DE GRUYTER OPEN

Information Technology and Management Science

doi: 10.1515/itms-2014-0008

2014 / 17

Knowledge-based System Design Based on Generic Method Conception

Henrihs Gorskis [1], Arkady Borisov [2], [1, 2] *Riga Technical University*

Abstract – **This paper proposes an approach to generic object concepts for problem solving method ontologies and knowledge-based systems. By using placeholders, whose main purpose is to define characteristics of the objects upon which the method acts without specifying a domain, method ontologies can become more reusable. The re-usability stems from the ability to merge such method ontologies more easily with fitting domains. The generic object concepts in the method are proposed to be used as a bridge to the domain ontology. The paper provides a comparison to similar methods in related studies.**

Keywords – **Domain ontology, intelligent system, problem solving method, reusable components.**

I. Introduction

An intelligent system based on storing knowledge in the form of ontologies requires a way of using that knowledge for the purpose of solving a problem. This solution is a problem solving method. A system can be required to solve several different problems using a number of problem solving methods. This paper tries to define a way of reusing existing problem solving methods with various domain descriptions given in the form of ontology models. The main idea is to provide a detailed description of a system in which it would be possible to use several different domain ontology models. Each model would contain a different description of a domain. The system would provide a way of combining almost any domain ontology with a method ontology model. In this paper, we try to define what information would be necessary in the given ontologies and what information it would be possible to obtain from such a combination of two models. For the purpose of reuse, it is necessary to use both ontology models as distinct and different models in the intelligent system. However, during the steps of solving a given problem, these two models would be used as one. The domain ontology is a model that describes all concepts of a certain domain. The method ontology in its turn describes all concepts required for the execution of the method. The method ontology should be as far removed from any domain specific knowledge as possible and the domain ontology should be self-sufficient and descriptive.

II. Related Work

There are several papers that propose similar systems and approaches of using methods and problem solutions with knowledge from an ontology. This paper was heavily influenced by another paper – “Reusable Ontologies, Knowledge-Acquisition Tools, and Performance Systems: Protégé-II Solutions to Sisyphus-2”. The paper describes a knowledge-based system that configures elevators and uses the Protégé-II architecture for this purpose. The paper proposes the use of both application ontology and mapping rules [1]. In doing so, it created the third ontology, which is specific to both the domain and the method ontology. It also keeps track of the mapping that was used in creating the ontology. Even though the paper provides an almost finished framework for a knowledge system that provides the ability for reuse, there is still room for improvement. This paper proposes certain alternatives to the approaches presented in the mentioned paper.

The proposal of separating general domain knowledge from specific tasks and the difficulties of doing this are not new [2]. Modular approaches to ontology building, which would allow for reuse, are also proposed [3]. Another approach to this problem is to build a new ontology from existing ones [4]. However, there still seem to be no specific solutions for all problems associated with reusing knowledge from ontology models [5], [6].

The main problem is that by changing the context of the problem any domain knowledge can be required to change with it [2]. During creation of a domain ontology, the context of it is the requirement to be descriptive. When a specific problem needs to be solved, the context changes and another structure for the ontology may be more fitting. This can be achieved by mapping knowledge from several ontology models [7].

The approach described in the clinical context based flexible workflow (CONFlexFlow) showed how to integrate information about pathways into an ontology-based system [8]. It also provides an approach of using method specific knowledge in the form of Jess rules with the domain ontology. However, the given approach is very reliant on the existing information from the domain ontology and, therefore, less reusable with other domain ontologies.

The paper “A Method-Description Language” provides a language that is capable of describing method ontologies and can be used for creating a “reuse library” [9]. The ideas in that paper have similarities to the approach proposed in this paper, especially the notion of creating and using a library that contains solutions to problems. The described language is useful for mapping ontologies to each other, by providing Meta information for the mapping process.

