Masterprojekt: Persistence Mechanism in Self-Referential Systems

Semester: Wintersemester 2019/20

Sprache: Deutsch / Englisch

Motivation

The "Language Engineering for Multi-level Modeling" (LE4MM) project conducted together with Tony Clark from Aston University aims at further developing an integrated meta-modeling and meta-programming environment XModeler, and, based on that, at the development of new self-referential enterprise systems that integrate enterprise software with conceptual models of themselves and the context they operate in at run time.

The XModeler is an extension of XMF-Mosaic and as such, constitutes a modeling and programing environment with an integrated language execution engine. Thus, the XModeler does not only support the creation of concepts on several classification levels within one model (i.e., a creation of a multi-level model), but it also allows for execution of model/its elements. It is possible as each entity of the system is not only a model, but also source code. A modification of the model content implies a modification of the source code, and the other way round. Due to this shared representation of model and code, each entity can be executed. Thereby, execution is not limited to querying model elements, but each entity can be enhanced by algorithms for an execution in the sense of a complete programming language.

One of the most critical tasks that any object-oriented systems needs to perform is to be able to make the data persistent, i.e., to enable the storage of data from working memory so that it can be restored when the application runs again. In object-oriented systems, there are several ways in which objects can be made persistent. A few classes of solutions for implementing persistence in object-oriented applications have been proposed, e.g.: serialization, the gateway-based object persistence approach, the object-relational database management system (DBMS) approach, and the object-oriented DBMS approach.

Please note that the choice of persistence method is an important part of the design of an application and influences highly its performance and usability. The novel architecture of XModeler as well as requirements towards a self-referential system, make the choice and the implementation of the persistence mechanisms an interesting task.

Institut für Informatik und Wirtschaftsinformatik (ICB)

Lehrstuhl für Wirtschaftsinformatik und Unternehmensmodellierung

Dr. Monika Kaczmarek-Heß
Tel.: 0201 / 183 - 4330
Monika.Kaczmarek-Hess@uni-due.de

R09 R04 H41
Universitätsstraße 9
45127 Essen

www.umo.wiwi.uni-due.de

Description

The main aim of this master project is to critically analyze existing persistence mechanisms and, taking into account characteristics of XModeler, formulated requirements and use scenarios, recommend and, depending on the size of the project team, implement a suitable persistence mechanism in the XModeler. To this aim the students should:

1. Make themselves familiar with object persistence mechanisms as well as the idea of self-referential systems, as well as integrated modeling and programming environment XModeler,
2. Define use scenarios and requirements, and use them to discuss/evaluate different mechanisms
3. Implement and test the selected mechanism in the XModeler (depending on the size of the project team and agreed upon scope of the project)

Expected Outcomes

A report pointing to identified mechanisms, formulated requirements, conducted comparison and rationale for selection. Implementation in the XModeler with accompanying documentation. A final presentation of the project results.

Exemplary Introductory Literature

Clark, T.; Sammut, P. et al.: Applied Metamodelling: A Foundation for Language Driven Development. 2008

Atkinson, M. P. (1978). Programming languages and databases. In S. B. Yao (Ed.), Fourth International Conference on Very Large Data Bases, September 13-15, 1978, West Berlin, Germany, pp. 408–419. IEEE Computer Society.

Atkinson, M. P., P. J. Bailey, K. Chisholm, W. P. Cockshott, and R. Morrison (1983). An approach to persistent programming. Comput. J. 26(4), 360–365.

Atkinson, M. P. and R. Morrison (1995). Orthogonally persistent object systems. VLDB Journal 4(3), 319–401.

Evans, H. (1999). Why object serialization is inappropriate for providing persistence in java. Technical report.

Atkinson, M. P. (2000). Persistence and Java - A Balancing Act. In Proceedings of Objects and Databases, International Symposium at ECOOP 2000, volume 1944 of Lecture Notes in Computer Science, pages 1-31, Dittrich, K. R. et al., Eds. Springer.

Atkinson, M. P., Bailey, P. J., Chisholm, K. J., Cockshott, W. P., and Morrison, R. (1983). An Approach to Persistent Programming. Computer Journal, 26(4):360-365.

Lamb, C., Landis, G., Orenstein, J., and Weinreb, D. (1991). The objectstore database system. Communications of the ACM, 34(10):50-63.

W. R. Cook and A. H. Ibrahim, "Integrating Programming Languages & Databases: What's the Problem?," in ODBMS.ORG, vol. Expert Article: Department of Computer Sciences, University of Texas at Austin, 2006.

Jordan, M. (2004, September). A comparative study of persistence mechanisms for the java platform. Technical Report SMLI TR-2004-136, Sun Microsystems Laboratories.

Stonebraker, M. and D. Moore (1996). Object-Relational DBMSs: The Next Great wave. Morgan Kaufmann Publishers, Inc.

Application procedure:

Please apply via email to the supervisor. Please attach a short letter of motivation (app. ½ A4 page) and a recent performance record ('Leistungsnachweis'). You can apply individually or in a group of 2-4 participants (in this case each person should still send a separate e-mail, however point to the other members of the group).

Application deadline: 20. October 2019, 23:59