# One Practical Algorithm of Creating Teaching Ontologies

Tatiana Gavrilova[1], Rosta Farzan[2], Peter Brusilovsky[2,3]

[1]Intelligent Computer Technologies Dept, St.-Petersburg State Technical University, Polytechnicheskaya st. 29, 195251 St.-Petersburg, Russia, gavr@fulbrightweb.org

[2] Intelligent System Program, University of Pittsburgh, Pittsburgh PA 15260 USA, rosta@cs.pitt.edu

[3] School of Information Science, University of Pittsburgh, Pittsburgh PA 15260 USA, peterb@mail.sis.pitt.edu

The paper presents one practical approach aimed at developing teaching ontologies. The underlying research framework is pursuing a methodology that will scaffold the process of knowledge structuring and ontology design. Moreover, special stress should be placed on visual design as a powerful learning mind tool. For more comprehensible understanding the process of developing a practical ontology from the domain of introductory C programming is described.

**Keywords**: ontology, visual knowledge engineering, knowledge acquisition, knowledge sharing and reuse, NBE.

## 1 Introduction

The achievements in the field of Artificial Intelligence help to develop a range of ways of symbolic and graphical representing of knowledge. A well-chosen analogy or diagram can make all the difference when trying to communicate a difficult idea to someone, especially a non-expert in the field. The idea of using visual structuring of information to improve the quality of student learning and understanding is not new. Knowledge engineers make use of a number of ways of representing knowledge when acquiring knowledge from experts. These are usually referred to as knowledge models.

Teachers are also knowledge engineers. They are used to work with concept maps, mind maps, brain maps, semantic networks, frames (Conlon 1997), (Jonassen 1998), (Sowa 1984) and other conceptual structures. As such, the visual representation of the general domain concepts facilitates and supports student understanding of both semantic and syntactic knowledge. A teacher operates as a knowledge analyst by making the skeleton of the studied discipline visible and showing the domain's conceptual structure. At the present time, this structure is called an *ontology*. However, ontology-based approaches to teaching are relatively new fertile research areas. They originated in the area of knowledge engineering (Boose 1990), (Eisenstadt et al 1990), (Wielinga &Schreiber 1992), which was then transferred to ontology engineering (Fensel 2001), (Jasper & Uschold 1999), (Mizogushi & Bourdeau 2000).

Knowledge Engineering traditionally emphasized and rapidly developed a range of techniques and tools including knowledge acquisition, conceptual structuring and representation models (Adeli 1994), (Scott & Clayton 1994).

Since 2000 a major interest of researchers focuses on building customized tools that aid in the process of knowledge capture and structuring. This new generation of tools – such as Protégé, OntoEdit, and OilEd - is concerned with visual knowledge mapping that facilitates knowledge sharing and reuse (Protégé 2004), (OntoEdit 2004), (OilEd 2004). The problem of iconic representation has been partially solved by developing knowledge repositories and ontology servers where reusable static domain knowledge is stored. Ontolingua, and Ontobroker are examples of such projects (Ontolingua 2004), (Ontobroker 2004).

This paper proposes a clear, explicit approach to practical ontology design. The underlying research is pursuing usage of the visual, iconic representation, and diagrammatical structures. The special stress is put on visual design as a powerful learning mind tool. For more comprehensible understanding of the process, the process of developing a practical ontology from a course of introductory C programming is described. In the remainder of the paper, we will describe some theoretical issues regarding ontological engineering and present our proposed algorithm for ontology design. Moreover, we will describe our

ISBN XXX-XXXX-XXX-X (pain.) / ISBN XXX-XXX-XXX-X (PDF)
[online http://ktk.ulapland.fi/ISBNXXX-XXX-XXX-X]

detailed practical example following the proposed algorithm. In conclusion, we provide insight into the discussion of the current and possible future work.

## 2 Using ontological engineering for teaching purposes

We start the discussion of theoretical issues of ontological engineering by reviewing different definitions of ontology from literature circulating within the field.