III. Domain Ontology

The domain otology is used as storage for domain knowledge. It holds descriptions given for concepts that are important in the domain. By describing the concepts of the domain, the relationships and properties between them, the description of the domain itself is also obtained. The domain ontology is usually created by a domain expert. Using the structure of the concepts and by defining restraints and other relationships, the expert creates a framework for the domain knowledge. Once the basis for the ontology is created, the expert fills the ontology with instances, which are the domain knowledge. The class model structure is a meta-layer for the knowledge, while the instances are the knowledge. The domain ontology usually features the description of objects that are important in the domain. For example, forest ontology will have concepts for trees and other vegetation, which can be separated into several different types. The domain ontology describes the domain in such a way that it is possible to obtain knowledge about it. Fig. 1 shows an abstract illustration of the domain ontology. The concepts and relationships are important parts of the knowledge about the domain. However, the domain ontology itself is not designed for solving or finding any solutions to problems. This is different from the method ontology, which provides a description of action.

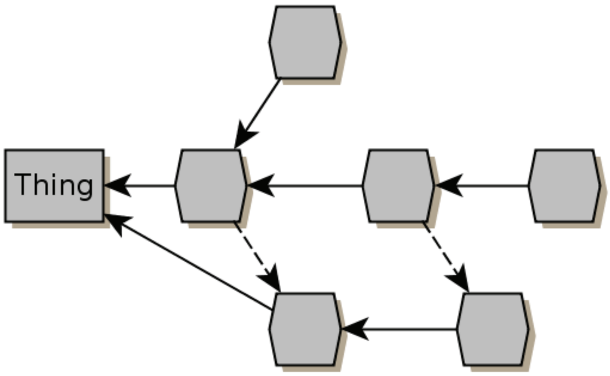

Fig. 1. Generalized visualization of a domain ontology.

IV. Method Ontology

The method ontology differs from the usual domain ontology. Instead of providing a description of a certain domain, that can be associated with any real world thing, it provides information about a method or action. The method ontology contains concepts, which are important for executing the provided method. Very important parts of the method ontology are concepts that describe the input and output of the method. The structure and properties of these classes provide the information necessary for preparing the method and provide information about the outgoing information that is created as a result of the execution of the method. Additionally, the method ontology contains concepts of things necessary during the execution. Those can be descriptions of variables and smaller actions, which are required for keeping track of changes. The method ontology describes these concepts in an abstract way, which is as far removed as possible from any specific domain. The concepts it describes need to be able to transmit all the information necessary for using these concepts. This includes the hierarchy, properties and restrictions of any concepts that are taken from domain ontology for the specific purpose of using them with the method.

Fig. 2 shows an abstract visualization of a method ontology.

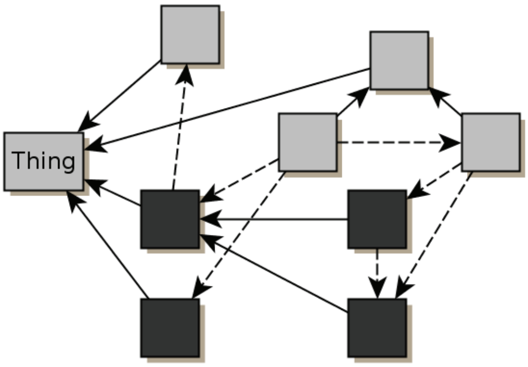

Fig. 2. Generalized visualization of a method ontology.

Note that in Fig. 2 the concepts of the method ontology are separated into two kinds. The light gray blocks symbolize concepts that are internal. They are used to describe actions and other objects and processes that are only used in the description of the method and are in no way connected to any other domains. The dark blocks symbolize objects of the method. The objects of the method are concepts the method acts upon. Their description is important since it is necessary to explain the hierarchy and properties of the things that are being used in the method. However, these objects should be described only in abstract terms. They will later be used for merging with a domain ontology that holds the information of actual concepts.

The method ontology is provided in addition to the method body. The method body is the description of the method itself. In contrast to the method ontology, it does not provide knowledge about the method, but is the pure actions of the method. The method body can be, for example, given as commands in a programming language. It can also be a general description of the actions and the order of the actions that have to be taken to solve a specific problem. Without the method ontology, the method body would not be understandable, and without the method body, the method ontology would provide a description of a method, but could not provide the order of actions.

