

KNOWLEDGE – BASED MAINTENANCE ENVIRONMENT FOR LARGE INFORMATION HANDLING SYSTEMS

Nikolay V. Tkachuk
Institute for Applied Informatics,
Klagenfurt University
Universitätsstr. 65 -67
A-9020 Klagenfurt
nikolay@ifit.uni-klu.ac.at

Dmitry V. Kuklenko
Department for Information Management Systems
National Technical University
Frunze Str. 21
61002 Kharkiv UKRAINE
dkuk@kpi.kharkov.ua

Sergey V. Ovasapov
Firm “PromAvto matika”
Tchuvashkay Str. 6
61117 Kharkiv UKRAINE
dkuk@kpi.kharkov.ua

Konstantin M. Shehekotykhin
Department for Information Management Systems
National Technical University
Frunze Str. 21
61002 Kharkiv UKRAINE
kostva@kpi.kharkov.ua

Abstract: Some modern trends in software maintenance domain are depicted and discussed. The case study related topics, solutions, as well as some challenges and open problems based on the real maintenance project’s experience in the Ukrainian gas-transport industry, are presented. The idea of knowledge-based unified Maintenance Environment for large legacy information systems creation necessity and possibilities is introduced and discussed, and its first knowledge model specification is presented.

1. Introduction

Nowadays in many branches of Ukrainian production complex and others post-USSR countries a lot of complex *information handling systems* (IHS) are used. They were created and maintained during the last 10-15 years. We will use term IHS (*it was introduced by prof. H.C. Mayr under discussion of some questions related to these problems*) for information systems, which support the execution of individual tasks within appropriate

business or technological process. Various, frequently diverse, design decisions and software tools were applied during IHS development and use. Further usage of such systems assumes taking into account the following factors, which had been appearing during last 5-7 years:

Advent of essentially new information technologies for global and corporate information systems: Internet / intranet technologies, data warehouse and data mining, distributed network agents, system mobile calculations etc.

Beginning of the integration processes for Ukrainian industrial enterprises, business organizations and government structures into global information space.

Caused by these reasons successful and effective problems solutions of functioning systems for complex objects controlling in various industry branches, economy and social sphere, defines realization of such *legacy* IHS (LIHS) continuous software maintenance (SM) process. There are a lot of definitions for this notion, e.g.[Pi97], [Mi98], [Wa99]. We will understand the term *legacy system* as complex computerized system for data capture and processing, decisions making support etc., which was developed and introduced earlier in some business (production) organizations and *has a vital importance providing normal business (production) process*. Especially urgent task is a SM issues of legacy dispatching control and management computer systems of various technological processes, which have complex technical structures, large material costs and, in many cases, with “non-stop” working cycle: in power engineering, in chemistry, on transport etc. These are so-called systems “7 x 365”, which are constantly operating 365 days per year round the clocks. Already for one this reason, as a rule, their replacement as a whole or in parts is technically impossible and is economically inexpedient. Below we will consider one of typical classes of such systems more detailed: so-called dispatching LIHS of gas-compressor station on main gas pipelines, which represent rather essential component of energy complex infrastructure in any industrial country, including Ukraine.

Taking into account these factors we suppose that for such LIHS some *special SM - approaches* have to be considered and elaborated. We will try to present in the current report our ideas concerning these problems.

Section 2 focuses on some modern trends in SM - domain and gives a briefly overviews of relevant related works concerning these issues. In *Section 3* we present our results in the real SM - project which is currently performed in Ukrainian gas-transport industry branch, and we figure out the opened problems, which we were facing with. We introduce and discuss our proposition about the metaphor of *Knowledge-Based Maintenance Environment* in *Section 4*. We present in this section although its first knowledge model specification, in order to meet the software maintenance challenges in large LIHS. Then in *Section 5* we come with some conclusions and define briefly our point for future work.

2. Related Works: Some Trends and Problems in Software Maintenance Domain for Large LIHS

From our point of view, two factor groups cause an appropriate SM - process of LIHS independently from system application domain’s nature and some its specific details
new features of SW&IT
changes in own system production (business) process organization.

They influence some conflicts and difficulties by LIHS - *actors* (users and running applications), which have to be re-solved providing SM for legacy system architecture (software topology and data resources organization) that has to be changed to some target architecture. The general strategic goal of any SM is to provide and to support such form of system software architecture which:

is as much as possible invariant concerning the influence of these both factor groups in order to meet the new user requirements
keeps investments already enclosed in an appropriate LIHS.

