Published by FESB-Split

Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture

University of Split

R. Boškovića b.b.

HR-21000 Split

Croatia

Text design: Hrvoje Dujmić

Cover design: Zoran Perdić Lukačević

Production organizer: Branko Burmaz

Printed in Croatia by Tiskara Franjo Kluz, Omiš

ISBN 953-6114-44-5

ISBN 953-6114-44-5

INTERNATIONAL CONFERENCE ON
SOFTWARE, TELECOMMUNICATIONS AND
COMPUTER NETWORKS

SoftCOM 2001

FACULTY OF ELECTRICAL ENGINEERING, MECHANICAL ENGINEERING AND NAVAL ARCHITECTURE - FESB SPLIT

TECHNOLOGY CENTER SPLIT

CROATIAN TELECOM, TELECOMMUNICATIONS CENTER SPLIT

Sponsored by:

IEEE COMMUNICATIONS SOCIETY (COMSOC)
Technical Committee of Communications Software
Technical Committee of Communication Switching and Routing

MINISTRY OF SCIENCE AND TECHNOLOGY
REPUBLIC OF CROATIA

UNIVERSITY OF SPLIT

October 9-12, 2001
Split - Dubrovnik, Croatia
Ancona – Bari, Italy

considered fair. In situations of an under-provisioned network (<75%), connections with larger requested rates do not achieve their fair bandwidth.

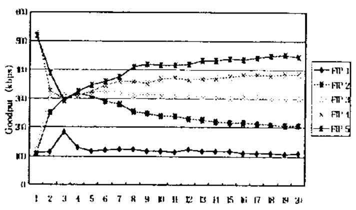

Figure 9. The Effect of Bucket Size

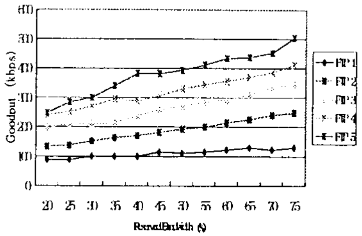

Figure 10. The Effect of Reserved Bandwidth

5. CONCLUSIONS

Differentiated Services is a newly proposed architecture to provide service differentiation and QoS guarantees for different users and applications. The DiffServ architecture does not maintain per-flow state information and overcomes the scalability problem of IntServ architecture.
In this paper, we present a prototype DiffServ-capable router (DSR) for Differentiated Services architecture. The prototype DSR consists of classifier, traffic conditioners, queues and scheduler. The performance evaluations of prototype DSR are done through NS-2 simulator with different router mechanisms and traffic. The results of simulations are useful for the design of future router.

REFERENCES

[1] Braden R, Clark D, Shenker S. Integrated services in the Internet architecture: An overview. IETF RFC 1633, June 1994
[2] Blake S, Black D, Carlson M, Davies E, Wang Z, Weiss W. An architecture for differentiated services. IETF RFC 2475, December 1998.
[3] Nichols K, Blake S, Baker F et al. Definition of the differentiated services field (DS Field) in the IPv4 and IPv6 headers. IETF RFC 2474, December 1998
[4] Jacobson V, Nichols K, Poduri K. An expedited forwarding PHB. IETF RFC 2598, June 1999
[5] Heinanen J, Baker F, Weiss W, Wroclawski J. Assured forwarding PHB group. IETF RFC 2597, June 1999
[6] Bernet Y, Blake S, Grossman D, Smith A. An informal management model for diffserv routers. IETF Internet draft <draft-ietf-diffserv-model-04.txt >, July 2000
[7] Clark D, Fang W. Explicit allocation of best-effort packet delivery service. IEEE/ACM Transaction on networking, 1998, 6(4):362-373
[8] Floyd S, Jacobson V. Random early detection gateways for congestion avoidance. IEEEE/ACM Transaction on Networking, 1993, 1(4):397-413
[9] Floyd S, Jacobson V. Link-sharing and resource management models for packet networks. IEEE/ACM Transactions on Networking, 1995, 3(4): 365-386
[10] Network Simulator –NS (version 2), http://www.isi.edu/nsnam/ns/

