Onto-Agent Methodology for Design of Ontology-based Multi-Agent Systems

Maja Hadzic, Elizabeth Chang

Curtin University of Technology, Digital Ecosystems and Business Intelligence Institute, PO Box 1987, Perth 6845, Western Australia, Australia
{m.hadzic, e.chang}@curtin.edu.au

Abstract. Multi-agent system provides a distributed collaborative platform and characterizes the system dynamics. Ontologies represent the domain knowledge and can be used to support various processes within a multi-agent system. Ontologies are high expressive knowledge models and as such increase the expressiveness and intelligence of a system. We propose Onto-Agent Methodology for design of ontology-based multi-agent systems, as the first methodology that unifies the different approaches of the existing ontology and multi-agent systems design methodologies. Onto-Agent Methodology is composed of the two interconnected processes: ontology design process and multi-agent system design process. Each of the two design processes is composed of five steps. We discuss each of the steps separately and provide some illustrative examples.

Keywords: ontology, multi-agent system, ontology design methodology, multi-agent system design methodology, ontology-based multi-agent systems, ontology-based multi-agent system design methodology, intelligent multi-agent system, intelligent multi-agent system design methodology

1 Introduction

Ontologies were brought into the computer and information society to be used by various agents and for different applications addressing the problems of various knowledge domains. Ontologies are high expressive knowledge models and as such increase expressiveness and intelligence of a system. A multi-agent system provides a distributed collaborative platform and as such determines the system dynamics. Ontologies represent the domain knowledge and can be used to support some important processes within a multi-agent system such as: problem decomposition and task sharing among different agents, result sharing and analysis, information retrieval, selection and integration etc.

There is not a single and consensual ontology design methodology. There are various multi-agent system design methodologies as well. Not much experience has been gained with these methodologies [1]. Some of these methodologies are discussed in Section 2. We propose an Onto-Agent Methodology as the first methodology that unifies the different approaches of the existing ontology and multi-agent systems design methodologies. This methodology is composed of the two interconnected processes. Firstly, we design an ontology on which the multi-agent system will be based. Secondly, we define the multi-agent system which functions and operates using the designed ontology. The five steps of first part of the Onto-Agent Methodology (Onto Methodology) are described in Section 3 and are accompanied by some illustrative examples. The second part of Onto-Agent Methodology (Agent Methdology) also consists of five steps which are described with illustrative examples in Section 4. The paper is summarized in Section 5. We conclude and give our final remarks in Section 6.

2 Related Work

In this section, we give a short overview of some of the existing ontology design methodologies and multi-agent systems design methodologies.

The Knowledge Engineering Methodology (KEM) [2] for designing and maintaining ontologies is identified as a six stage process: definition of the domain, purpose and scope of ontology; acquisition and conceptualization of the domain knowledge; reusability of existing ontologies; formal specification of the

ontology; populating ontology with individual instances; and evaluation and documentation. The DOGMA approach is based on the principle of double articulation [3]. Its framework consists of two layers: an ontology base which holds intuitive conceptualizations of a domain and a layer of ontological commitments. The TOVE methodology [4] consists of the following six steps: identify motivating scenarios, formulate informal competency questions, specify terminology within a formal language, formulate formal competency questions using a formal terminology, specify axioms and definitions within a formal language and specify conditions characterizing ontology completeness. The METHONTOLOGY [5] framework consists of two steps: characterization of the ontology development process (which includes ontology management, ontology development-oriented and ontology support activities) and ontology life cycle based on evolving prototypes.

Methodologies for the analysis and design of agent-based systems can be broadly divided into two groups [1]: the group of researchers who take their inspiration from object-oriented development (BDI, Gaia, Agent UML and Agents in Z methodology), and the group of researchers who adapt knowledge engineering and similar techniques (DESIRE and Cassiopeia methodology). BDI methodology and the modeling technique for a system of agents are based upon the belief, desire and intention paradigm [6]. In applying the Gaia methodology [1], the designer moves from the abstract to increasingly concrete concepts. AUML, the Agent UML (Unified Modeling Language) is a set of UML idioms and extensions that can be used to model multi-agent systems [7]. Agents in Z methodology [8] construct a model of the computational system by using the standard properties of the Z specification language and refining the abstract definitions to include the relevant system constraints. DESIRE, originally designed for formal specification of complex reasoning systems, has been extended to obtain a useful formal specification framework for multi-agent systems [9]. According to Cassiopeia method [10], a multi-agent system should be designed in terms of agents provided with three levels of behavior: elementary, relational and organizational.

None of these methodologies has become a standardized methodology for ontology or multi-agent system design. The importance of the ontology-based multi-agent system has been highlighted by different research groups [11, 12] but no standardized methodology for the design of such systems has been developed yet. There is a need to establish a standardized methodology for the development of ontology-based multi-agent systems. Each of the mentioned design methodologies shows some advantages. We believe that advantages of the mentioned methodologies can be combined and merged into a new and single methodology for the design of ontology-based multi-agent systems. This will enable defining ontology-based multi-agent systems from different points of view which will add stability and confidence to the design of such models.

We introduce a new Onto-Agent Methodology for the design of ontology-based multi-agent systems that unifies the different and complementary advantages of the existing ontology design methodologies into the Onto Methodology, and of the existing multi-agent systems design methodologies into a Agent Methodology.

3 Process I - Ontology Development

We consider the Onto Methodology to be composed of the following five stages:

1. Generalization and conceptualization of the domain
2. Aligning and merging ontologies
3. Formal specification of conceptualization
4. Specifying of ontology commitments
5. Ontology evaluation

This new ontology design methodology combines the advantages of the existing ontology design methodologies, and introduces some new features into the design process. The first and the second stage of KE methodology are included in the first stage of the Onto Methodology. We believe that two additional aspects need to be taken into account: (1) the community for which the ontology is being designed and (2) the application(s) which will operate on the basis of the designed ontology. This is the first step of TOVE methodology. The third stage of KE methodology is the second stage of Onto Methodology. We strongly agree that existing ontologies should be reused where possible. The fourth stage of KE methodology corresponds with the third and the fourth stage of Onto Methodology as we adopted the DOGMA principle of separation of the ontology base from the ontology commitments. In the TOVE methodology, the step of terminology specification is also separated from the step of specification of axioms and definitions (ontology commitments). Population of the ontology with individual instances needs to be carefully considered. In some cases, excessive number of ontology instances may affect ontology functionality, flexibility and

reusability. Highly specialized ontologies with a large number of instances have less chance to be used by a large number of users. In these or similar situations, access of the instances from external information resources (e.g. TAMBIS ontology [13]) through use of agents may be more appropriate. This gives the ontology a dynamic nature and enables it use by a larger range of users. All the concepts needed for the design of a new ontology are chosen in the second stage of Onto Methodology, and are formally specified in the third and the fourth stage. We agree with KE methodology that the designed ontology should be evaluated. This is the fifth and the last phase of Onto Methodology. One of the ways to an evaluate ontology is through the use of evolving prototypes as mentioned by METHONTOLOGY.