### 2.1 Ontology Definitions

Ontology is a set of distinctions we make in understanding and viewing the world. There are numerous well-known definitions of this milestone term (Neches 1991), (Gruber 1993), (Guarino & Giaretta 1998), (Gomez-Perez 2004), that may be generalized, e.g.:

- ✓ *Ontology is a hierarchically structured set of terms for describing a domain that can be used as a skeletal foundation for a knowledge base.*

Such definition clarifies the ontological approach to knowledge structuring while giving enough freedom to open-ended, creative thinking. For example, ontological engineering can provide a clear representation of a course structure, main terms, methods, and their inter-relationship.

Ontology as a useful structuring tool may greatly enrich the teaching process, providing students an organizing axis to help them mentally mark their visions in the information hyper-space of the domain knowledge.

### 2.2 Creating Teaching Ontologies

Ontology creating also faces the knowledge acquisition bottleneck problem. The ontology developer encounters the additional problem of not having any sufficiently tested and generalized methodologies, which would recommend what activities to perform and at what stage of the ontology development process. An example of this can be seen when each development team usually follows their own set of principles, design criteria, and steps in the ontology development process. The lack of structured guidelines and methods hinders the development of shared and consensual ontologies within and between the teams. Moreover, it makes the extension of a given ontology by others, its reuse in other ontologies, and final applications difficult (Guarino & Giaretta 1998).

Until now, only few effective domain-independent methodological approaches have been reported for building ontologies (Swartout, Patil, Knight& Russ 1997), (Fensel 2001), (Mizogushi 2000). What they have in common is that they start from the identification of the purpose of the ontology and the needs for the domain knowledge acquisition. However, having acquired a significant amount of knowledge, major researchers propose a formal language expressing the idea as a set of intermediate representations and then generating the ontology using translators. These representations bridge the gap between how people see a domain and the languages in which ontologies are formalized. The conceptual models are implicit in the implementation codes. A re-engineering process is usually required to make the conceptual models explicit. Ontological commitments and design criteria are implicit in the ontology code.

**Figure 1** presents our vision of the mainstream state-of-the-art categorization in ontological engineering (Guarino, Welty 2000), (Jasper & Uschold 1999), (Uschold & Ggruninger 1996) and may help the knowledge analyst to figure out what type of ontology he/she really needs. We use Mindmanager™ as it proved to be a powerful visual tool.

Frequently, it is impossible to express teaching information in a single ontology. Accordingly, subject knowledge storage consists of a set of related ontologies. However, some problems may occur when moving from one ontological space to another that could be solved by constructing meta-ontologies that may help to resolve these problems.

We can propose different types of teaching ontologies that can aid effective learning:

- ❖ Main concepts ontology,
- ❖ Historical ontology (genealogy),
- ❖ Partonomy of the discipline,
- ❖ Taxonomy of the theories, methods and techniques, etc.