QUESTIONS AND POSSIBLE ANSWERS ABOUT ONTOLOGIES CONCERNING INTELLIGENT AGENTS

Maja Stula, Darko Stipanicev, Mirjana Bonkovic
University of Split, FESB
Ruđera Boškoviæa bb, 21000 Split, Croatia
Tel: +385 21 305816, E-mail: maja.stula@fesb.hr, darko.stipanicev@fesb.hr, mirjana.bonkovic@fesb.hr

Abstract: *The main purpose of ontologies, according to the recent works in Artificial Intelligence, is to enable communication among software systems. An ontology is an explicit specification of knowledge conceptualization. Conceptualization is simplified, abstract review of a domain defined as a part of the world that we are dealing with. Ontologies are closely connected with multiagent systems. In this paper the development of ad hoc ontologies for multiagent system for monitoring and control of dislocated greenhouse model is described. Also some possibilities of integrating ontologies in further development of intelligent agent architecture are discussed.*

KEYWORDS: Ontology, knowledge, conceptualization, agent technology

INTRODUCTION

An ontology could be defined as specification of a conceptualization [1]. More precisely, an ontology is a description of the entities, concepts and relationships existing in the real world we are dealing with. The term ontology was originally employed in the field of philosophy as study of being, a branch of metaphysics relating to the nature and relations of being. We are primarily concerned with using ontology for enabling communication between computer systems in a way that is independent of the individual system technology, information architecture and application domain. Formally represented knowledge, in any domain, is based on a *conceptualization.* Conceptualization defines objects, concepts, and other entities that are assumed to exist in some area of interest and the relationships that exist among them [4]. A conceptualization is an abstract, simplified view of the part of the world that we wish to represent for the some purpose. That part of the world is usually called domain. Every knowledge base, knowledge-based system, or knowledge-level agent is, explicitly or implicitly, committed to some conceptualization.
What could be conceptualized? For AI systems, everything that "exists" could be represented. What is difference between ontology and knowledge base? Ontology provides the basic structure for building knowledge base. The basic structure contains set of concepts and terms for describing a certain domain. When the knowledge of a domain is represented in a declarative formalism, the set of represented objects is called the universe of discourse. This set of objects, and the relationships among them, are reflected in the representational vocabulary and they represent knowledge [2]. Similar to any other application, an agent application is connected with problems solving in a certain domain. To be able to share their knowledge, agents need to share common ontology, or they have to be able to translate between different ontologies.
In context of agents Wooldridge and Nwana [3] classify ontologies in three different categories:

- ad hoc ontologies
- standard ontologies
- global ontologies

Majority of existing agent systems use implicitly defined ontologies for special problem which that system is covering. Such ad hoc ontologies are usually not reusable, they are hard to change or expand. And they oppose obstacle in connecting those systems with another systems (agent system, knowledge base system, etc.).
Standard ontologies are currently in the phase of development. Probably, the best known work in this area is ARPA KSE (Knowledge Sharing Effort). Such standard ontologies should have possibility to be shared across disparate software developers. There are also efforts to define and generate global ontologies.
Global ontologies, in opposite to standard ontologies, should be general, rather than directed at specific domain. Projects for developing global ontologies are WordNet and Cyc [3].
Another classification of ontologies according to their level of generality, is shown in Fig. 1. [4].