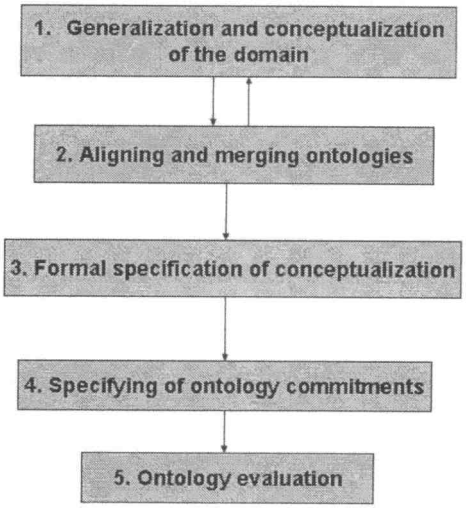

Fig. 1. Ontology design methodology

The five stages of the ontology design methodology are shown in Figure 1. The first and second stage may need to be alternated until the ontology borders and conceptual coverage of the domain have been clearly established and agreed upon.

We recommend a combination of top-down and bottom-up approaches during the ontology design process. In the first stages of the ontology design, a top-down ontology design approach may be preferred as it gives a better overview and control over the ontology design process. The amount of detail to be provided by the ontology will influence the starting point of the ontology design using the bottom-up approach. We believe that a combination of both approaches is the best option. Eventually, the ontology concepts incorporated during the bottom-up approach will meet those incorporated during the top-down approach.

3.1 Generalization and Conceptualization of the Domain

The aims of this step of the ontology design process are to:

- Identify main ontology concepts
- Establish ontology framework
- Determine organization of the concepts within the ontology

For this reason, four important points need to be taken into account:

- Community for which the ontology is being designed
- Purpose of the ontology
- Knowledge domain that will be represented by the ontology
- Application based on the designed ontology

Within the community for which the ontology is being designed, smaller communities can be further differentiated. Within the ontology context, we have identified three groups that may be associated with the term community. Firstly, we are talking about the community of ontology users. For example, within the community of people working towards the development of the ontology representation language, smaller communities working on specific ontology languages such as RDF, DAML, OIL, DAML+OIL, OWL etc. can be identified. Secondly, we have the community of agents. The ontology may be designed for a community of agents that are cooperatively working towards the same goal (multi-agent systems). Thirdly, we have the community of ontology designers. An ontology is best designed by a group of researchers with different and complementary capabilities because of the complexity of the ontology design process. The three identified types of communities are actually a part of one big community as they are working towards the same goal but on different levels. All the members within the community need to agree and commit to the designed ontology.

The main ontology purpose is to represent the knowledge of a specific knowledge domain. The following points also need to be included:

- why does this knowledge need to be represented
- what is to be expected from the proposed ontology-based application
- the end users and the advantages offered to them through this application

Ontologies are usually proposed as a solution to some problems. For this reason, the purpose for which an ontology is designed is strongly related to the issues present within a certain community. Automatic problem solving information systems is facilitated through the use of ontologies.

Ontologies are domain knowledge representations. The level of detail represented by the ontology depends mainly on the application for which the ontology is being developed. For some applications, it may be necessary to capture a small part of the domain but in greater detail. Other ontologies may cover a large portion of domain knowledge not including detailed descriptions. As each knowledge domain is being constantly researched and new knowledge is continuously emerging, a huge ontology may be needed to cover all the knowledge within a domain. Due to disciplinary interconnections, one knowledge domain may be closely related to another knowledge domain.

Even though the ontology needs to be designed with a specific application in mind, ontology dependence on this application needs to be kept minimal. Separation of the design of the ontology base from the design of ontology commitments may support this objective. Here, the ontology base will remain application independent and the application dependency will be incorporated in the ontology commitment layer. This makes it possible for the different applications within the same domain to use the same ontology base and include the variations in the ontology commitment layer.

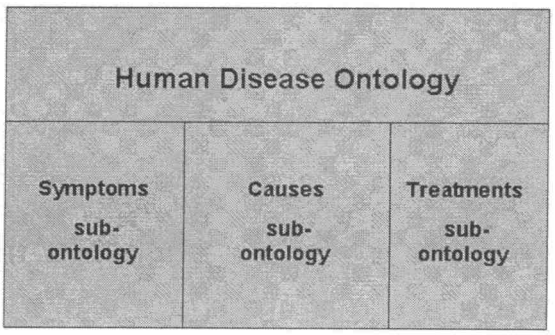

Fig. 2. Human Disease Ontology and its three sub-ontologies

As an example, consider the task of constructing an intelligent retrieval system for the information about human diseases. This is the ontology application. To define the ontology community, user categories will be considered. The two main user categories in this domain may be medical researchers who are mainly interested in the *causes* of a disease, and physicians and patients who are faced with a situation of a disease and are mainly interested in *symptoms* and *treatments* of a disease. Once the purpose of the ontology is revealed, a decision can be made on how to represent the domain knowledge. *Human Disease Ontology (HDO)*, that represents the general knowledge regarding human diseases, can be represented

through three subontologies: Causes, Symptoms and Treatments (shown in Figure 2). Each of these subontologies will be developed using the concepts from the existing ontologies, where possible. In the next section we discuss aligning and merging of existing ontologies.

