonging to X (i.e. this is the shortest path):

$$dist_R(v_0, v_k) = 10^{length(path_R(y,v_0))} + length(shorthestPath_R(y, v_k))$$

Note, that this is a simplified case of a method introduced in [34]. The basic difference between them is as follows. The method proposed in [34] assumes existence of multiple relations linking any class instance (node) from a single node and merging edges which represent relations of the same direction between the same nodes; thus, the distance is computed including all properties (relations) of classes (concepts). The algorithm presented above, on the other hand, is restricted to one selected property (relation) of a class (concept) and an inverse of the selected property. This pair represent generalization and specialization relations between concepts. The algorithm presented here can be substituted for the algorithm of [34] by adjusting appropriate weights to concepts relations. Specifically, used here weights of 1 for specialization and 10 for generalization.

Let us now describe calculation of distance between research interests of a human resource and grant announcement topics, while utilizing examples introduced above. According to the proposed algorithm the following distance values can be found (here we calculate all-against-all distance values):

$dist_{SF} = path_{isSubfieldOf}$
$dist_{SF}$(Volcanology−13105, GeologicalScience−13100)=10
$dist_{SF}$(Volcanology−13105, Geochemistry−13200)=101
$dist_{SF}$(Volcanology−13105, Geochronology−13204)=102
$dist_{SF}$(Volcanology−13105, Paleontology−13108)=11
$dist_{SF}$(Paleontology−13108, GeologicalScience−13100)=10
$dist_{SF}$(Paleontology−13108, Geochemistry−13200)=101
$dist_{SF}$(Paleontology−13108, Geochronology−13204)=102
$dist_{SF}$(Paleontology−13108, Volcanology−13105)=11
$dist_{SF}$(Geochronology−13204, Geochemistry−13200)=10
$dist_{SF}$(Geochronology−13204, GeologicalScience−13100)=101
$dist_{SF}$(Geochronology−13204, Volcanology−13105)=102
$dist_{SF}$(Geochronology−13204, Paleontology−13108)=102
$dist_{SF}$(Geochemistry−13200, GeologicalScience−13100)=11
$dist_{SF}$(Geochemistry−13200, Volcanology−13105)=12
$dist_{SF}$(Geochemistry−13200, Paleontology−13108)=12
$dist_{SF}$(Geochemistry−13200, Geochronology−13204)=1
$dist_{SF}$(GeologicalScience−13100, Geochemistry−13200)=11
$dist_{SF}$(GeologicalScience−13100, Volcanology−13105)=1
$dist_{SF}$(GeologicalScience−13100, Paleontology−13108)=1
$dist_{SF}$(GeologicalScience−13100, Geochronology−13204)=12

These values allow us to utilize a number of strategies to establish "closeness" of two resources. The simplest one would be, if for any two properties the distance is below a certain threshold, then a exemplary grant announcement should be recommended as potentially interesting. In case of Mr. Chan who is interested in *Volcanology*, *Paleontology* and *Geochronology* we may measure the distance between his interests and the exemplary grant announcement which main topic is *Geochemistry*. The results, which are part of the all-against-all calculation presented in the listing above, are as follows:

$dist_{SF}(\mathrm{Geochemistry}-13200,\ \mathrm{Volcanology}-13105)=12$
$dist_{SF}(\mathrm{Geochemistry}-13200,\ \mathrm{Paleontology}-13108)=12$
$dist_{SF}(\mathrm{Geochemistry}-13200,\ \mathrm{Geochronology}-13204)=1$

Here, since in one of the areas the distance is equal to 1, this grant announcement should be delivered to Mr. Chan. Note that distance could be also scaled by the level of knowledge of the specialist in the field. Furthermore, a number of more involved considerations are also possible. In this context let us note that values of the $dist_R(v_0, v_k)$ function allow us to specify how far are the graph nodes located from each other in terms of a transitive, not symmetric relation R. In the case of research specialization modeling relation we can assume that the maximum length of $path_R(v_0, v_k)$ is 9. In our ontology an example of such relation is the *isSubfieldOf* property of the *ScientificField* class, where the maximum length of $path_{SF}(v_0, v_k)$ is 2. Additionally, infinite distance is not considered. With such assumptions we are able to distinguish following groups of conclusions which can be drawn from the function values:

1. If $dist_R(v_0, v_k) = 0$, then v_0=v_k
2. If $dist_R(v_0, v_k) = n$ and $0 < n < 10$, then $R(v_0, v_k) = true$ and v_k is n-deep specialization of v_0
3. If $dist_R(v_0, v_k) = n$ and $n = 10^k, k > 0$, then $R(v_0, v_k) = false$ and v_0 is k-deep specialization of v_k
4. If $dist_R(v_0, v_k) = n$ and $10^k < n < 10^{(k+1)}, k > 0$, then $R(v_0, v_k) = false$ and v_0 is $n - 10^k$-deep specialization of k-deep specialization of v_k

For instance, if

$$dist_{SF}(Volcanology - 13105, Paleontology - 13108) = 11,$$

we may say that there is a node of which both *Volcanology* and *Paleontology* are direct specializations. In terms of the exemplary grant announcement and Mr Chans experience profile we may conclude that the grant announcement may be of potential interest to him. In particularly the distance between *Geochemistry* and *Geochronology* equals to 1, meaning that *Geochronology* is the direct offspring of the *Geochemistry* field of science.