These assumptions are corresponded with well-known considerations about common SM issues presented, e.g. in [Pi97, Mi98, Wa99]. Nowadays more and more attention is given to SM problems in a context of new information systems development and implementation, and, first of all, determined by Internet/intranet - technologies. They are used such areas as distributed information systems, multi-agents systems and mobile computing systems, heterogeneous and federated databases, Ecommerce and remote-control systems on the Internet etc. So, one of an examples of such rapidly increased interest to this problematic became the fact of providing in February – March, 2000 in Zurich (Switzerland) simultaneously at once three international conferences on SM and software reengineering (SR) problems in frameworks of Reengineering Weeks'2000 [EV00], [Ch00],[TME00]. The analysis of this representative international scientific - practical forum's materials, and some another publications allows us to emphasize the following modern trends in this research domain:

Closing integration between SM and SR activities. Furthermore, the idea about the necessity to involve them into a common software construction process is proposed, from the very first steps of the initial development, e.g. in [Vo99], [OI00], [ST00], [TS00]

Elaborating of several *environment concepts* for software maintenance process [Pi97], [Br00], [W99], e.g. in [Pi97] a Software Engineering Environment (SEE) is presented. As one special case of this one the so-called Maintenance-Specific Environment is proposed, but from our point of view it only is considered as a conglomerate of CASE-tools, without any appropriate modeling issues

Development a new maintenance approaches based on modern communication and business paradigms such as *Tele-maintenance via the Internet* [Te01], *disposable COTS (Commercial Off-The Shelf)* system maintenance [CHP00], [Vo99], [Vo00] etc.

Knowledge – based approaches for establishing and providing of maintenance activities for complex software systems, e.g. in [Ki99], [KR00].

Taking all these facets into account, we can state that SM activities are constantly presented in a whole spectrum of SW & IT - development and implementation problems, and *they have to be considered as a permanent evolutionary sub-process involved in common software system life cycle.*

This assumption is especially important for real-time LIHS where computer applications are characterized as sub-systems of enterprises used to control several interconnected technological (business) processes (tasks). This may only be done by focusing on *information infrastructure* of these processes and their actors (end-users and applications), which are working upon certain tasks they are responsible for. Not only legacy application programs, and data resources have to be continuously changed, but also *the knowledge*

about legacy system’s development and system operation issues have to be collected, re-organized, processed and analyzed continuously in order to perform an appropriate modifications, which have to be done in system maintenance process. Below we consider some our experience points, which confirm this statement.

3. Our LIHS Maintenance Case Study: Solutions, Experience, and Their Critical Analyses

Our research team at the Kharkiv National Technical University “KhPI” in close cooperation with the design firm «PromAvtomatika» (Kharkiv, Ukraine) have been working for two last years on some SM and SR problems in LIHS of gas-compressor stations (GCS). Such stations are the essential part of the gas-main pipeline system in Ukraine. Moreover, this system is a part of the gas pipeline connecting eastern Siberia (Russia) and western Europe, which provides more than 90% of Russian gas-export volumes, and ca. 30% of the West-Europe annual consumes of this resource [RMS00]. There are more than 70 such stations in several regions of Ukraine. We deal with the Romny GCS, which is situated at the northeast of Ukraine as first representative of this complex distributed system. Some of our starting points, which used in this project, were presented in [Tk00], and now we can consider achieved solutions, and analyze them critically. But at first we propose to have a short look on our Universe of Discourse (UoD) as a typical example of dispatching LIHS (“7 x 365” systems).

3.1 UoD Description and Maintenance Process Tasks

Each GCS is a complex technological object, which is intended to provide required gas pressure level in main gas-pipeline. The diagram in UML – notation [Om01], which represents the GCS structure, is shown on Fig. 1. Typically GCS includes a few Production Divisions (PD), where a set of gas-pumping aggregates is placed.

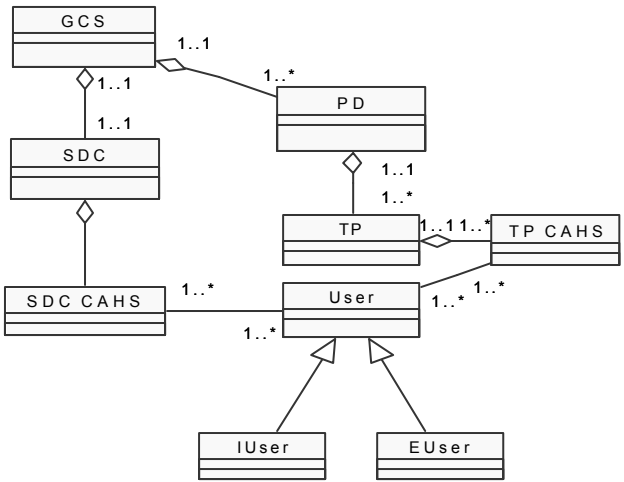


Figure 1. Common GCS structure presented in UML notation

Farther, each PD also contains the Technological Process (TP) Computer-Aided Handling System (TP CAHS). Such TP CAHS includes a number of subsystems, which control the appropriate TP. In typical PD we can find about 20 subsystems, e.g. Common Station Equipment (CSE) CAHS, Automated Air - Cooling (AAC) CAHS and many others. Each technological process is also characterized by a lot of appropriate parameters: their number can be estimated at tens or hundreds. The values of these parameters are important for estimation of PD operation quality and they have to be stored and analyzed in some system log-files. Besides that, each GCS has a Station Dispatcher Center (SDC), which provides full GCS operating control. The SDC has its own dispatching CAHS.