3.2 Aligning and Merging Ontologies

The goal of this step of the ontology design process is to:

- identify available ontologies suitable for reuse
- choose a suitable ontology merging and alignment tool
- import the source ontologies
- identify correspondence among these ontologies
- align and/or merge ontologies

It is possible to build a new ontology by reusing and adapting terms from other ontologies. An ontology is designed to be agreed upon and shared within a community. It would be contradictory to the ontology definition if each domain application would use its own ontology. Rather, uniformity across the existing ontologies needs to be achieved. Researchers in the ontology-design field have developed different ontologies for various knowledge domains. A large number of the existing ontologies cover overlapping domains and/or are expressing the same domain in different ways. Many ontologies are already available in electronic format in the library of ontologies. Rather than creating a completely new ontology, concepts from the existing ontologies can be used where available.

There are many tools available for ontology merging and alignment such as Chimaera [14], FCA-Merge [15], PROMPT [16]. Each tool has some advantages and disadvantages. The tool characteristics together with the nature of the application need to be analyzed carefully in order to identify the most appropriate tool for the task at hand.

The available ontologies are imported into a new ontological development environment, modified and adapted to be reused by another application. A set of sharable and reusable ontologies can be used during this process.

It is important to recognize correspondences among the source ontologies. Overlapping concepts, concepts with similar meaning but different names e.g. synonyms and concepts unique to each of the source ontologies need to be identified. This work must be done in both ontology merging and ontology alignment [16].

In ontology alignment, the sources are made consistent and coherent with one another but are kept separate. Links are established between the original source ontologies. This allows the ontology designer to compare different knowledge representations and choose the most appropriate one for its own purpose. Alignment is usually performed in situations where the source ontologies cover complementary domains.

Ontology merging results in a single coherent ontology that includes the information from all the source ontologies. The process of ontology merging takes as input two (or more) source ontologies and returns a single ontology which is a merged version of the original ontologies.

LinkBase ontology [17] is the largest medical ontology that consists of 1.5 million language-independent medical and general-purpose concepts. For the example shown in Figure 2, a part of the LinkBase ontology that can be aligned according to the three subontologies can be used to design the HDO ontology. If all the required information cannot be represented by the concepts of the LinkBase ontology, UMLS [18] or other medical ontologies can be used until the HDO ontology content is completely developed.

Usually, the ontology is firstly represented in an informal way. This enables the ontology designers and engineers to capture, discuss and agree upon the preliminary ideas. Following this, the ontology is represented in a formal way which enables the ontology to be used by computers.

Different formalization approaches exist and the purpose for which ontology is built will determine the choice. For some purposes, an ontology may be formalized by a hierarchy of concepts. Other purposes may require more expressive mechanisms to formalize the same vocabulary. Regardless of the ontology comprehensiveness and expressiveness, the formalism for specifying a conceptualization should be kept as simple as possible.

We adopted the two layer approach for formalizing ontologies [3, 20]: specifying of ontology conceptualization and specifying of ontology commitments. We firstly define a set of (binary, even) conceptual relationships. The other domain knowledge and its formal semantics are specified in the commitment layer.

3.3 Formal Specification of Conceptualization

When specifying ontology conceptualization, the goal is to define:

- ontology concepts
- relationships between these concepts
- lexons
- relationships between different lexons
- groups of related lexons

Concept is a unit of thought and image formed by generalization. Term is a lexical representation of a concept (or name of a concept). Reaching agreement for the definition of a concept is a difficult task. Every concept definition is dependent on the definitions of other concepts. In most cases, it is necessary to rely on a commonly accepted understanding of some basic terms.

Relationships between the ontology concepts also need to be precisely defined. This step involves identifying intentional and extensional relations [19]. Intentional relations (conceptual relations) map every concept from the domain to the ontology structure. The ontology structure consists of extensional relations between the chosen domain concepts representing the domain knowledge that we want to describe. The purpose and scope of the ontology will determine the domain concepts that will be mapped by the intentional relations and the kind of extensional relations that will exist among the mapped concepts.

Correct contextual identification of the concepts and the associated relationships must be possible. In our ontology design methodology, we adopted the following definition of an ontology base [20]. An ontology base is:

- A set of context-specific binary fact types, called lexons (notation: $<\mu, t1, r, cr, t2>$). Here is $\mu \in \Delta$ an abstract context identifier chosen from a set Δ; t1, r, cr, t2 are term1, role, co-role and term2 respectively. The lexical terms (t1, r, cr, t2) are constructed from a given alphabet.
- For each $\mu \in \Delta$ and each term t occurring in a lexon, the pair (μ, t) specifies exactly an unique concept.

Lexons express a binary conceptual relationship that is agreed to be true within a given context and among all the ontology community members. The binary nature of such relationships enables modeling of the most basic and atomic fact within a particular domain.

Different ontology lexons may be related to each other. It is possible to treat a lexon as a term in another lexon. In this way, semantic relationships of a higher order than binary relationships are introduced.

Context identifiers ($\mu \in \Delta$) group related lexons in an intended conceptualization of a domain. A context can be defined as a mapping from Δ to a collection of sources, for example a corpus of publications on the same topic. In some cases, context identifiers may group related lexons into a subontology [27].

As an example, we take the following lexon: *<human disease symptoms, disease, has, is of, symptom>* of the form $<\mu, t1, r, cr, t2>$. In this case, 'human disease symptoms' is context identifier μ, 'disease' is term t1, 'symptom' is term t2, 'has' is role r, and 'is of' is co-role cr. This means that in the context of human disease symptoms, 'disease has symptom' and 'symptom is of disease'. For the same context, another lexon may be *<human disease symptoms, symptom ,is, is a, headache>*. This means that 'symptom is headache' and 'headache is a symptom'. Those two lexons are related and belong to the same subontology, the Symptoms subontology of Human Disease Ontology (HDO).

All lexons in the ontology base are free of any specific interpretation. All rules and constraints implied on the lexons are captured in the commitment layer.

3.4 Specifying of Ontology Commitments

The commitment layer consists of a set of ontological commitments. The ontological commitment is expressed through rules and axioms in a given syntax and provides a specific interpretation to a subset of lexons in the ontology base. The commitment layer is concerned with the grounding, instantiation, logically connection, composition and constraining of lexons. Namely, the commitment layer adds the semantics to the model.

