5th Slovakian-Hungarian Joint Symposium on Applied Machine Intelligence and Informatics
January 25-26, 2007 ▒ Poprad, Slovakia

Knowledge-based Approach in Information Systems Life Cycle and Information Systems Architecture

Zdeněk Havlice, Ondrej Pločica

Technical University of Kosice
zdenek.havlice@tuke.sk, ondrej.plocica@tuke.sk

Abstract: A choice of suitable specifications of Information Systems (IS) in Software Life Cycle (SwLC) (represented by formal or semi-formal models, dependencies between them and the way they are used during SwLC phases execution) influence the cost and the outcomes quality for a given SwLC phase. Research area is oriented to analysis and design of the methods that serve for selection of formal and semi-formal models set together with dependencies between them. These methods and tools are based on critical characteristics of developed software system. In addition interests are oriented to an integrated CASE system configuration to support the selected set of modeling methods/tools based on intelligent extended project database, and to design of multilayer architecture of IS with intelligent layer based on formal models of IS and user requirements.

Keywords: software life cycle, information systems architecture, knowledge-based techniques, abstract model, CASE, software process

1 Introduction

Explicit knowledge in information systems and other software applications life cycle is becoming more important because their domains are inherently knowledge-intensive. This knowledge is often not explicitly dealt with in software development and there are lack of suitable methods and tools for knowledge based computer aided decisions in software processes. Software development requires expertise and experience, which are also implicit. There are critical important decisions about methodology/methods/tools, which have to be made in earlier phases of software life cycle (SwLC). These decisions are usually done on the base of previous experiences with similar software projects. Other critical important decisions are made in usage and maintenance phases of SwLC – decisions about what have to be changed or corrected for fulfillment of new requirements or elimination of recognized errors. These decisions are made on the base of good project documentation (project database in CASE system), or often

on the base of existing project documentation and analytics/programmers experiences. Representation of these experiences by using knowledge-based or related techniques can be advantageous in all phases of SwLC [4], [10]. Knowledge requires a declarative representation such as constraints or rules, thus requiring to expand conventional software development with these representations.

Proposed approaches deal with SwLC domain, where goal is looking for a suitable software technology that can be applied to the realization of a concrete information system with pre-defined properties, and deal with IS architecture domain, where goal is to introduce independent software layer which should manage, maintain and modify the information system at the development level.

2 Knowledge Engineering and Software Engineering

The communities of Software Engineering and Knowledge Engineering share a number of common topics. However, both communities mostly live in their own worlds and the number of forums, journals and conferences for discussing synergies is still relatively small [6], [7]. Examples include Joint Conference on Knowledge-Based Software Engineering (JCKBSE), Workshop on Knowledge-Based Object-Oriented Software Engineering (KBOOSE) [5], or International Journal of Software Engineering and Knowledge Engineering [6].

There are many potential benefits the Software Engineering community can achieve by applying knowledge-based (KB) techniques in various phases of the software life cycle [7]:

- Requirements engineering can benefit from KB techniques in terms of knowledge representation and process support.
- Component reuse is chosen as a potential application area during design.
- Software modeling languages and methodologies can benefit from the integration with knowledge representation languages such as RDF and OWL in various ways, e.g. by reducing language ambiguity, enabling validation and automated consistency checking
- The main advantage in business rules approach, where business logic is modeled separately and processes by a rule engine, is the declarative specification of knowledge which tends to change frequently.
- KB techniques help to connect the electronic communication (via forums and mailing lists) of the developers with bug-reports and the affected areas in the source code.
- KB techniques could help to generate basic test cases since they encode domain knowledge in a machine processable format.

3 Knowledge-based Approach in Development of Information Systems

One of the problems of Software Engineering [4] in the area of information systems is with finding of adequate methodology, methods and tools for developing new large software systems. The idea about its solving is to find a methodology for the life cycle of an information system, which uses knowledge-based methods for searching a suitable technology for realization of a system with pre-defined properties. To reach this goal an expert system of software projects should be built.