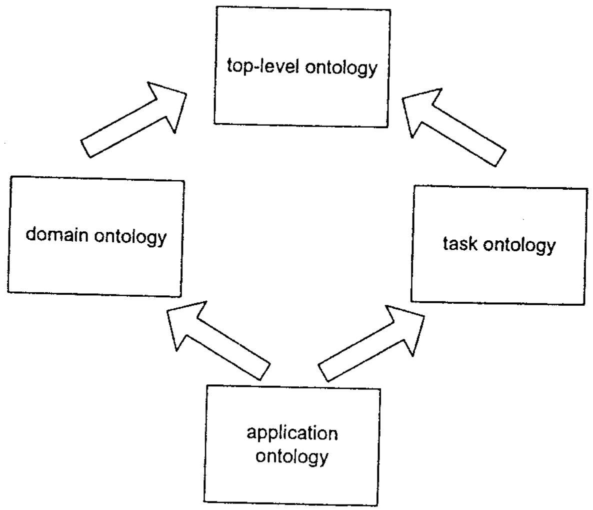

Fig. 1. Classification of ontologies, according to their level of dependence on a particular task or point of view. Arrows represent specialization relationships.

- *Top-level ontologies* describe very general concepts like space, time, matter, object, event, action, etc., independent of a particular problem or domain.
- *Domain ontologies* and *task ontologies* describe, respectively, the vocabulary related to a generic domain (like medicine, or automobiles, or in out case greenhouse model) or a generic task or activity (like controlling or selling), by specializing the terms introduced in the top-level ontology.
- *Application ontologies* describe concepts depending both on a particular domain and task, which are often specializations of *both* the related ontologies.

1. DESIGNING ONTOLOGIES

Designing ad hoc ontologies according to the submitted taxonomies will be considered. When we choose how to represent something in an ontology design decisions are made. This design decisions should lead to ontologies with following properties [5]:
1. Clarity: An ontology should effectively communicate the intended meaning of defined terms. Definitions should be objective. All definitions should be documented with natural language.
2. Coherence: An ontology should be coherent: that is, it should sanction inferences that are consistent with the definitions. At the least, the defining axioms should be logically consistent. Coherence should also apply to the concepts that are defined informally, such as those described in natural language documentation and examples. If a sentence that can be inferred from the axioms contradicts a definition or example given informally, then the ontology is incoherent.
3. Extendibility: An ontology should be designed to anticipate the uses of the shared vocabulary. It should offer a conceptual foundation for a range of anticipated tasks, and the representation should be crafted so that one can extend and specialize the ontology monotonically. In other words, one should be able to define new terms for special uses based on the existing vocabulary, in a way that does not require the revision of the existing definitions.
4. Minimal encoding bias: The conceptualization should be specified at the knowledge level without depending on a particular symbol level encoding.
5. Minimal ontological commitment: An ontology should require the minimal ontological commitment sufficient to support the intended knowledge sharing activities. An ontology should make as few claims as possible about the world being modeled, allowing the parties committed to the ontology freedom to specialize and instantiated the ontology as needed.

2. THE REAL SYSTEM

The actual process of designing ontologies is not yet well defined, but when we are designing ontologies we have to take care about these properties.
What do we have now?
First we need to conceptualize the problem we are working on. As it has been already stated, the conceptualization defines objects, concepts, and other entities that are assumed to exist in some area of interest and the relationships that exist among them.
The process of conceptualization is quite individual. Our case study was monitoring and control of dislocated greenhouse.
Three basic tasks were considered:

- monitoring temperature and moisture values
- automatic control of temperature and moisture,
- video-surveillance of greenhouse.

The problem is schematically shown on Fig. 2.