In this step, the goal is to:

- identify intra- and inter-commitments
- formalize the commitments
- identify reusable knowledge components

Ontology commitments need to be identified and precisely defined. We differentiate intra- commitments and inter-commitments. A selected subset of lexons contained within the ontology base is con-

strained and attributed with intra-commitments [20]. The elements of this subset of lexons are linked to elements of specific applications with inter-commitments [21].

Ontology commitments can be formalized in two ways. Given a lexon <μ, t1, r, cr, t2>, its commitment can take the form of two perspectives: <t1, r, t2> or <t2, cr, t1>. The first element is referred to as the theme, the second as the transition, and the third the rheme [20].

An ontology base with ontological commitments may be seen as a set of reusable knowledge components. Similar applications or agents reuse or inherit ontology components from each other. The ontology as the ontology base plus the ontology commitment, which is the agent's commitment to a part of it, represents the real world of the agents. An application or agent that operates in the domain is able to further extend and specialize the ontology base by adjusting the commitment layer. Cooperatively working agents differ from each other but have minor differences in their ontologies as they are working together towards the same goal. We can use the same ontology base (or part of a common ontology base) in combination with different sets of ontological commitments in order to design different agent's ontologies. This principle is shown in Figure 3.

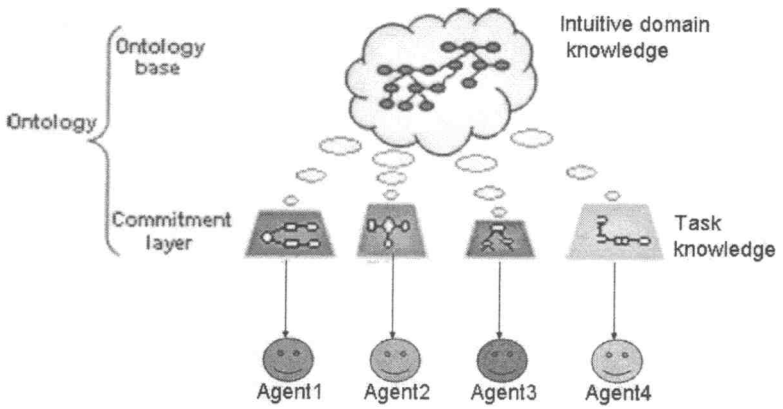

Fig. 3. Different agents share the same ontology base

3.5 Ontology Evaluation

We may consider the ontology evaluation process from different points of view:

(1) technical point of view, assessing the quality of the designed ontology
(2) practical point of view, assessing the usability of the designed ontology
(3) mathematical point of view, evaluating the ontology design through the use of mathematical models.

For the purpose of evaluation of **quality** of the designed ontology, we adopted five criteria suggested by Gruber [22] against which the ontology needs to be evaluated:

- clarity
- coherence
- extendibility
- minimal encoding bias
- minimal ontological commitment

Secondly, we focus on the evaluation of the **usability** of the designed ontology. For this purpose, we focus on the following:

- matching of the originally defined ontology purpose and scope against the designed ontology
- evaluation of conceptual coverage
- evaluation of practical usefulness

Ontology models can be evaluated using mathematical theories. Ontologies can be viewed as sets of related concepts. In most cases, Set Theory is used for constructing mathematical models to define and reason about ontological models [23].

We introduce some level of formality into this discussion by adopting the five criteria suggested by Gruber [21] against which the ontology needs to be evaluated.

Clarity. It needs to be clear why do we chose a particular set of ontology terms and what is the intended meaning of those terms. The number of possible interpretations of a term needs to be restricted. Complete definitions of ontology terms, which are defined by necessary and sufficient conditions, are preferred over partial definitions of ontology terms, which are defined by necessary or sufficient conditions. The definitions need to be objective, documented in natural language but stated in formal axioms within the ontology. Clear definition of ontology terms contribute to the effectiveness of communication between agents.

Within the two layered approach, the ontology base is defined to directly address this point [20].

Coherence. Sentences that can be inferred from the ontology definitions and axioms may not contradict other ontology definitions and axioms. Inferences need to be consistent with the existing definitions and axioms, and should be clear and logical. Improbable or impossible (contradictory) worlds can also be specified especially in the early ontology engineering stages, but practically no applications can commit to them.

Extendibility. As new knowledge emerges each day, this information may need to be added to an already existing ontology. For this reason, the ontology needs to be easily extendable. The new ontology terms should be easily defined based on the existing vocabulary and/or without the need to alter the existing ontology definitions.

The ontology designed by the two-layered approach provides the flexibility in both the ontology base and commitment layer. The binary relationships within the ontology base are easy to add or remove without influencing the other lexons. The same holds for the commitment layer, where new commitments can be easily added or old ones removed.

Minimal encoding bias. Conceptualizations need to be specified at a knowledge-level and not at a symbol-level. The goal of the ontology is to represent true facts, even when it seems to be more convenient to compromise ontology correctness.

Lexons in an ontology base are always true and free of further interpretations. Alternative truths that may emerge during the ontology engineering process must then be provided through additional lexons.

Minimal ontological commitment. More commitments in the ontology make the ontology structure more rigid and limit the number of ontology users. Ontological commitment needs to be minimal but sufficient to support the intended knowledge sharing activities. For this purpose, it may be needed to specify the ontology which is necessary for effective communication consistent with the domain conceptualization and easily extendable by individual agents.

The separation of the ontology base from the ontological commitments allows the design of the minimum ontological commitment sufficient to support the intended knowledge sharing activities. Agents are able then to more easily extend the ontology where needed for their individual purposes.

The ontology's utility and usability can be evaluated from different perspectives:

Matching of the originally defined ontology purpose and scope against the designed ontology. By matching the originally defined purpose and scope of the ontology against the resulting ontology, it is possible to assess how well the originally defined expectations have been met. Some gaps may be found during this process. In this situation, the ontology design processes needs to be continued to meet the originally defined expectations where possible. In other situations, the mismatches may be results of the originally inaccurately defined ontology purpose and scope. This is possible as it may be difficult to have a clear overview of the ontology purposes, goals and requirements at the very beginning of the ontology design process. In this case, the mismatches found need to be clearly explained, defended and well documented.