These observations allow us to develop a number of reasoning scenarios that utilize not only information about numerical distance, but also following form it knowledge about the structure of relations. Developing a reasoning engine utilizing this information is next step in our work.

6 Concluding Remarks

The aim of this chapter was to summarize our work concerning development of system responsible for resource management in an agent-based virtual organization. First, we have outlined the proposed system in the context of a task introduced to the organization. We followed with a discussion of interrelations between entities belonging to the real-world organization and software agents that belong to the virtual organization. Next, we have focused on ontologies that are to be used in the system. We have introduced the general ontology of a virtual organization and followed it with discussion of the way that these ontologies can be extended to deal with a specific case of a research institute. We have concluded the chapter with an overview of matchmaking procedures that are needed in the proposed system.

In the paper we have outlined a number of future research directions. At this stage the most important ones are: (1) completing subsystem for geospatial data processing, (2) implementing interface between the web-based data input and the agent-based system, (3) completing implementation of ontological matchmaking (as pertinent to the two application areas), (4) completing implementation and testing of two sample applications (*Duty Trip Support* and *Grant Announcement Support*). We will report on our progress in subsequent publications.

References

1. `http://www.eil.utoronto.ca/enterprise-modelling/index.html`
2. Cervenka, R., Trencansky, I., AML: The Agent Modelling Language, Birkhäuser, 2007
3. Fox, M.S., Gruninger, M., (1998), "Enterprise Modelling", AI Magazine, AAAI Press, Fall 1998, pp. 109-121
4. `http://www.w3.org/TR/2004/REC-owl-guide-20040210/\#OwlVarieties`
5. `http://www.w3.org/TR/rdf-schema/`
6. Biesalski, E., Abecker, A.: Human Resource Management with Ontologies. In: *Wissensmanagement. Professional Knowledge Management, Third Biennial Conference, WM 2005, Kaiserslautern, Germany, 2005, Revised Selected Papers.* LNAI 3782, Springer, 2005, 499–507.
7. Bizer, C. Heese, R., Mochol, M., Oldakowski, R., Tolksdorf, R. and Eckstein, R.: The Impact of Semantic Web Technologies on Job Recruitment Processes. In: *Proc. International Conference Wirtschaftsinformatik (WI 2005), Bamberg, Germany*, 2005.
8. Ganzha, M., Paprzycki, M., Popescu, E., Bădică, C., and Gawinecki, M.: Agent-Based Adaptive Learning Provisioning in a Virtual Organization. In: *Advances in Intelligent Web Mastering. Proc.AWIC'2007*, Fontainebleu, France. *Advances in Soft Computing* 43, 25–40, Springer, 2007
9. Ganzha, M., Paprzycki, M., Gawinecki, M., Szymczak, M., Frackowiak, G., Bădică, C., Popescu, E., Park, M.-W.: Adaptive Information Provisioning in an Agent-Based Virtual Organization—Preliminary Considerations. In: *Proceedings of the SYNASC'07 Conference*, IEEE CS Press, pp. 235-241.
10. Ganzha, M., Gawinecki, M., Szymczak, M., Frackowiak, G. Paprzycki, M., Generic Framework for Agent Adaptability and Utilization in a Virtual Organization—Preliminary Considerations, Proceedings of the 2008 WEBIST Conference, May, 2008, to appear