All the GCS staff can be divided into several users groups. Members of these groups have different rights on information access and several responsibilities for technological process control (it means they could have *roles*). These users groups are enumerated above:

Operators of separate TP CAHS

Sub-managers (engineers) in each PD

Dispatchers of SDC

The users of these groups are the internal system agents (IUser), which participate in system information processing. Besides that, some external (remote) users (EUser) can interact with GCS CAHS:

Operators of regional control office (this department is situated in Sumy city, 25 kilometers distance from Romny GCS)

Top-level managers and technical experts, who observe a few GCS of main pipeline region, and who make decisions about their coordination if necessary.

Obviously, external users need to be provided with GCS technological processes aggregated information. This information must be represented in analytical style and be comfortable in work.

The GCS legacy CAHS architecture is a conglomerate of typical PC-based LAN subsystems, designed and implemented using a 2-level file-server architecture paradigm and it was analyzed critically by us earlier [Tk00]. It was build on partly basing on COTS soft – and hardware solutions, elaborated e.g. by Compressor Controls Co., (CCC) [Cc01], Serck Control Co., [Se01], WinTECH Software [Wi01]. Some applications have been developed especially for mission critical GCS - facilities, such as e.g. our own advanced software solution for AAC CAHS [Tk99].

The main goals by our SM and SR activities have been defined as the following points:

- (a) elimination of functional redundancy in legacy software applications
- (b) support of new data transaction processing with improving process control quality
- (c) providing an unified operation data visualization mode for mission critical control subsystems

3.2 Some Achieved Maintenance and Reengineering Solutions, and Their Critical Analyses

(a) software reengineering solutions: in order to eliminate a functionality redundancy in legacy software applications the new component – based software architecture for some legacy applications was elaborated, one typical approach is shown on Figure 2, a) and b). The legacy software structure for data exchange processing was 2 - level solution including a lot of modules for each type of protocols used in several subsystems (e.g. MODBus protocol [Mo01] elaborated by WinTECH Software). These protocols facilitate data exchange between hardware controllers of several types (e.g. C-102, C-112, CCC etc.) and

appropriate software control subsystems (see Fig. 2 a.) The target software solution is the 3-level architecture, where all business logic tasks (data exchange protocol's functionality) are provided by only two new reusable software components (business objects): XProtocol and TechXObject, which were implemented using COM/DCOM technologies [Bo98].