Evaluation of conceptual coverage. A test set is used to evaluate conceptual coverage of the designed ontology. For example, this test set may consist of 20 abstracts of published papers randomly chosen from publication databases. The developed ontology is used to encode the knowledge from this test set. The evaluation of conceptual coverage includes calculation of the percentage of sentences that can be fully represented by the designed ontology.

Evaluation of practical usefulness. In order to evaluate the practical usefulness of the designed ontology, an application prototype based on the designed ontology needs to be developed, and tested to ascertain the practical feasibility for the given domain.

4 Process II - Ontology-based Multi-agent System Design

The Agent Methodology consists of the following steps:

1. Classify Agents According to Their Responsibilities
2. Identify the Need for an Ontology to Support Agent's Intelligence
3. Define agent's collaborations
4. Construct individual agents
5. Protect the system by implementing security requirements

The five steps of the Agent Methodology are shown in Figure 4. The approaches of the existing methodologies are unified, and some new characteristics are introduced into the multi-agents systems design process.

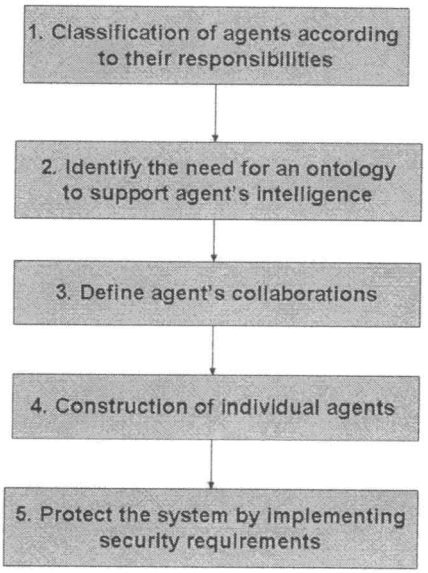

Fig. 4. Multi-agent system design methodology

According to Cassiopeia, a multi-agent system should be designed in terms of agents provided with three levels of behavior: elementary, relational and organizational. Based on the elementary behaviors, the designer defines different roles of agents. We identify different roles of agents in the first stage of the Agent Methodology. In the second stage of the Agent Methodology we focus on making the agents intelligent through use of ontologies. The third phase of the Agent Methodology addresses the dynamics of the organization where relational and organizational behaviors (as defined by Cassiopeia method) are specified. The importance of the organization of agents within a multi-agent system is also emphasized in the Gaia methodology. Here, we define agent's collaborations using an approach similar to BDI methodology. We firstly define the organization of agents within the multi-agent system and then we specify communication routes between different agents. We also emphasize the importance of internal structure of an agent and this is the fourth stage of the Agent Methodology. The fifth stage of the Agent Methodology focuses on the implementation of the security requirements within the system.

4.1 Classification of Agents According to Their Responsibilities

A multi-agent system is a community of agents with different capabilities that complement each other and are cooperatively working towards the same goal. The different agents need to share the overall task, coordinate their actions and integrate their results. When classifying agents according to their responsibilities, it is important to:

- establish intuitive flow of problem solving, task and result sharing
- identify different agent functions needed to establish this kind of flow

- identify different agent roles according to these different functions

As a multi-agent system is composed of cooperatively working agents, an intuitive flow of problem solving, task and result sharing has to be established. In most cases, the overall problem can be decomposed into smaller subproblems so that the tasks can be shared among different agents. The whole problem solving process needs to be systematically followed in our minds from the beginning to the end, and different aspects of solutions to the problems progressively identified.

Corresponding functions required to solve the identified problems have to be clearly defined. It may be necessary to assign exclusive tasks, responsibilities and functions to different agents.

It is common for agents within a multi-agent system to have diverse functions. Different agent roles are identified according to their functions within the system.

For example, the circle may start with the Interface agent that is helping a user in the querying process. Once a query is formulated, the Interface agent sends this query to a Manager agent. According to this query, the Manager agent assigns task(s) to different Information agents. The Information agents search for and retrieve the requested information. The Smart agent analyzes and assembles this information. The resulting assembled information is returned to the user via the Interface agent. In this case, we have four different roles of agents: Interface, Manager, Information and Smart agents. In complex multi-agent systems it may be needed to design various Smart agents so that the results are *assembled* at the different levels of complexity. Other complex multi-agent systems may require different Manager agents that have the ability to *decompose* the problems at the different levels of complexity. One group of agents may be designed for decomposing the problems (and/or assembling the results) on the higher and more generic knowledge level, while the other group of agents may be needed to decompose the problems (and/or assemble the results) on the lower and more detailed knowledge level.

4.2 Identify the Need for an Ontology to Support Agent's Intelligence

The ontology is used to represent the knowledge domain. Use of the ontology enables the multi-agent system to retrieve the information intelligently. The ontology may be used at different stages within a multi-agent system, such as:

- to enable decomposition of the overall problem
- to support the process of information retrieval
- to enable communication between the cooperatively working agents
- to support the process of analyzing and manipulating information
- to enable presentation of the information in a meaningful way

The ontology provides the agents with knowledge of the task structure and this can be used during the process of problem decomposition. In most cases, the overall problem needs to be decomposed into smaller subproblems. This kind of decomposition is usually hierarchical so that the subproblems are further decomposed into smaller sub-subproblems and so on. The goal of the problem decomposition is to reach a stage where the subproblems can be solved by individual Information agents.

The ontology can be used to locate and retrieve requested information. Machine-readable information content within an information resource can be described using an ontology. This provides agents with a more controlled and systematic way to look for the requested information. Use of the ontology enables the agent (that commits to this ontology) to 'understand' the meaning of the information contained within these resources and to exactly locate and retrieve requested information.

The ontology is used to enable cooperatively working agents to communicate with each other during the process of information retrieval. The ontology provides the vocabulary needed for communication. The different agents of a system can reach a shared understanding by committing to the same ontology. Agents are then able to communicate knowledge, make statements and ask queries about a subject domain. Use of the ontology permits coherent communication and easier information sharing between different agents, enabling agents to cooperate and coordinate their actions.

The ontology can be used to analyze and manipulate retrieved information. As the ontology represents the domain knowledge, it allows agents to reason about the retrieved information. Any redundant and/or inconsistent information can be easily recognized and removed. Only relevant information is selected and integrated to be presented to the user.