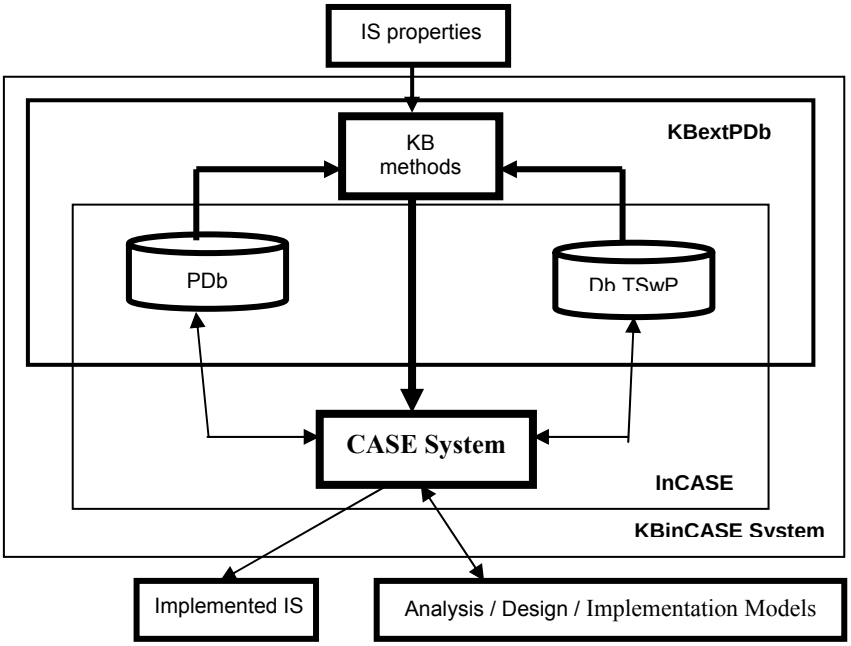

Figure 1
Knowledge-based integrated configurable CASE system

Knowledge-based integrated configurable CASE system (KBinCASE) is our proposed approach in this field. KBinCASE architecture is shown in Figure 1. Integrated configurable CASE system (InCASE) is system, where software lifecycle, modeling methods and tools can be configured. Configuration is based on Knowledge-extended project database (KBextPDb), consisting of database of software technologies (Db TSwP), project database (PDb) and knowledge-based methods (KB methods), which help to select suitable methodology, methods and tools for developing software system with specified properties (IS properties). Configured CASE system is then used to IS implementation.

New software methodologies appear and disappear rapidly. The need to use them together with suitable CASE tools is obvious. It is not economically effective to buy a new CASE tool and to re-educate staff (how to use the CASE tool), each time the methodology has been switched. KBinCASE usage is the possible solution.

Based on problem domain, it is possible to identify the software project category of such solution and which particular technologies will be applied. Rather than wasting the time by searching and testing different CASE products from different vendors, suitable software technology is specified. Software technology specification is reusable and it can be later re-used for each solution that falls within the same software projects category too. Software technology specification is used for the inCASE configuration and configured InCASE is applied in IS development.

4 Knowledge-based Approach in Maintenance and Using of Information Systems

One of the main problems of Software Engineering in the area of information systems is insufficient flexibility and efficiency of existing tools and methods for modifying of existing successful large software systems to rapidly changing new user requirements. The idea about its solving is by applying knowledge-based methods to IS architecture by using of an independent software layer which should manage, maintain, and modify the information system at the development level.