V. Generic Object Concepts

A method is a description of actions, which are performed on some object or thing. For example, a sorting method describes how to put differently sized objects into order. A general description of such a method does not care about the object, which is being sorted. However, for the sorting method to work, the object which is sorted has to have a size, which can be compared, in order to determine the position of the object in the list of sorted objects. The method ontology can, therefore, provide a general description of an object that is being sorted. This object has to have a property, which tells the size of the object. The method ontology can also provide several descriptions and a hierarchy of these objects. For example, a method ontology that describes the process of sorting some objects can provide the following information:

- The method sorts objects;
- These objects have a property or slot that provides information about the size of the object;
- There are small objects, which are a sub-concept of object and have a small size;
- There are medium objects, which are a sub-concept of object and have a medium size;
- There are large objects, which are a sub-concept of object and have a large size.

If the ontology provides a hierarchy of sub-objects, it can use it for its explanatory powers. A sub-object can be used in the description just like any other object, but it provides its unique properties. However, from the information above, we can see that such a description needs to be finished before it can be used. A generally described method will not provide fixed information about what size objects are to be considered small or large. This information has to be provided at the time, when a domain ontology is connected to the method. Different domains will have different ideas about what size objects are small or large.

Once the objects that are used in the method are defined, the method can provide a general explanation of how it uses these objects, what information is required at the beginning of the method, and what information is returned by the end of the method execution. The inputs and outputs of the method can also be generic objects, but they do not have to be. Let us take a look at an example. Let the example use an ontology that describes different foods. Food ontologies are common and are also used in medicine [10], [11] for the purpose of describing the different kinds and properties of food a person may consume. The example contains:

- The method ontology, which describes a sorting method. It has its generic object structure of small, medium and large generic objects that are used in the sorting;
- The domain ontology, which describes the food domain. It contains concepts like “Nut”, “Fruit”, “Bread” and others. These concepts have individuals like “Walnut”, “Hazelnut”, “Apple”, “Pear”, “White Bread” and others.

Concepts from the food ontology have to be mapped to the generic objects of the method ontology. This can be done by providing the information about the sizes of the foods and the category sizes. Or the information about the sizes can be skipped (as long as that information is not vital to the method itself) and the concepts can be mapped directly by the knowledge expert. The sorting method has to provide the description of the input. In this case, the method input is a list of generic objects (or sub-concepts of the generic object since they can be viewed as the same). From this it is clear that a list of food is the input for the method. The method sorts the objects by their size and returns a sorted list of the same food as the result of the method.

Mapping domain object to these generic method objects should be easier if the method ontology can provide a clear description of what is expected from the concepts that are mapped to the generic objects.

VI. The Relation Between the Generic Object Concepts and the Domain Concepts

The mapping and usage of these generic objects can be simplified if a new kind of relation is introduced to the ontology description of the concepts. This relation is “of functional equality”. This relation would signify to any user or process that uses the ontology description of a concept that any function, action or other process has to perform the same action with the related concept. If we have a relation of functional equality between the concept “apple” and “medium sized object” and a process has to check the size of the concept apple, the result should be that the apple is medium sized, even if that information is provided in the concept “medium sized object” and not in “apple” since both are functionally equal for the purpose of the method. Such an effect can be achieved by using deductive reasoning or by implementing a process of equal functionality (polymorphism). A similar effect can be achieved by using a simple “is a” relation, but that approach would make it necessary to perform additional reasoning, which has to be defined specifically for the connected concepts. Fig. 3 shows a connection between two concepts from two different ontologies, which have to be functionally equal for the purposes of the method.

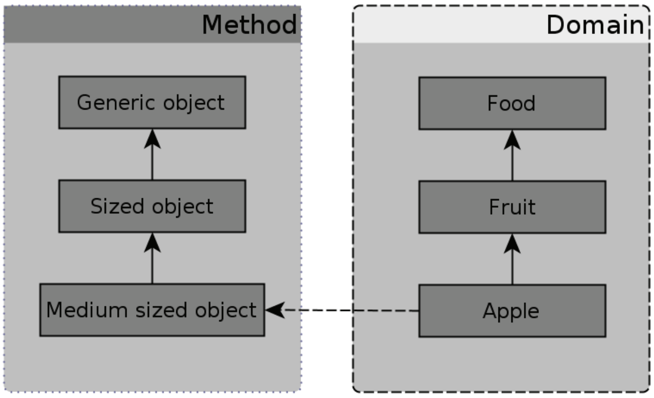

Fig. 3. Relations between ontologies.