The ontology can be used to present the retrieved information to the user in a meaningful way. Structured and organized presentation of the requested information becomes possible through the use of ontologies. Moreover, the inherited organization of ontologies adds a taxonomical context to search results, making it easier for the researcher to spot conceptual relationships in data.

As an example, we take the Human Disease Ontology (HDO) shown in Figure 2. HDO represents the knowledge about human disease and is composed of three different subontologies: Disease Symptoms, Disease Causes and Disease Treatments. The overall problem to be solved can be decomposed according to these three subontologies, and further decomposed according to the sub-subontologies. For example, one sub-subontology of Disease Treatments subontology may represent information about Drug Disease Treatments. This hierarchical ontology structure can also be used to present the results in a meaningful way.

4.3 Define Agent's Collaborations

In the first stage of the Agent Methodology, we described how to identify different roles of agents according to their different functions within the multi-agent system. In this stage, we emphasize the importance of structural organization of the agents within a system. These two stages may sound similar, but the difference is that in the first stage we defined activities of agents within a multi-agent system, while in this stage we define structural organization of agents within the multi-agent system. The aim of this step is to:

- organize agents within the multi-agent system so that the tasks are performed in the most efficient way
- establish correspondence between different agent roles and positions of these agents within the multi-agent system

The agents need to be organized in a way that the problem solving process can easily flow into its completion and that the communication between different agents of the system can be established at any point. The overall task needs to be efficiently and effectively performed. Sometimes, a system structured in a simple way functions the best. In other cases, a complex system may be more appropriate for the task at hand.

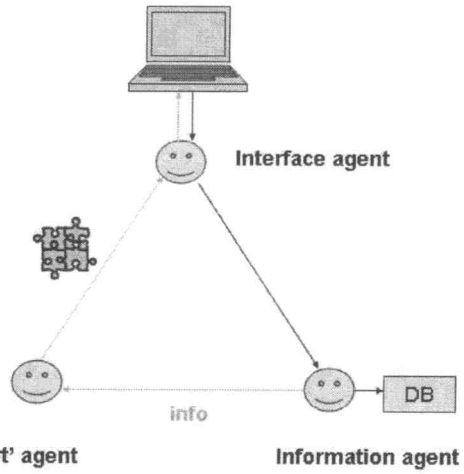

Fig. 5. An example of organization of different agents

It is necessary to establish correspondence between different agent roles (defined in the first stage) and the positions of these agents within the multi-agent system. We do not recommend overloading the agents with responsibilities. It is possible for one agent to correspond to a number of different agent roles. For example, an agent can be a Manager and Smart agent as it may have the responsibility of problem decomposition in the first stages and the responsibility of result assembly in the last stages of the problem solving process. On the other hand, many different agents may correspond to a similar role. This can be the case with Information agents which are situated over different databases. They all have one function of retrieving the information and correspond to the same agent role (Information agents) but are situated over different databases and thus differ in their position within the multi-agent system.

We may have a simple triangle structure such as that shown in Figure 5. The Interface agent helps the user to form queries. Once the query is formulated, the Interface agent sends this query to the Information agent. The Information agent searches and retrieves the requested information. The Smart agent analyzes

and assembles this information. The resulting assembled information is presented back to the user via the Interface agent.

Usually, a multi-agent system is more complex than the one shown in Figure 5. There may be more than one Information agent situated over different databases. The different Information agents may form a hierarchy of agents. There may be more than one Smart agent so that the information can be assembled at different levels of complexity. As a more complex example, we may have a holonic multi-agent system [24, 25]. Holonic multi-agent systems consists of autonomous agents with holonic properties that group together forming holons and holarchies. Such a hierarchically structured multi-agent architecture provides a framework where each agent represents a node in the hierarchic tree. Each agent has the ability to perceive and evaluate changes that occur in the environment. It interacts with other agents in the system in order to reach an optimal decision.

4.4 Construction of Individual Agents

Agents need to be carefully constructed so that they meet the requirements specified by their function and role. In this stage of the Agent Methodology, the aim is to:

- identify required agent components
- assemble the components for the purpose of constructing various agents

Different agent components may include the Human interface, Agent interface, Communication component, Cooperative, Procedural, Task, Domain and Environment knowledge, History files, and so on. The Human interface is needed in agents that interact with users e.g. in Interface agents. The Agent interface enables agents of the system to interact and communicate with each other. The Communication component processes the messages which can be written in an agent communication language and ontology. All agents within the system need to agree and commit to this common language and ontology. The Cooperative knowledge is required by an agent in order to negotiate with other agents of the system, and cooperate and coordinate its actions with their actions. The Procedural knowledge contains information about the procedures involved in the problem solving and goal prioritization method. The Task knowledge is the knowledge regarding tasks assigned to that particular agent. It enables agents to decompose the task into smaller tasks where needed, and perform atomic actions. The Domain knowledge describes the domain of interest of that particular multi-agent system. The Environment knowledge contains information regarding the environment in which the multi-agent system is situated. This component may, for example, contain the information regarding content and structure of each of the different information resources assigned to different Information agents. The History files contains information about the past experiences of the agent.

The variety of agents within a multi-agent system can be achieved in three different ways:

1. Different components that are used to construct different agents are the same, but the *content* of these components is different for different agents. For example, all the agents within a system may have Task knowledge, Environmental knowledge and Cooperative knowledge components. The component describing task(s) of an agent will differ for different agents as they have different functions within the multi-agent system. As all the agents of the system are not placed into the same environment, the Environmental knowledge components will be different for different agents. Not all agents will cooperate in the same way because they have different functions and are situated in different environments. The Cooperative knowledge module will also be different for different agents.
2. The content of the components used to construct different agents are the same, but different agents are constructed by a different *combination* of used components. For example, the first agent may have only a Human interface and Cooperative knowledge component, while the second agent may contain Procedural knowledge, Task knowledge and Cooperative knowledge component. The third agent may have only a History files module, and so on.
3. The third and most common option is that different agents differ in the *combination* of the components used to construct them, *and* in the *content* of these components.