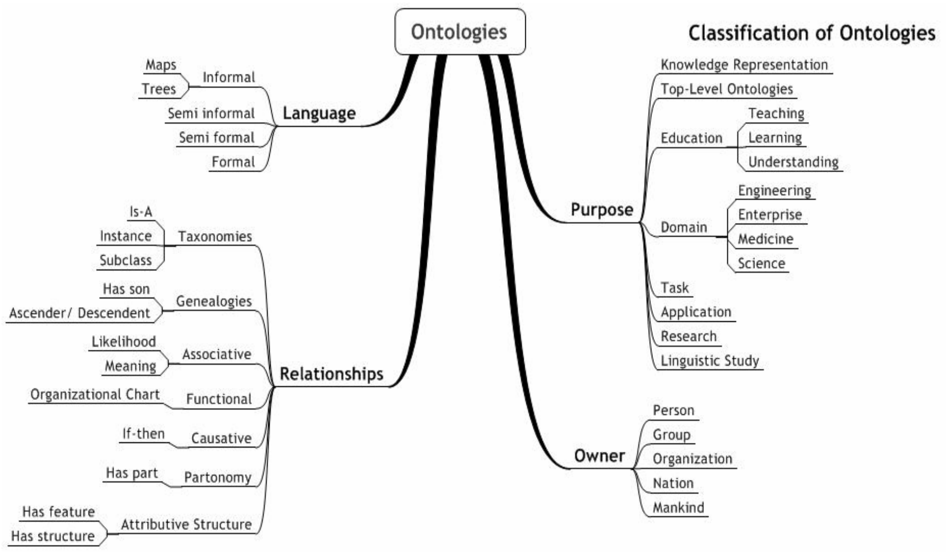

**Figure 1.** Ontology classification

The concrete set of ontologies depends on personal vision, teaching subject and awareness level of the students. Generalizing our experience in developing different teaching ontologies for e-learning in the field of artificial intelligence and neurolinguistics (Gavrilova & Voinov 1996), (Gavrilova & Voinov 1998), (Gavrilova, Voinov, Vasilyeva 1999), (Gavrilova 2003), we propose a five-step algorithm that may be helpful for visual ontology design. We put stress on visual representation as a powerful mind tool (Jonassen 1998) in structuring process. Visual form influences both analyzing and synthesizing procedures in ontology development process. That is why we believe that the "beauty" of the ontology plays an important role in understanding of the knowledge. The following section describes our 5-step recipe of ontology design and some consideration on layout harmony

## 3 Ontology Creating

### 3.1 Five-Steps Recipe

We can propose 5-steps recipe for creating ontology:

1. **Glossary development**: The first step should be devoted to gathering all the information relevant to the described domain. The main goal of this step is selecting and verbalizing all the essential objects and concepts in the domain.
2. **Laddering**: Having all the essential objects and concepts of the domain in hand, the next step is to define the main levels of abstraction. It is also important to elucidate the type of ontology according to **Figure 1** classification, such as taxonomy, partonomy, and genealogy. This is being done at this step since it affects the next stages of the design. Consequently, the high level hierarchies among the concepts should be revealed and the hierarchy should be represented visually on the defined levels.
3. **Disintegration**: the main goal of this step is breaking high level concepts, built in the previous step, into a set of detailed ones where it is needed. This could be done via a top-down strategy trying to break the high level concept from the root of previously built hierarchy.
4. **Categorization**: At this stage, detailed concepts are revealed in a structured hierarchy and the main goal at this stage is generalization via bottom-up structuring strategy. This could be done by associating similar concepts to create meta-concepts from leaves of the aforementioned hierarchy.
5. **Refinement**: The final step is devoted to updating the visual structure by excluding the excessiveness, synonymy, and contradictions. As mentioned before, the main goal of the final step is try to create a beautiful ontology. We believe what makes ontology beautiful is harmony and clarity.

### 3. 2 Harmony as Conceptual balance

A well-balanced ontological hierarchy equals a strong and comprehensible representation of the domain knowledge. However, it is a challenge to formulate the idea of a well-balanced tree. Here we offer some tips to help formulate the "harmony":

1. Concepts of one level should be linked with the parent concept by one type of relationship such as is-a, or has part.
2. The depth of the branches should be more or less equal (±2 nodes).
3. The general outlay should be symmetrical.
4. Cross-links should be avoided as much as possible.

### 3.3 Harmony as Clarity

Moreover, when building a comprehensible ontology it is important to pay attention to clarity. Clarity may be provided through number of concepts and type of the relationships among the concepts. Minimizing the number of concepts is the best tip according to Ockham's razor principle proposed by William of Ockham in the fourteenth century: "Pluralitas non est ponenda sine neccesitate," which translates as "entities should not be multiplied unnecessarily."

The maximal number of branches and the number of levels should follow Miller's magical number (7±2) (Miller 1956).

Furthermore, the type of relationship should be clear and obvious if the name of the relationship is missing. Some tips to achieve visual clarity are described later in section 4.4.

## 4 Developing Practical Ontology

In this section we describe our attempt to develop ontology for C programming language following the aforementioned 5-step algorithm. We have tried to report the exact practical procedures we followed at each step by including all the visual structures.