11. Szymczak, M., Frackowiak, G., Ganzha, M., Gawinecki, M., Paprzycki, M., Park, M., Resource Management in an Agent-based Virtual Organization — Introducing a Task Into the System. In: Proceedings of the MaSeB Workshop, IEEE CS Press, Los Alamitos, CA, 2007, 458-462
12. Gawinecki, M., Modelling User on the Basis of Interactions with a WWW Based System, Master Thesis, Adam Mickiewicz University, 2005
13. Gawinecki, M., Gordon, M., Paprzycki, M., Vetulani, Z.: Representing Users in a Travel Support System. In: Kwasnicka, H. et. al. (eds): *Proceedings of the ISDA 2005 Conference*, IEEE Press, 2005, 393–398.
14. Gawinecki M., Gordon M., Nguyen N.T., Paprzycki M., Szymczak M., RDF Demarcated Resources in an Agent Based Travel Support System. In: M. Golinski et. al. (eds.), Informatics and Effectiveness of Systems, PTI Press, Katowice, 2005, 303-310
15. Gawinecki, M., Gordon, M., Nguyen, N.T., Paprzycki, M., Zygmunt Vetulani, Z.: Ontologically Demarcated Resources in an Agent Based Travel Support System. In: R. K. Katarzyniak (ed.): *Ontologies and Soft Methods in Knowledge Management*, Advanced Knowledge International, Adelaide, Australia, 2005, 219–240.
16. Warner, M., Witzel, M.: Zarzadzanie organizacja wirtualna, Oficyna Ekonomiczna 2005
17. Barnatt, C.: Office Space, Cyberspace and Virtual Organization, "'J. of General Management"', (1995) 20(4), 78–92
18. Bleeker, S.E.: The Virtual Organization. In: Leading Organizations, ed.: Hickman, G.R., Sage, 1998
19. Grenier, R., Metes, G.: Going Virtual: Moving Your Organization into the 21st Century, Prentice-Hall, 1995
20. Goldman, S.L., Nagel, R.N., Preiss, K.: Agile Competitors and Virtual Organizations, Van Nostrand Reinhold, 1995
21. Dunbar, R.: Virtual Organizing. In: International Encyclopedia of Business and Management, ed.: Warner, M., Thomson Learning, 2001, 6709–6717
22. Greer, J., McCalla, G., Student modeling: the key to individualized knowledge based instruction. NATO ASI Series F, 125, 1993
23. HR-XML Consortium. `http://www.hr-xml.org/`.
24. Jena Semantic Web Framework, `http://jena.sourceforge.net/`
25. Jennings,N.R., An agent-based approach for building complex software systems, In: *Commun. ACM*, ACM Press,vol. 44, num. 4, 2001, pp. 35–41
26. Âă Agent Technology: Foundations, Applications, and Markets, ed. Nicholas R. Jennings and Michael J. Wooldridge, Âă Springer, 2002
27. Kim, H.M., Fox, M.S., Gruninger, M., An ontology for quality management—enabling quality problem identification and tracing, BT Technology Journal, Vol. 17, No. 4, 1999, 131–140
28. Korea Science and Engineering Foundation `http://www.kosef.re.kr/english_new/index.html`
29. Mochol, M., Wache, H., and Nixon, L.: Improving the Accuracy of Job Search with Semantic Techniques. In: *Business Information Systems*. LNCS 4439, Springer, 2007, 301–313.
30. M. Montaner, B. López, and J. L. de la Rosa, A taxonomy of recommender agents on the Internet, *Artif. Intell. Rev.*, 19(4), 2003, 285–330.
31. Open Travel Alliance, `http://www.opentravel.org/`
32. PostGIS : HOME, `http://postgis.refractions.net/`
33. `http://www.cs.rmit.edu.au/agents/#prometheus`

34. Rhee, S. K., Lee, J., Park, M. W.: Ontology-based Semantic Relevance Measure, In: S. Kim et.al. (eds.), Proceedings of the The First International Workshop on Semantic Web and Web 2.0 in Architectural, Product and Engineering Design, 6 pages, 2007 http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-294/
35. Schmidt, A., Kunzmann, C.: Towards a Human Resource Development Ontology for Combining Competence Management and Technology-Enhanced Workplace Learning. In: *OTM Workshops (2). On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops.* LNCS 4278, Springer, 2006, 1078–1087.
36. Semantic Web. http://www.w3.org/2001/sw/.
37. Szymczak, M., Frackowiak, G., Ganzha, M., Gawinecki, M., Paprzycki, M., Park M.-W.: Resource Management in an Agent-based Virtual Organization—Introducing a Task Into the System. In: *Proc. of the WI/IAT 2007 Workshops*, IEEE CS Press, Los Alamitos, 2007, in press.
38. Szymczak, M., Frackowiak, G., Ganzha, M., Gawinecki, M., Paprzycki, M., Park M.-W.: Adaptive Information Provisioning in an Agent-Based Virtual Organization—Ontologies in the System, submitted for publication
39. Szymczak, M., Gawinecki, M., Vukmirovic, M., Paprzycki M., Ontological Reusability in State-of-the-art Semantic Languages, in: C. Olszak et. al. (eds), Knowledge Management Systems, PTI Press, Katowice, 2006, 129-142
40. Training. http://en.wikipedia.org/wiki/Training.
41. Tzelepis, S., Stephanides, G.: A conceptual model for developing a personalized adaptive elearning system in a business environment. In:*Current Developments in Technology-Assisted Education*, Formatex Publishing House, v.III, 2006, 2003–2006.
42. Wooldridge, M.: *An Introduction to MultiAgent Systems*, John Wiley & Sons, 2002.
43. Wooldridge, M., Jennings, N.R., Kinny, D., The Gaia Methodology for Agent-Oriented Analysis and Design, In: *Autonomous Agents and Multi-Agent Systems*, vol. 3, num. 3, 2000, pp. 285-312
44. http://ontoweb.org/About/Deliverables/ppOntoweb.pdf
45. http://ontoweb.aifb.uni-karlsruhe.de/Ontology/index.html
46. Spyns, P., Oberle. D., Volz, R., Zheng, J., Jarrar, M., Sure, Y., Studer, R., Meersman R., OntoWeb—a Semantic Web Community Portal,Proc. Fourth International Conference on Practical Aspects of Knowledge Management, Vienna, Austria, 2002