4.5 Protect the System by Implementing Security Requirements

Security plays an important role in the development of multi-agent systems. The risks of jeopardizing the system security must be minimized by providing as much security as possible. The aim of this Agent Methodology stage is to:

- identify critical security requirements within the multi-agent system

- implement the requirements within the system

Security is usually defined in terms of the following five properties [26]:

1. Authentication: proving the identity of an agent.
2. Availability: guaranteeing the accessibility and usability of information and resources to authorized agents.
3. Confidentiality: information is accessible only to authorized agents and inaccessible to others.
4. Non repudiation: confirming the involvement of an agent in a certain communication.
5. Integrity: assuring that the information remains unmodified from source to destination.

The abovementioned properties are critical inside the multi-agent system as well as outside the multi-agent system, such as during the agent interaction with the environment.

After the identification of required security properties, it is necessary to implement them within the multi-agent system. As different agents have different functions within the system, some agents will be more critical than others in regard to the security of the system. As a consequence, the critical agents will be assigned more security requirements than the others.

When considering security of a multi-agent system according to the abovementioned requirements, it is necessary to identify:

- role of each individual agent in the security of the system
- the most critical agents with respect to security
- role of each environmental factor in the security of the system
- the most critical environmental factor with respect to security
- the most critical actions that operate within the system with respect to security
- the most critical actions that operate during the interaction of the system with the environment with respect to security
- the part of the system that is most susceptible to attack from the outside

A designer needs to create a security network from two perspectives:

1. the security of the *individual* agent within the multi-agent system
2. the security of the *whole* multi-agent system

Designers should avoid overloading the agents with security requirements. Sometimes, it is necessary to incorporate new agents into the system that function as Security agents. In other cases, it is enough to modify and adapt existing agent's components or to build additional ones.

For example, in the Section 4.4 we mentioned that the Procedural knowledge of an agent contains information regarding the procedures involved in the problem solving and goal prioritization method. This component may be designed to take two inputs. First input will contain information about the action this agent needs to perform, while the second input will describe how secure is it to performing this action. The first input is provided by the Manager agent, while the second input may be provided by a Security agent or it may have a constant value and be contained in the internal structure of this agent. On the basis of the two inputs, the agent decides whether or not to perform the action.

In the example shown in Figure 5, the whole system operates as a sequence of actions of different agents: Interface agent → Information agent → Smart agent → Interface agent. Interface and Information agents operate inside as well as outside the system and are more critical with respect to security than Smart agents. The Smart agent operates only with the agents within the multi-agent system.

Information agents have access and need to prove their identity to the outside agents of the information resources (authentication). After their identification validation, they are able to access the information resource or a part of it (availability). The integrity property needs to be guaranteed by the agents of the information resources to the Information agents. Namely, a part of the information retrieved from an information resource should not lose its meaning once found outside the context of the information contained within the information resource. Only agents that can prove their authority over the information have access to the information (confidentiality). This is true for inside as well as for outside the system. For example, a Smart agent needs to prove its identity to the Information agents in order to access the information retrieved by the Information agents. Furthermore, Information agents need to prove to the Smart agents that the information they provide is from the correct source (non repudiation).

5 Conclusion and recapitulation

We have proposed Onto-Agent Methodology for the design of ontology-based multi-agent systems as the first methodology that unifies the different approaches of the existing ontology and multi-agent systems

design methodologies. The Onto-Agent Methodology is composed of the two interconnected parts. The initial step consists of designing of an ontology on which a multi-agent system will be based (Onto Methodology). Thereafter, this multi-agent system is designed (Agent Methodology). Each of the two parts is composed of five steps. All the steps were discussed separately and some illustrative examples were provided.

In the first stage of Onto Methodology, the domain of interest is *generalized and conceptualized.* The goal of this step is to clearly identify the knowledge domain, community associated with the ontology design, purpose of the ontology and application(s) based on the designed ontology. In the second stage of Onto Methodology, existing ontologies from the same domain are *aligned and merged* and the existing knowledge reused where possible. Here, it is necessary to identify available ontologies suitable to be reused; choose a suitable ontology merging and alignment tool; import the source ontologies; identify correspondence among these ontologies; and align and/or merge them. It may be necessary to alternate the first two stages of the Onto Methodology until ontology boundaries have been established which correspond with the ontology design objectives. We adopted the DOGMA principle of separation of ontology base from commitment layer [18]. In the third stage of Onto Methodology, the resulting conceptualization of the domain is formally specified in the *ontology base.* It is necessary to define ontology concepts, relationships between these concepts, lexons, relationships between different lexons and form groups of related lexons. The layer of *ontology commitments* operates on the ontology base and is defined in the fourth stage of Onto Methodology. When specifying ontology commitments, it is necessary to identify intra-commitments; identify inter-commitments; formalize the commitments; and identify reusable knowledge components. In the fifth stage of Onto Methodology, the resulting ontology is *evaluated* from technical; practical; and mathematical perspectives. From the technical point of view, ontology is evaluated for its clarity; coherence; extendibility; minimal encoding bias; and minimal ontological commitment. From the practical point of view, originally stated ontology purpose and scope is matched against the designed ontology; conceptual coverage is evaluated; and practical usefulness is evaluated. For the evaluation from the mathematical point of view, Set Theory is predominantly used.

In the first stage of Agent Methodology, different agents are *classified* according to their responsibilities. Here, it is important to define the intuitive flow of problem solving, task and result sharing; identify different agent functions needed to establish this kind of flow; and identify different agent roles according to these different functions. In the second stage of Agent Methodology, we identify the *need for ontology* to support the agents' intelligence. Ontology may be used for different purposes within a multi-agent system such as during the process of problem decomposition; information retrieval; analyzing and manipulating information; enabling communication between cooperatively working agents; and presenting of information in a meaningful way. In the third stage of Agent Methodology, we define agents' *collaborations.* The way that the different agents are organized within the system needs to be defined in such way that the system operates at optimum efficiency. Here, it is also important to establish correspondence between different agent roles defined in the first stage and the positions of agents within the multi-agent system. In the fourth stage of Agent Methodology, we construct *individual agents.* Each agent and its components need to be carefully constructed to satisfy the defined objectives. Agent's components may include Human interface, Agent interface, Communication component, Cooperative knowledge, Procedural knowledge, Task knowledge, History files etc. In the fifth phase of Agent Methodology, we protect the system by implementing *security requirements* of authentication; availability; confidentiality; non repudiation; and integrity [24]. The security of individual agents as well as of the complete multi-agent system needs to be taken into account when designing multi-agent systems.