VII. Ontology Use within the System

The descriptions of the two kinds of ontologies that are used in the system given in the previous sections are very important. The system is based on the presupposition that the method ontology provides generic objects that can be mapped to the knowledge of the domain ontology. It is based on the process of merging. Merging is important for explanation purposes, so the method is no longer abstract, but describes actions and objects of a certain field. During merging it is necessary to define which abstract method concepts are equivalent to some of the domain concepts. The working of the system can be described with the following steps:

Step 1 – Obtaining the ontologies: The domain ontology is the description of the field, in which the problem arose. The method ontology should be picked based on the problem itself. A reuse library as mentioned in another paper [9] can be very helpful for finding fitting ontologies and making sure that they are compatible.

Step 2 – Mapping: Based on the description of the method ontology, every generic object used by the method has to be mapped to one or several concepts within the domain ontology. Fig. 4 shows how method objects are mapped to domain concepts. Note that in a certain situation it required knowledge be missing from the domain ontology. It can be necessary to add missing information to the ontology.

Step 3 – Creation of the application ontology: Similar to the process described in the related paper, an ontology model that holds all the information for the given task is created [1]. Based on the mapping information and any added knowledge, an application ontology is created. Fig. 5 shows the new ontology that resulted from the combination of the domain and the method ontology. The doted block symbolizes an added concept. The light and dark octagons symbolize the concepts that resulted from the combining method and domain knowledge. It is important to note that these concepts keep all the properties from the concepts they were created from.

Step 4 – Problem solving: Using the application ontology and the method body, it is possible to solve the problem, in the context of the provided domain. This is possible since the concepts used in the method body were connected to the method ontology. By creating the application ontology, any missing information (the inputs) and the context of the problem (the domain) are provided to the method. The actual execution of the method is task for the system. This can be done either by using an API (Application Programming Interface) created for this task, which is capable of looking up the required information from the connected ontology or by generating the executable code by adding the information from the ontology to the method template. An API, for the purpose of using OWL (Ontology Web Language) already exists [12], but it would require additional functionality as described, in order to perform the required actions.

Step 5 – Explanation of the solution: Any solution found by the method can, at this point, be explained further by using the knowledge of the ontologies. If the user of the system requires more information about the solution, the system can provide it, by using connected concepts or by reasoning. Since the solution is described by using the application ontology, the descriptive powers of the ontology can be used.

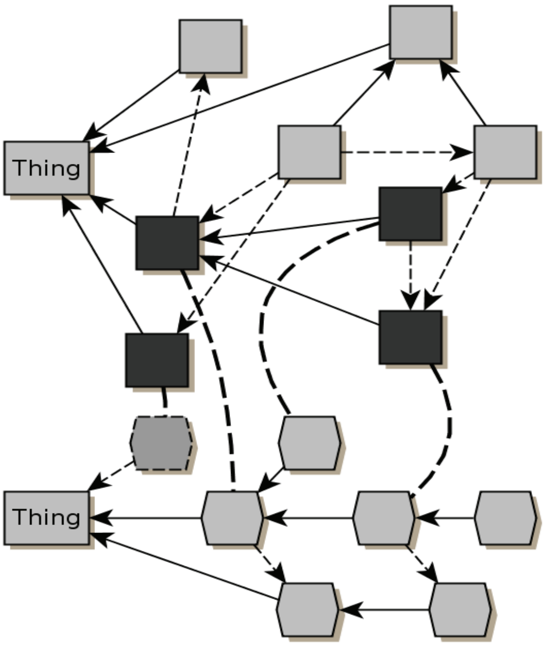

Fig. 4. Mapping the method ontology to the domain ontology.