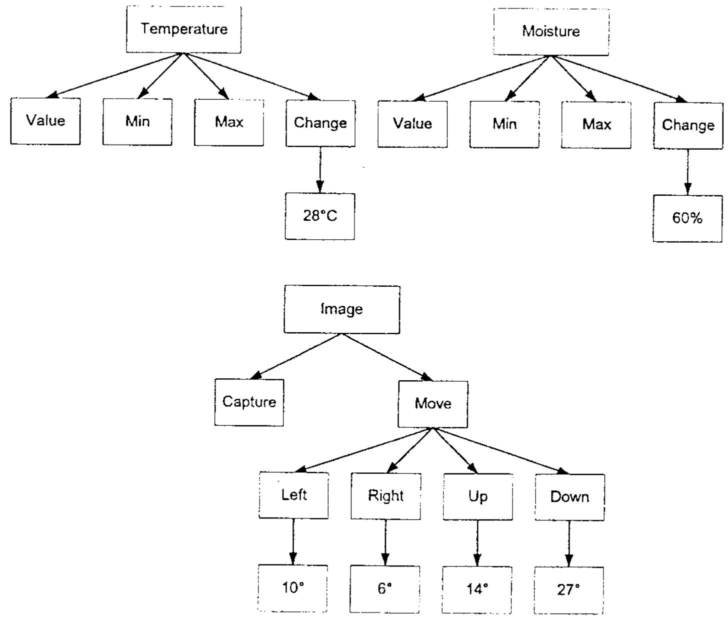

Fig. 2. Conceptualization of monitoring and control of dislocated greenhouse

Main classes are temperature, moisture and image. Subclasses of classes Temperature and Moisture are Value, Min, Max, and Change. Change is defined as the reference value of moisture or temperature. Subclasses of class Image are Capture and Move. Subclass Move is defined for controlling the movement of camera in four directions. Directions Left, Right, Up and Down are also defined as subclasses of class Move. Directions are defined camera movement values, in gradients. This conceptualization which defines ontologies temperature, moisture and image is not based only on previously defined ontology properties. This is designed for the real application. This application has some delimiters which have been taken in consideration. The main delimiter was demand to integrate existing parts of the system which where developed and designed prior to developing and designing the hole system.

3. PROPOSITION OF ONTOLOGY AGENT NAME SYSTEM

We developed multiagent system for monitoring and control of dislocated greenhouse using JATLITE and KAPI [6]. Fig. 3 shows JATLite infrastructure.

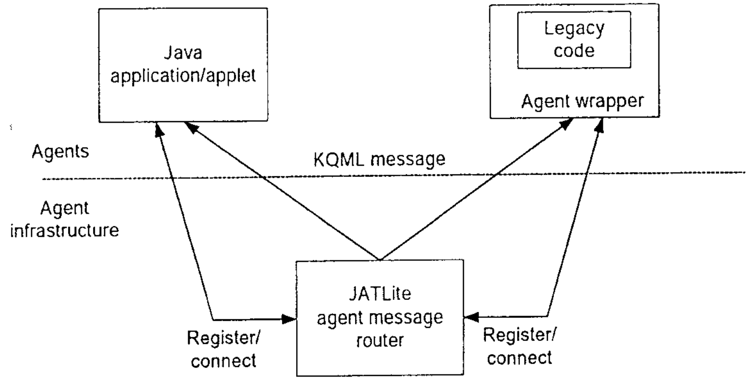

Fig. 3. JATLite infrastructure

The infrastructure is based on JATLite AMR (Agent Message Router). All agents communicate through AMR. The multiagent theory does not require AMR but most applications use some kind of central ANS (agent name server or service) or in case of JATLite packet, AMR. This could be used for resolving problem with ontologies concerning agents in the way that particular ANS or AMR are connected with particular domain or ontology. Something like DNS (Domain Name System) service could be made for agent systems. The Domain Name System is a distributed database [7]. Programs called *name servers* constitute the server half of DNS's client-server mechanism. Name servers contain information about some segment of the database and make it available to clients, called *resolvers*. Resolvers are often just library routines that create queries and send them across a network to a name server.

Each domain has a unique domain name, so the organization that runs the domain is free to name hosts and sub domains within its domain. Whatever name they choose for a host or sub domain, it won't conflict with other organizations' domain names, since it will end in their unique domain name (Fig. 4.).