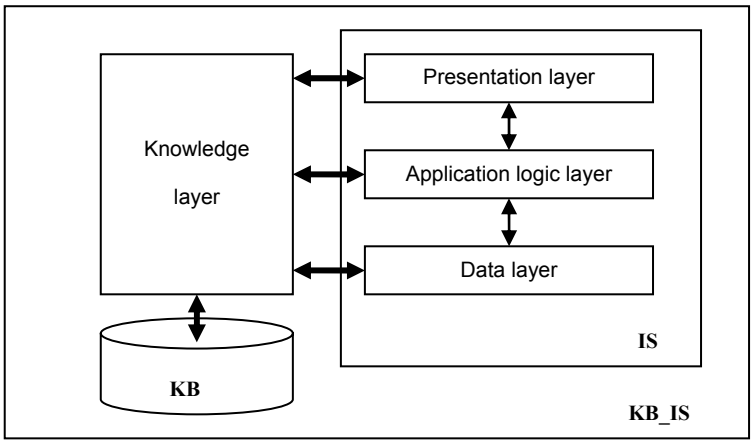

Figure 2
Knowledge-based information system architecture

Knowledge-based information system (KB_IS) architecture, as is shown in Figure 2, is based on three layers architecture of information systems [3], [8]. Knowledge layer with knowledge base (KB) is added to information system (IS) architecture. Knowledge layer plays important role in this architecture. Through knowledge layer information system knows itself. Knowledge layer is used to modification and maintenance of information system at all three layers.

4.1 Separating Business Rules in Knowledge Layer

Business rules [9] are contained in every information system that's ever been built, but for a long time have been implicit, hidden behind the other elements. Either they are embedded in programming code or they are implicit in mindless habits and procedures. Finding them is hard, understanding their business meaning is harder, and changing them is often almost impossible. Furthermore, companies are in danger of losing those few personnel who understand the business rules through attrition and retirement.

A Business Rules Approach is the right solution for all these problems [2]. Separating business rules from application procedural code give us ability to change rules without modifying application code.

4.2 Open Design Architecture

Open Design Architecture is possible way of simplifying IS design and maintenance. The main idea of this architecture lies in coupling IS design together with IS. Layer containing information about architecture, structure, relationships and other important project information relevant to IS and also information about user requirements are closely associated with IS.

On the present, this information is part of project database of CASE system used to IS development, or exists in different forms in different databases associated with project. Proposed architecture introduces knowledge layer with project information and user requirements information, which provides knowledge needed for higher level automation of IS modification and activities associated with IS maintenance.

Conclusions

Knowledge-based approach for IS development and maintenance was presented. Project information and user requirements information coupled with IS provides better maintenance and modification of IS.

Acknowledgement

This work was supported by VEGA Grant No. 1/2176/05 Technologies for Agent-based and Component-based Distributed Systems Lifecycle Support

References

[1] Havlice, Z., Kollár, J., Chladný, V.: Object-Oriented Analysis / Design and Data Flow Modeling. In: Acta Electrotechnica et Enformatica, Vol. 4, No. 4, 2004, pp. 64-70, ISSN 1335-8243

[2] Pločica, O.: High Level Intelligent Software Layer in Information Systems, 6th PhD Student Conference and Scientific and Technical Competition of Students of Faculty of Electrical Engineering and Informatics, Technical University of Košice, Proceedings from conference and competition: Košice, May 17, 2006, Košice: Technická univerzita, 2006, pp. 105-106, ISBN 80-8086- 035-1

[3] Szabó, C., Pločica, 0., Havlice, Z.: Application of AI in the Multi-Layer Architecture of Information Systems, Electronic Computers and Informatics ECI 2006, The 7th International Scientific Conference, Košice-Herlany Slovakia, September 20-22, 2006, Košice, VIENALA Press, Košice, 2006, First Edition, pp. 52-57, ISBN 80-8073-598-0

[4] Somerville, I.: Software Engineering (7th Edition), Addison-Wesley, Menlo Park, CA, 1999

[5] Workshop on Knowledge-Based Object-Oriented Software Engineering (KBOOSE), http://ssel.vub.ac.be/kboose

[6] International Journal of Software Engineering and Knowledge Engineering, http://www.ksi.edu/ /ijsk.html

[7] Happel, H. J., Seedorf, S.: Applications of Ontologies in Software Engineering, 2nd International Workshop on Semantic Web Enabled Software Engineering (SWESE 2006), November 2006, Athens, USA

[8] Fowler, M.: Patterns of Enterprise Application Architecture, Addison-Wesley, 2003

[9] Morgan, T.: Business Rules and Information Systems: Aligning IT with Business Goals, Addison Wesley, 2002

[10] Boehm, B.: A Spiral Model of Software Development and Enhancement, IEEE Computer, Vol. 21, #5, May 1988, pp. 61-72