Interoperating Distributed Expert Systems at a Semantic Level

Takahira Yamaguchi and Daiki Kishimoto
School of Information, Shizuoka University
3-5-1 Johoku Hamamatsu, 432 JAPAN
{yamaguti, kisimoto}@cs.inf.shizuoka. ac.jp

In order to develop robust cooperative knowledge systems in real and large scale industrial applications, we try to present facilities for deep interoperation at a semantic level between distributed expert systems, modeling expert systems at a proper granularity and taking a cooperation method based on the difference between the models.

CommonKADS [J.Breuker and W.Van de Velde 94] is a well-organized knowledge library and has a set of canonical functions to describe inference engines, such as Select, Compare, Merge. Because canonical functions have just declarative semantics using (extended) first-order calculus, they are too general to be exchanged from one expert system to another. So we put generic procedural semantics on canonical functions and they are called inference primitive structure templates (IPST). In our interoperation environment, inference primitives are represented by IPST.

Because IPST can be taken as a common task ontology, we can adopt a specification sharing based approach to interoperate distributed expert systems. Although one expert system (originator) can get the information about capabilities of the other expert system (recipient) through the shared specification. it is important to identify the capability available to the originator. A method to find out the the difference arising in the context of the correspondence between inference primitives of an originator and those of a recipient is presented here as follows:

1. Make a set of correspondence in which inference primitives are the same between an originator and a recipient.
2. By taking a look at the context (pre-inference primitives, post- inference primitives, inputs, outputs and reference knowledge) of the inference primitive in the correspondence, the correspondence value is computed. The more similar the context, the larger the value.
3. The correspondence value is propagated to pre- and post- inference primitives.
4. After completing propagation over all inference primitives in the correspondence, the difference arising in the context of the correspondence with large value can be used as a reply message to modify the originator's inference engine.

Using the above modeling and cooperation methods and then communication facilities with conversion function using a common domain ontology developed manually, we can build up an interoperation environment for distributed expert systems. In order for the interoperation environment to be enacted at each development site, each expert system should be manually modeled there, using IPST. However, there is no more tasks than modeling expert systems in order to get into the interoperation environment.

After implementing the interoperation environment by SICStus-Prolog, we have applied it to the operation between the following two expert systems: an enterprise diagnosis expert system called FIMCOES (originator) and a troubleshooting expert system (recipient). In order for the originator to improve its performance, it got into the interoperation environment and sent its models to the recipient through the interoperation environment. The interoperation has been done four times. Finally the following rely message has been accepted by the originator: Add a select primitive using an enterprise model just before a compare primitive. The inference structure of FIMCOES has been modified according to the message using some simple enterprise model developed manually. FIMCOES after the modification has better performance than FIMCOES before modification.

Based on the work here, we are planning to approach the interoperation among more heterogeneous distributed expert systems.

Acknowledgments

Thanks for discussions and implementation to Mr. Masaru Kawaguchi and Mr. Hidenari Inomata.

References

Common KADS Library for Expertise Modeling. IOS Press.