### 4.1 Step 1 - Glossary Development

As previously mentioned the first step in building ontology is collecting information in the domain and building a glossary of the terms of the domain. To build a glossary for teaching introductory C programming course, we collected the terms from two different types of resources: closed-corpus material and open-corpus material.

The closed corpus materials are in the form of lecture notes that are precisely designed for the course. The open corpus materials include several online tutorials in C programming. We extracted the terms from the lecture notes manually by carefully reviewing the lecture handout. The terms from open-corpus material were extracted automatically (Brusilovsky & Rizzo 2002 ). Consequently, we tried to combine the automatically extracted terms with manually extracted terms to build a single glossary. **Figure 2** presents the combined unsorted glossary.

| | | |
|---|---|---|
| Data type | While loop | User defined function |
| Integer | Statement | Parameter passing |
| Character | Block | By Values |
| Float | Sequential execution | By reference |
| Primitive data type | Condition | Kernel command |
| Expression | Nested if | Function prototype |
| Data structure | If-else | Pointer & Function |
| Array | If | Pointer |
| Array processing | Multiple selections | Computer memory |
| Data aggregate | Switch | Memory address |
| String | Complex conditionals | Declaring pointer |
| Two dimensional arrays | File | Malloc |
| Declaration of two dimensional arrays | Simple file processing | Stack |
| Variable | Advanced file processing | Recursion |
| Variable initialization | IO | Operator |
| Variable assignment | Input | Pointer operation |
| Variable declaration | Standard input | Push |
| Readable program | Scanf | Conditional operator |
| Indentation | Input redirection | Arithmetic operation |
| Understandable program | Standard input | Decrement expression |
| Comments | Output | Pre-decrement |
| Constant | Printf | Post-decrement |
| Define | Printing | Increment expression |
| Literal constant | Standard output | Post-increment |
| Pre-processor | Output redirection | Pre-increment |
| Directive | Conversion | Logical operator |
| C compiler | Explicit conversion | OR operator |
| Loop | Typecasting operator | Not operator |
| Do-while loop | Casting | AND operator |
| Counter controlled loop | Automatic type conversion | Character operator |
| Sentinel controlled loop | Assignment conversion | Relational operator |
| For loop | Safe type conversion | Associativity of operators |
| Nested loop | Unsafe type conversion | Order of calculation |
| Threshold controlled loop | Function | Precedence of operators |
| | Function parameter | |
| | Standard function | |

**Figure 2.** Glossary of the terms for teaching C programming

## 4.2 Step 2 - Laddering: Building an Initial Mind Map Structure

At the second step we built an initial visual structure of the glossary terms. The main goal of this step is the creation of a set of preliminary concepts and the categorization of those terms into concepts. A mind map can be a useful visual structure for this step. **Figure 3** presents the mind map of our initial categorization. Since the categorization in this step is preliminary, some of terms might not fit into any of the initial categorization. We should mention that the categorization in this step is done entirely manually. However, we employed the lecture notes, which were used to build the glossary in the previous step, to build the initial categories as well. We can consider the lecture notes, as expert help, to design the ontology. This is due to the fact that the lecture notes were designed by the expert who is teaching the course for couple of years.

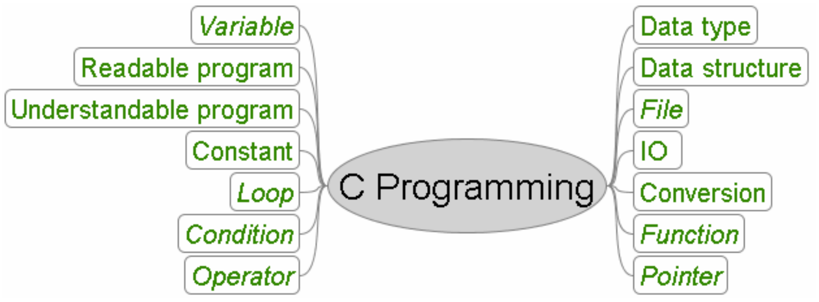

**Figure 3.** Trivial categorization

**Figure 4** presents the details of our initial categorization of the terms into the concept in **Figure 3**. The visual structures presented at this step, illustrates the idea of how an ontology can bridge the gap between the chaos of unstructured data presented in the glossary and be a clear means of showing mapped representations. Another good example of this bridge can be the ontology of Italian artistic schools which is built from chaos of names of great Italian artists (Gavrilova 2003).