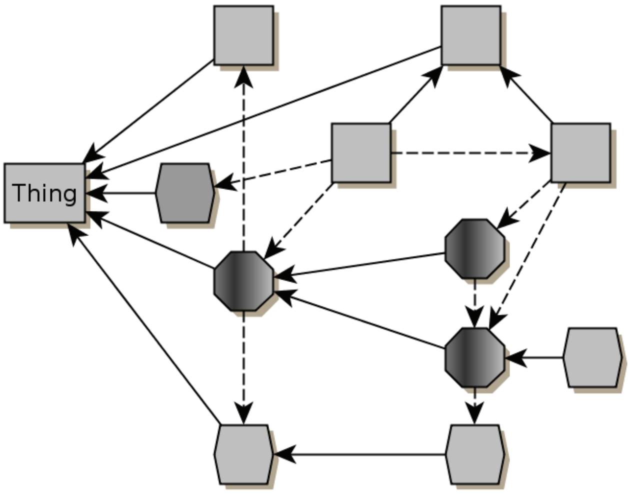

Fig. 5. The application ontology.

The main idea of this approach is the use of method objects in the description of the method ontology. By using these abstract objects, it is easier to define and perform mapping and merging of the two ontology models.

VIII. Conclusion

By combining method and domain ontology, it is possible to reuse the generally defined methods with any fitting domain ontology. This way we can reuse knowledge to solve specific problems and provide explanations for the solutions. The proposed is an approach for a knowledge-based system that can use ontologies in the described way. It is important to note that this approach is heavily based on the way the method ontology is created. The reusability of ontologies is achieved by having method object descriptions in the method ontology.

If the method ontology defines all the concepts it needs in order to perform, the mapping process can be simplified, and the method ontology can become very reusable. These generalized method objects provide the hierarchy and properties, which can be used in the method. Once these objects are combined with domain concepts, the new concepts inherit the hierarchy and properties of both. This way a clear interface for mapping or merging between ontologies is provided. This approach of using generalized non-specific concepts in the description of the method can be applied to any of the methods concepts. It is not limited only to the inputs and outputs of the method. If the method uses variables or objects during execution, which rely or can be described in the domain, these objects can be provided in a generalized way for mapping to the domain ontology.

The application ontology created as a result of combining two different ontologies and adding any required information is distinct from the domain and method ontologies. The reuse comes from using the basic ontologies in different combinations. The application ontology is a specific ontology usable only for solving the task it was created for. Only by creating such application ontologies for different situations, by using preexisting ontologies, reuse of knowledge can be achieved.

The ability to describe the obtained solution from the method comes from the idea that the execution of a method is connected to the ontological description of the concepts used in the method. Once the method ontology is replaced by the application ontology, the method has to be capable of using this new knowledge about the concepts that are used in the method. The generic method objects are also renamed and combined with concepts from the domain. This means that these concepts now contain information from two ontologies usable for solving the task. That means that the execution needs to be capable to look up the properties of concepts at runtime. For example, if a method is designed to sort different sized objects by their size, the application ontology can be created with a domain ontology that contains knowledge about fruit. And a certain fruit was mapped to a certain method related object that provides information about size. The method has to be capable of using the concept that describes the fruit as a sized object. At this point, it can become clear if a fundamental difference exists within these ontologies, which would require further work from the domain export to make such ontologies compatible.

The language used in the method body to describe actions and the exact specifications of the execution are subjects for future work. They use concepts and information from the ontology, but they are not part of the ontology and do not necessarily use the ontology directly.

This paper is part of research in progress to create a specification of an intelligent system that will be capable of reusing ontology knowledge. The approach provided in this paper about reusing methods by creating combinations of different ontologies is part of this research.

REFERENCES

[1] T. E. Rothenfluh, J. H. Gennari, H. Eriksson, A. R. Puerta, S. W. Tu and M. A. Musen, "Reusable Ontologies, Knowledge-Acquisition Tools, and Performance Systems. Protégé-II Solutions to Sisyphus-2" in *International Journal of Human-Computer Studies*. 2002, pp. 303–332. http://dx.doi.org/10.1006/ijhc.1996.0017

[2] B. Chandrasekaran, "Generic Tasks in Knowledge-Based Reasoning: High-Level Building Blocks for Expert System Design," *IEEE Expert*, vol. 1, Issue 3, 1986, pp. 23–30. http://dx.doi.org/10.1109/MEX.1986.4306977

[3] C. Torniai, M. H. Brush, N. Vasilevsky, E. Segerdell, M. Wilson, T. Johnson, K. Corday, C. Shaffer and M. Haendel "Developing an Application Ontology for Biomedical Resource Annotation and Retrieval: Challenges and Lessons Learned," in *ICBO*, 2011.