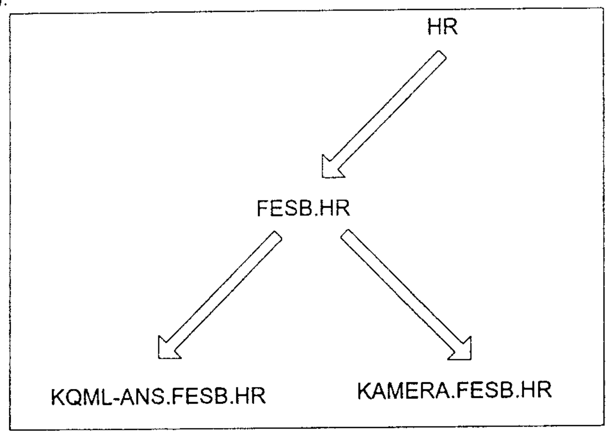

Fig. 4. Hosts names and sub domains within HR domain

The similar technique and the same hardware could be used to connect ontology with IP address of ANS. ANS should be specialized for particular ontology. Agents on the Internet would use that service (named, for example, OANS-Ontology Agent Name System) to connect with ANS that covers ontology that they are interested in. There is possibility to organize subontologies like sub domains in DNS service (Fig. 5). Maybe managing of domains would not be geographically but more *ontologically*. For example, ontology STOCK EXCHANGE would be managed by the WORLD BANK, ontology Pharmacy by large pharmaceutical companies etc.

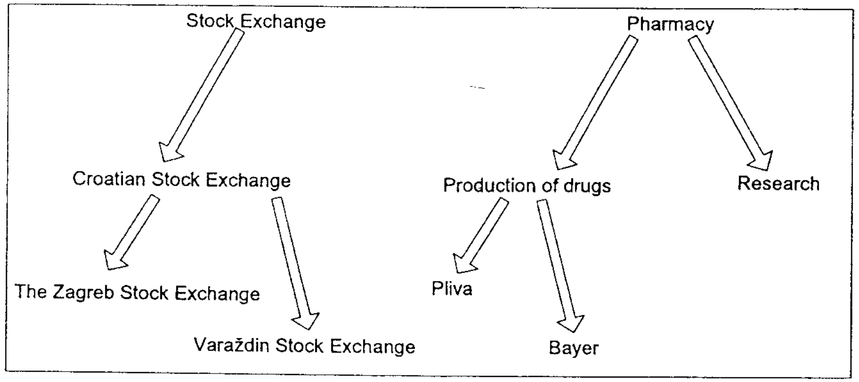

Fig. 5. Organization of subontologies like sub domains in DNS service

In such a way the Internet agent applications could be easily implemented for information gathering.

4. CONCLUSION

The field of ontology is closely connected with agent systems and it is still in the phase of development. New standards, new propositions and new tools are emerging every day.
In this paper we have proposed OANS-Ontology Agent Name System, as one possible solution for integrating ontologies in intelligent agent architecture. The idea of such solution has emerged from DNS service, the necessary infrastructure for its implementation exists. Our further work will be multiagent system for monitoring and control of dislocated objects.

REFERENCES

[1] T. R. Gruber: A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199-220, 1993.
[2] B. Chandrasekatan, J. R. Jospehosn, V. R. Benjamins: What are ontologies, and why do we need them?, *IEEE Intelligent Systems*, January/February 1999, 20-26.
[3] H. Nwana, M. Wooldridge: Software Agent Technologies.
[4] Guarino, N. 1997. Semantic Matching: Formal Ontological Distinctions for Information Organization, Extraction, and Integration. In M. T. Pazienza (ed.) *Information Extraction: A Multidisciplinary Approach to an Emerging Information Technology*. Springer Verlag: 139-170.
[5] In Formal Ontology in Conceptual Analysis and Knowledge Representation, edited by Nicola Guarino and Roberto Poli, Kluwer Academic Publishers, in press. Substantial revision of paper

presented at the International Workshop on Formal Ontology, March, 1993, Padova, Italy. Available as Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University. Toward Principles for the Design of Ontologies Used for Knowledge Sharing Revision: August 23, 1993 Thomas R. Gruber
[6] JATLite: A Java Agent Infrastructure with Message Routing, H. Jeon, C. Petrie, M. R. Cutkosky, IEEE Internet Computing, March-April 2000, 87-96
[7] O'Rilley DNS and Bind, By Cricket Liu & Paul Albitz; Third Edition, September 1998, O'Reilly & Associates.