### 4.3 Steps 3 & 4 - Disintegration/Categorization: Building a Concept map with more Precise Hierarchy

At next step we composed more precise concepts and hierarchies by analyzing the glossary and previously built visual structure. First we employed the top-down design strategy to create meta-concepts such as “Date”, “Structure”, and “IO”. Then using the bottom-up strategy we tried to fit the terms and concepts into the meta-concept. Moreover, we created the relationships between the concepts. A concept map is the most useful visual structure for representation of the results of this stage, since it gives the ability of defining the relationship in addition to building the hierarchy. The output of this step is a large and detailed map, which covers the course in the hierarchical way[1]. However, since this ontology is designed for teaching purposes it is important to offer the overall picture and a general hierarchy as well.

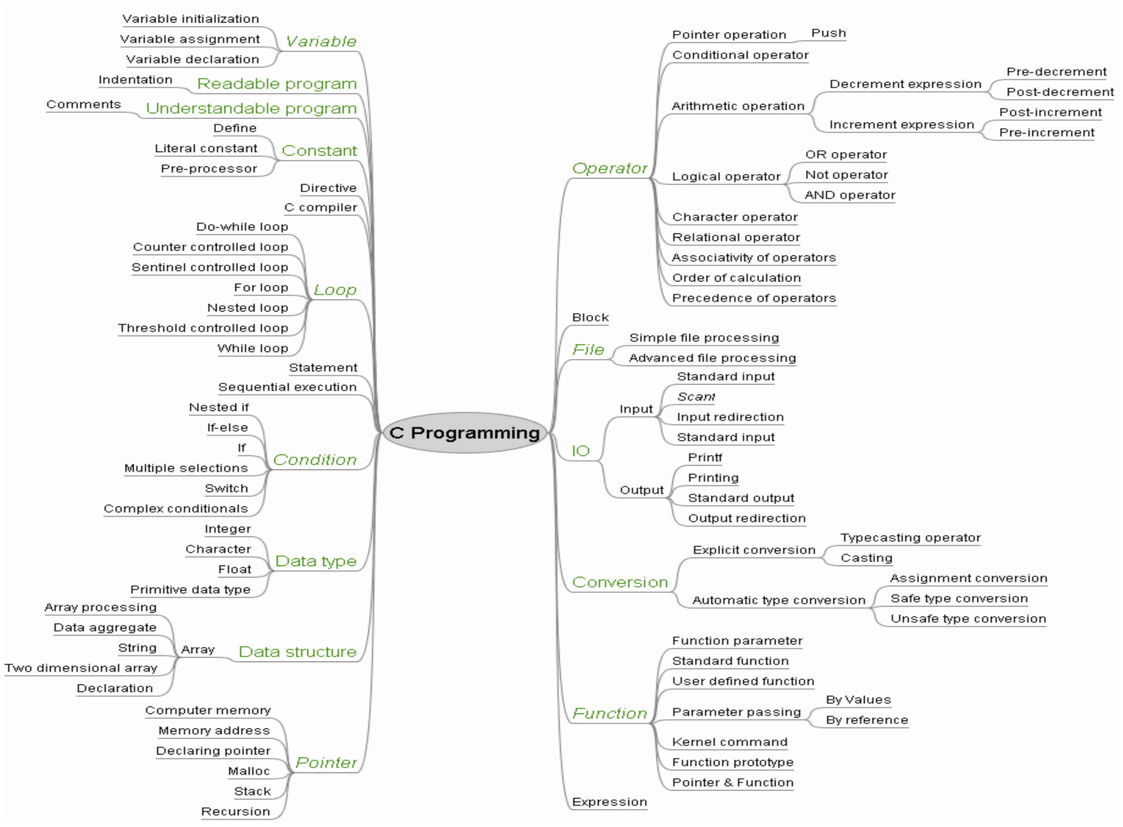

**Figure 4.** Details of first level categorization

Therefore, based on the detailed concept map, we built the general ontology which is shown in **Figure 5**.