[4] N. K. Leung, S. K. Lau, J. Fan and N. Tsang, "An integration-oriented ontology development methodology to reuse existing ontologies in an ontology development process," in *Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Service*s. ACM. December 2011, pp. 174–181.

[5] D. Lonsdale, D. W. Embley, Y. Ding, L. Xu and M. Hepp, "Reusing ontologies and language components for ontology generation" in *Data & Knowledge Engineering* , vol. 69, no. 4, 2010, pp. 318–330.

[6] E. Simperl, "Reusing ontologies on the Semantic Web: A feasibility study" in *Data & Knowledge Engineering*, vol. 68, no. 10, 2009, pp. 905–925.

[7] J. J. Jung, "Reusing ontology mappings for query routing in semantic peer-to-peer environment," in *Information Sciences*, vol. 180, no. 17, 2010, pp. 3248–3257. http://dx.doi.org/10.1016/j.ins.2010.04.018

[8] W. Yao, A. Kumar, "CONFlexFlow: Integrating Flexible clinical pathways into clinical decision support systems using context and rules" in *Decision Support Systems*, vol. 55, no. 2, 2013, pp. 499–515. http://dx.doi.org/10.1016/j.dss.2012.10.008

[9] J. H. Gennari, W. Grosso, M. Musen, "A Method-Description Language: An initial ontology with examples". *Stanford Medical Informatics*. 1998, pp. 1–18.

[10] J. Cantais, D. Dominguez, V. Gigante, L. Laera, V. Tamma, "An example of food ontology for diabetes control". *Proceedings of the International Semantic Web Conference 2005 workshop on Ontology Patterns for the Semantic Web*. 2005, pp. 1–9.

[11] J. Graça, M. Mourão, O. Anunciação, P. Monteiro, H. S. Pinto, V. Loureiro, "Ontology building process: The wine domain" in: *Proc. of the 5 th Conf. of EFITA*. 2005, pp. 1138 – 1145.

[12] J. Henß, J. Kleb, S. Grimm, "A Protégé 4 Backend for Native OWL Persistence," *11th Intl. Protégé Conference*, June 23–26, 2009, pp. 1–4.

Henrihs Gorskis is a second-year Doctoral Student majoring in Information Technology at Riga Technical University (RTU). He received his *Mg. sc. ing.* degree in 2013. His research interests include data mining, ontology engineering, evolutionary computing and programming. In 2013 he participated in the AICT conference and also provided a paper on ontology engineering. He is especially fond of the Java programming language and uses it for both work and personal application development.
E-mail: henrihs.gorskis@rtu.lv.

Arkady Borisov received his Doctoral Degree in Technical Cybernetics from Riga Polytechnic Institute in 1970 and *Dr. habil. sc. comp*. degree in Technical Cybernetics from Taganrog State Radio Engineering University in 1986. He is a Professor of Computer Science at the Faculty of Computer Science and Information Technology, Riga Technical University (Latvia). His research interests include fuzzy sets, fuzzy logic and computational intelligence. He has 235 publications in the field.
He has supervised a number of national research grants and participated in the European research project ECLIPS. He is a member of IFSA European Fuzzy System Working Group, Russian Fuzzy System and Soft Computing Association, honorary member of the Scientific Board, member of the Scientific Advisory Board of the Fuzzy Initiative Nordrhein-Westfalen (Dortmund, Germany).
E-mail: arkadijs.borisovs@cs.rtu.lv.

Henrihs Gorskis, Arkadijs Borisovs. Uz zināšanām bāzētas sistēmas izstrāde, lietojot vispārīgās metodes konceptus