This paper can provide a more concise understanding of ontology-based multi-agent systems and their design, and increase the control over the design process. This methodology is to be improved and refined as more experience is gained with the design of such systems.

Use of ontologies to make agent systems intelligent is the most important reason for importing ontologies into the computer and information society. Various ontologies describing different knowledge domains are yet to be used by some application. We believe that our application-driven approach to the ontology and agent design methodology will increase the usage of existing ontologies, and effectiveness and efficiency of the designed multi-agent systems.

References

1. Wooldridge, M (2002). *An Introduction to Multiagent Systems*. John Wiley and Sons.
2. Uschold, M, Gruninger, M (1996). Ontologies: principles, methods, and applications. *Knowledge Engineering Review*, **2** (11) 93-155.
3. De Bo, J, Spyns, P, Meersman, R (2003). *Creating a "DOGMAtic" multilingual ontology infrastructure to support a semantic portal*. On the Move to Meaningful Internet Systems Workshop (OTM'03), Lecture Notes in Computer Sciences **2889**, 253–266.
4. Gruninger, M, Fox, Ms (1995).*Methodology for the design and evaluation of ontologies*., Workshop on Basic Ontological Issues in Knowledge Sharing (IJCAI'95).
5. Fernandez, M, Gomez-Perez, A, Juristo, N (1997). *METHONTOLOGY: From ontological Art towards ontological engineering*. Spring Symposium Series on Ontological Engineering (AAAI'97), 33-40.
6. Kinny, D, Georgeff, M, Rao, A (1996). *A methodology and modeling technique for systems of BDI agents*. Seventh European workshop on modeling autonomous agents in a multi-agent world (MAAMAW96), Lecture Notes in Artificial Intelligence **1038**, 56-71.
7. Odell, J, Van Dyke, HP, Bauer, B (2001). Representing Agent Interaction Protocols in UML. *Agent-Oriented Software Engineering*, 121–140.
8. Luck, M, Griffiths, N, d'Invernoy, M (1997). *From Agent Theory to Agent Construction: A Case Study*. Third International Workshop on Agent Theories, Architectures and Languages, Lecture Notes in Artificial Intelligence **1193**, 49-63.
9. Brazier, F, Keplicz, BD, Jennings, NR, Treur, J (1995). *Formal Specification of Multi-Agent Systems: a Real-World Case*. First International Conference on Multi-Agent Systems (ICMAS'95), 103-112.
10. Collinot, A, Drogoul, A, Benhamou, P (1996). *Agent Oriented Design of a Soccer Robot Team*. First International Conference on Multi-Agent Systems (ICMAS'96), 41-47.
11. Abramov, V, Szirbik, N, Goossenaerts, J, Marwala, T, De Wilde, P, Correia, L, Mariano, P, Ribeiro, R (2001). *Ontological Basis for Open Distributed Multi-Agent System*. Symposium on Adaptive Agents and Multi-Agent Systems, 33-43.
12. Girardi, R, Gomes de Faria, C, Balby, L (2004). *Ontology-based Domain Modeling of Multi-Agent Systems*. Third International Workshop on Agent-Oriented Methodologies.
13. Stevens, R, Baker, P, Bechhofer, S, Ng, G, Jacoby, A, Paton, NW, Goble, CA, Brass, A (2002). TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources. *Bioinformatics*, **16** (2) 184-186.
14. McGuinness, DL, Fikes, R, Rice, J, Wilder, S (2000). *An Environment for Merging and Testing Large Ontologies*. Seventh international conference on Principles of Knowledge Representation and Reasoning (KR2000).
15. Stumme, G, Maedche, A (2001). *Ontology Merging for Federated Ontologies on the Semantic Web*. Workshop on Ontologies and Information Sharing (IJCAI'01), 91-99.
16. Noy, NF, Musen, MA (2000). *PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment*. Seventeenth national conference on Artificial Intelligence (AAAI-2000), 450-455.
17. Montyne, F (2001). *The importance of formal ontologies: a case study in occupational health*. International workshop on Open Enterprise Solutions: Systems, Experiences, and Organizations (OES-SEO2001).
18. Bodenreider, O (2004). The Unified Medical language System (UMLS): integrating biomedical terminology. *Nucleic Acids Res*, **32** (1) 267-270.
19. Guarino, N (1998). *Formal Ontology in Information Systems*. Conference on Formal Ontology in Information Systems (FOIS'98), 3-15.
20. Jarrar, M, Meersman, R (2002). *Formal Ontology Engineering in the DOGMA Approach*. CoopIS/DOA/ODBASE 2002, Lecture Notes in Computer Science **2519**, 1238–1254.
21. Deray, T, Verheyden, P (2003). *Towards a semantic integration of medical relational databases by using ontologies: a case study*. On the Move to Meaningful Internet Systems 2003 Workshop (OTM'03), Lecture Notes in Computer Sciences **2889**, 137–150.
22. Gruber, T (1995). Towards Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human and Computer Studies*, **43** (5-6) 907-928.
23. Wouters, C, Dillon, TS, Rahayu, JW, Chang, E, Meersman, R (2004). *Ontologies on the MOVE*. Ninth International Conference on Database Systems for Advances Applications (DASFAA 2004), 812-823.
24. Ulieru, M (2003). Internet-Enabled Soft Computing Holarchies for e-Health Applications. *New Directions in Enhancing the Power of the Internet*, 131-166.
25. Unland, R (2003). *A Holonic Multi-agent System for Robust, Flexible, and Reliable Medical Diagnosis*. On The Move to Meaningful Internet Systems Workshop (OTM'03), 1017-1030.
26. Mouratidis, H, Giorgini, P, Manson, GA (2003). *Modelling secure multi-agent systems*. Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2003), 859-866.
27. Wouters, C, Dillon, T, Rahayu, W, Chang, E (2002). *A Practical Walkthrough of the Ontology Derivation Rules*. 13th International Conference on Database and Expert Systems Applications (DEXA 2002), Lecture Notes in Computer Sciences **2453**, 73-108.