[1] Because of the huge size of the concept map it is difficult to include it in the paper

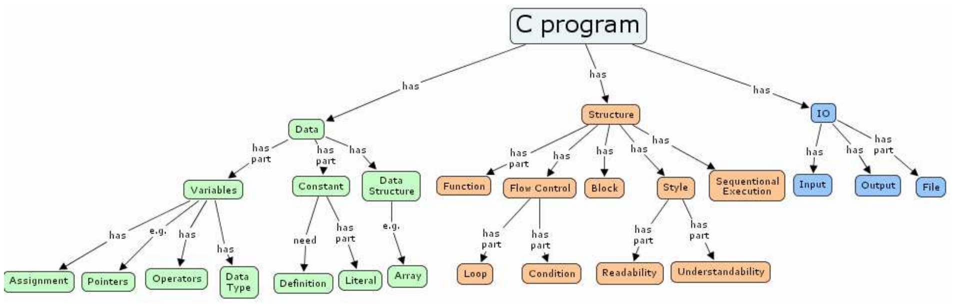

**Figure 5.** General ontology

### 4.4 Step 5: Refinement

As described in the algorithm, the final step is devoted to making the ontology beautiful. The followings are some practical tips that we may be taken into consideration while designing the ontology:

1. Use different font sizes for different strata (as shown in **Figure 3** and **Figure 4**).
2. Use different colors to distinguish particular subsets or branches (as shown in **Figure 5**, not very clear in the black and white printout ).
3. Use a vertical layout of the tree structure/diagram (as shown in **Figure 5**).
4. If needed, use different shapes for different types of nodes.

Moreover, we re-built the general ontology while taking into consideration the harmony and clarity factors. Comparing **Figure 5** and **Figure 6** presents these changes. Consequently, we tried to balance the depth of the branches by adding one more level to the "IO" branch. Another feature of harmony is having the same relationship at each level. Since this is not easy to achieve, we tried to differentiate the level of the nodes based on the relationships in the same depth. For example, all nodes with the "has" relationship are at the same level and all the node with the "has part" relationship are also at the same level. Moreover, to achieve clarity we removed all unnecessary nodes and use the standard relationships that are easy to understand.

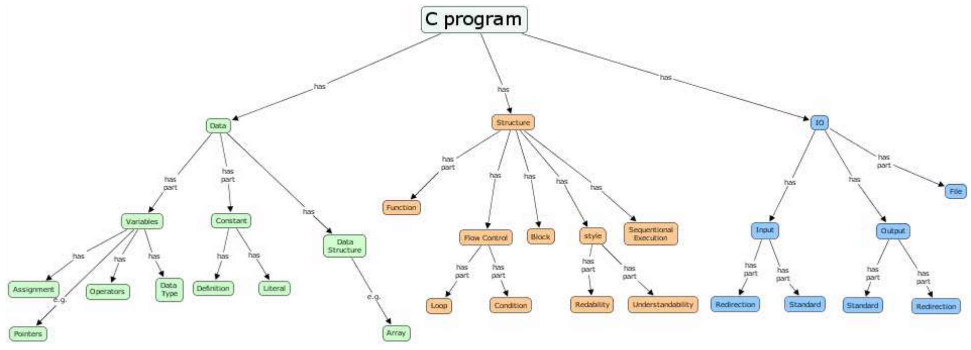

**Figure 6.** Harmony and clarity in the ontology

## 5 Discussion

We have already developed more than 20 teaching ontologies. Our research stresses the role of knowledge structuring for developing ontologies quickly, efficiently, and effectively. To achieve this goal we follow David Johnassen's idea of "using concept maps as a mind tool." The use of visual paradigm to represent and support the teaching process not only helps a professional tutor to concentrate on the problem rather than on details, but also enables pupils and students to process and understand great volume of information.