Intelektuālās sistēmās var būt nepieciešams lietot vairākas ontoloģijas. Ontoloģijas glabā konceptuālās pamatzināšanas šādās sistēmās: lai atrisinātu dažādus uzdevumus ar dažādām pamatzināšanām, ir nepieciešams lietot dažādas ontoloģijas. Šajā darbā tiek piedāvāta pieeja, kas palīdz atkārtoti lietot dažādas ontoloģijas ar vienu un to pašu sistēmu jeb atrisināšanas metodi. Sistēmā tiek glabāta metodes ontoloģija. Tās ir konceptuālās pamatzināšanas par pielietojamo metodi un to, kādā veidā tiek atrisināts uzdevums. Lai atvieglotu metodes ontoloģiju apvienošanu ar ontoloģijām, kas apraksta problēmas nozari, tiek piedāvāts lietot vispārīgus konceptus par objektiem metodes aprakstā. Šiem konceptiem piemīt visas īpašības, kas ir nepieciešamas, lai metode varētu tos lietot problēmas atrisināšanas gaitā. Šādam aprakstam ir jābūt pēc iespējas vispārīgam un nesaistītam ar trešo problēmas apgabalu. Tajā brīdī, kad metodes ontoloģija tiek apvienota ar problēmas ontoloģiju, šie vispārīgie metodes objekti kalpo par apvienošanas punktiem. Savienojot vispārīgo metodes objektu ar konceptu no problēmas ontoloģijas, tiek izveidota saite ar citiem konceptiem no problēmas nozares. Vienlaikus tiek garantēts, ka saistītiem konceptiem piemīt visas īpašības, kas ir nepieciešamas, lai tos varētu lietot metodes izpildes laikā. Kad metodes un problēmas vides ontoloģijas ir apvienotas, tad izveidojas aplikācijas ontoloģija. Tā ir ontoloģija, kas kalpo par pamatu problēmas risināšanas procesā. Izmantojot apvienoto ontoloģiju, ir iespējams uz tās pielietot ontoloģijas iespējas, jauno zināšanu atklāšanai vai esošo faktu aprakstīšanai. Sakarā ar to, ka vispārīgie metodes koncepti tiek savienoti ar citas ontoloģijas konceptiem, šajā solī var atklāties, ka izvēlētais koncepts ir nederīgs, jo var izveidoties pretrunas ar vienu no ontoloģijām. Tiek piedāvāts lietot šos vispārīgos metodes konceptus, lai atvieglotu ontoloģiju apvienošanas procesu un padarītu ontoloģijas atkārtoti lietojamas. Tādējādi ontoloģiju apvienošanas procesā jau iepriekš ir norādīts veids, pēc kura ir iespējams ātrāk apvienot dažādas ontoloģijas.

Генрих Горский, Аркадий Борисов. Разработка системы, основанной на знаниях, используя общие концепты метода

В интеллектуальных системах иногда необходимо использовать несколько онтологий. В таких системах онтология хранит базовые концептуальные знания. Для решения различных задач с разными базовыми знаниями необходимо использовать целый ряд различных онтологий. В данной работе предлагается подход, который помогает повторно использовать разнообразные онтологии с той же системой или методом решения задачи. В системе хранится онтология метода. Она является концептуальной основой знаний, которые применяются в методе и описывают способ, с помощью которого эта задача решается. Для того чтобы облегчить слияние онтологии метода с онтологиями, описывающими проблему в отраслях, предлагается использовать общие концепты объектов, описанных в методе. Эти концепты имеют все качества, которые необходимы методу, для того чтобы они могли быть использованы в ходе решения задачи. Такое описание должно быть по возможности максимально общим и не связанным с третьей проблемной областью. Когда онтология метода объединяется с онтологией отрасли, эти общие объекты метода служат точками объединения. Объединяя общие объекты метода с концептами из онтологии проблемы, создаётся связь с другими понятиями отрасли. В то же время это гарантирует, что соответствующие понятия будут обладать всеми качествами, которые необходимы для использования во время выполнения метода. Когда онтологии метода и окружающей среды объединяются, создаётся онтология приложения. Эта онтология служит основой для процесса решения проблемы. Используя объединённую онтологию, можно применять функции онтологии для открытия новых знаний или описания существующих фактов. В связи с тем, что общие понятия метода связаны с другими понятиями онтологии, на этом этапе может выясниться, что выбранная концепция является недействительной, так как могут появиться противоречия в одной из онтологий. Предлагается использовать эти общие объекты метода для облегчения процесса слияния онтологий и сделать онтологии повторно используемыми. Таким образом, в процессе слияния онтологий уже существовало указание, по которому можно было бы быстрее комбинировать различные онтологии.