The development of knowledge structures in the form of ontologies, provides learning support and scaffolding that may improve student understanding of substantive and syntactic knowledge. As such, they can play a part in the overall pattern of learning, facilitating for example analysis, comparison, generalization, and transferability of understanding to analogous problems. Therefore, a visual knowledge structure editor provides a two-dimensional, iconic model that represents the author's understanding of the key elements in the concerned knowledge based.

At a basic level of knowledge representation, within the context of everyday heuristics, it is easier for educationalists simply to draw the ontology using conventional "pen and pencil" techniques. However, for more sophisticated knowledge representations, it is necessary to master appropriate programming and the involved language, or to use ell-known ontology editors.

It is possible to use any of the available graphical editors to design visual ontology, e.g. Paintbrush, Visio, Inspiration, or Mind Manager. But any effective computer program for knowledge engineering should perform the functions described for structuring the stages of a subject domain. Accordingly, it should correspond to the phenomenological nature of the knowledge elicitation involved using the appropriate algorithms previously detailed. This program must support the knowledge engineer through incorporating game rules that are clear, transparent, and functional. Ideally, the knowledge engineer should be able to tailor the program to his or her specific requirements. Concerning this, each analytical stage may be represented visually and accurately modelling the knowledge domain, elements already being realized in some commercial expert system shells.

This described approach can be applied to developing those tutoring systems where general understanding is more important than factual details. Furthermore, ontology design may be used as an assessment procedure for expressive as opposed to exploratory learning. For both formative and summarizing assessment purposes, students can clearly indicate the extent as well as the nature of their knowledge and understanding through creating ontology and explaining the involved processes.

## Acknowledgements

This work is partly supported by the grant of Russian Foundation for Basic Research No.04-01-00466.

## References

Adeli, H. (1994) Knowledge Engineering. McGraw-Hill, New-York.

Boose, J.H. (1990) Knowledge Acquisition Tools, Methods and Mediating Representations. In Knowledge Acquisition for Knowledge-Based Systems (Motoda, H. et al., Eds), IOS Press, Ohinsha Ltd., Tokyo,.123-168.

Brusilovsky, P. and Rizzo, R.(2002), Map-Based Horizontal Navigation in Educational Hypertext. In Proceedings of Hypertext, University Of Maryland, College Park, USA

Conlon, T. (1997) Towards Diversity: Advancing Knowledge-based Modelling with Knowledge Acquisition. In Proceedings of 8th International PEG Conference, Sozopol, Bulgaria, 379-386.

Eisenstadt, M., Domingue, J., Rajan, T. & Motta, E. (1990) Visual Knowledge Engineering. In IEEE Transactions on Software Engineering, Vol.16, No.10, 1164-1177.

Fensel, D. (2001) Ontologies: A Silver Bullet foe Knowledge Management and Electronic Commerce. Springer.

Gavrilova, T.A. & Voinov, A. (1996) Visualized Conceptual Structuring for Heterogeneous Knowledge Acquisition . In Proceedings of International Conference on Educational Multimedia and Hypermedia, EDMEDIA'96, MIT, Boston, USA, 258-264.

Gavrilova, T., Voinov, A. (1998) Work in Progress: Visual Specification of Knowledge Bases // Lecture Notes in Artificial Intelligence 1416 "Tasks and Methods in Applied Artificial Intelligence", A.P.del Pobil, J.Mira, M.Ali (Eds), Springer, 717-726.

Gavrilova, T.A., Voinov, A., Vasilyeva E. (1999) Visual Knowledge Engineering as a Cognitive Tool / Proc. of Int. Conf. on Artificial and Natural Networks IWANN'99, Spain, Benicassim, 123-128.

Gavrilova, T. (2003) Teaching via Using Ontological Engineering // Proceedings of XI Int. Conf. "Powerful ICT for Teaching and Learning" PEG-2003, St.Petersburg, 23-26.

Gavrilova, T., Kurochkin M., Veremiev V. (2004) Teaching Strategies and Ontologies for E-learning // Int. Journal "Information Theories and Applications", vol.11, N1, 61-65.

Gómez-Pérez, A., Fernández-López, M., Corcho, O. (2004) Ontological Engineering
with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web
Springer.

Gruber, T. (1993) A translation approach to portable ontology specifications. Knowledge Acquisition , Vol. 5, 199- 220.

Guarino, N. & Giaretta, P. (1998) Ontologies and Knowledge Bases: Towards a Terminological Clarification. In Towards Very Large Knowledge Bases: Knowledge Building & Knowledge Sharing, IOS Press, 25- 32.

Guarino, N., Welty, C. (2000) A Formal Ontology of Properties. In R. Dieng and O. Corby (eds.), Knowledge Engineering and Knowledge Management: Methods, Models and Tools. 12th International Conference, EKAW2000. Springer Verlag, 97-112.

Jasper, R. and Uschold, M (1999). A Framework for Understanding and Classifying Ontology Applications. In Twelfth Workshop on Knowledge Acquisition Modeling and Management KAW'99.

Jonassen, D.H. (1998) Designing constructivist learning environments. In Instructional design models and strategies (Reigeluth, C.M. (Ed), 2nd ed., Lawrence Eribaum, Mahwah, NJ.

Miller, G. (1956) The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. The Psychological Review, vol. 63, 81-97.

Mizogushi, R. and Bourdeau J. (2000), Using Ontological Engineering to Overcome Common AI-ED Problems. International Journal of Artificial Intelligence in Education, volume 11, .1--12.

Neches, et al . (1991) Enabling Technology for Knowledge Sharing. AI Magazin,. Winter, .36- 56.

OilEd (2004) Bechhofer, S. and Ng G. Accessed from http://oiled.man.ac.uk/ at December 7,

OntoEdit (2004) AIFB, University of Karlsruhe. Accessed from http://www.ontoknowledge.org/tools/ontoedit.shtml at December 07.

OntoBroker (2004)Accessed from http://ontobroker.aifb.uni-karlsruhe.de/index_ob.html at December 7.

Ontolingua (2004)Stanford University. Accessed from http://www.ksl.stanford.edu/software/ontolingua/ at December 7.

Protégé (2004) Stanford Medical Informatics. Accessed from http://protege.stanford.edu/ at December 07.

Sowa, J. F. (1984) Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, Reading, Massachusetts.

Scott, A., Clayton, J.E. & Gibson E.L. (1994) A Practical Guide to Knowledge Acquisition, Addison-Wesley.

Swartout, B., Patil, R., Knight, K. & Russ, T. (1997) Toward Distributed Use of Large-Scale Ontologies. Ontological Engineering, AAAI- 97 Spring Symposium Series, 138- 148.

Tu, S., Eriksson, H., Gennari, J., Shahar, Y. & Musen M. (1995) Ontology-Based Configuration of Problem-Solving Methods and Generation of Knowledge-Acquisition Tools. Artificial Intelligence in Medicine, N7, 257-289.

Uschold, M., Gruninger M (1996). Ontologies: Principles Methods and Applications. Knowledge Engineering Review, vol 11, N.1

Wielinga, B., Schreiber, G. & Breuker J. (1992) A Modelling Approach to Knowledge Engineering. In Knowledge Acquisition, 4 (1), Special Issue, 23-39.

Yourdon, E. (1989) Modern Structured Analysis. Prentice-Hall Bristol, Computer Science Department.

Wielinga, B., Schreiber, G. & Breuker J. (1992) A Modelling Approach to Knowledge Engineering. In Knowledge Acquisition, 4 (1), Special Issue, 23-39.

Yourdon, E. (1989) Modern Structured Analysis. Prentice-Hall Bristol, Computer Science Department.