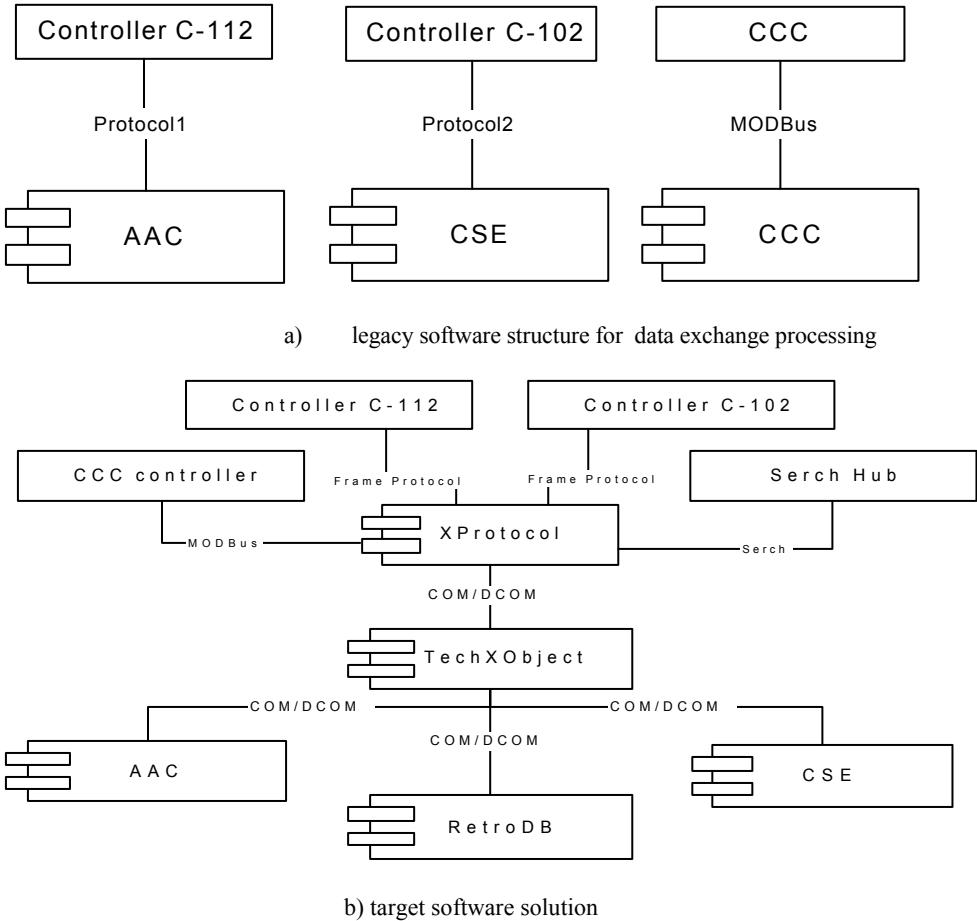


Figure 2. Software reengineering solution (continued)

(b) data processing maintenance solutions: with expect for new quality of operation data processing in LIHS Retrospective Database (RDB) was designed and was integrated into the LIHS structure [KT01] for decision making support by LIHS staff. The RDB provides the following intelligent functions versus some file-oriented data capturing facilities, which were earlier used as system log files in LIHS (the corresponding IDEF1X -diagram of RDB is given on Fig. 3):

- data collection and data accumulation for critical technological processes. It also gives the possibility of further graphical representation and analysis of these data.
- automatic aggregation of parameter's values after the expiration of some pre-defined operating time period

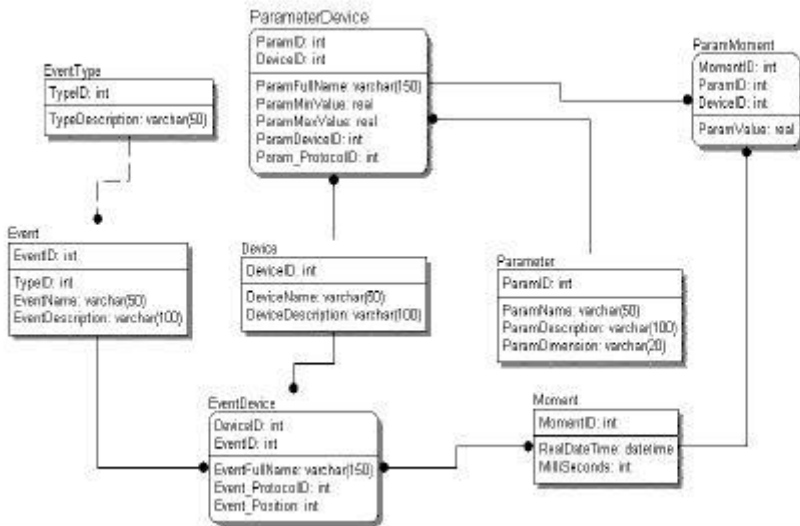


Figure 3. The IDEF1X diagram of RDB

It is important to mention here, that designing and successful implementation of this subsystem can be recognized as a prototype of some new system facilities, which have to be developed basing on the our main concept presented in Section 4.

(c) user interface reengineering solution: LIHS users had to operate with different programs that visualize technological processes that actually decreased efficiency of their work. In order to solve these problems in a complex we had elaborated the Web-based unified solution for LIHS operating data visualization. It had given us: (1) homogeneous platform for all control data representation inside the station's LIHS; (2) possibility to involve into control and troubleshooting process external experts, who are situated in regional office of gas-transport company. So from this point of view we can put a new quality in GCS management process. The visualization facility is implemented as a collection of COM components, which can be used in any Web-browser container, e.g. the MS Internet Explorer (MS IE) in our case (see example of screen snapshot on Figure 4).

3.3 Some Open Problems

The critical analyses of our maintenance and reengineering activities let us make the following intermediate conclusion: approx. 30 % of such tasks, needed to be provided in one department of LIHS, have been actually performed. The next steps in our LIHS improving process have to be done taking into account the following points:

- 1) a large number of subsystems and complex character of their interconnections in whole LIHS,
- 2) necessity of involving in maintenance and reengineering process some experts who are working outside of given system,
- 3) the fact, that there are currently more than 70 of such LIHS (in appropriate gas-compressor stations) at the Ukrainian main gas pipe-line system.

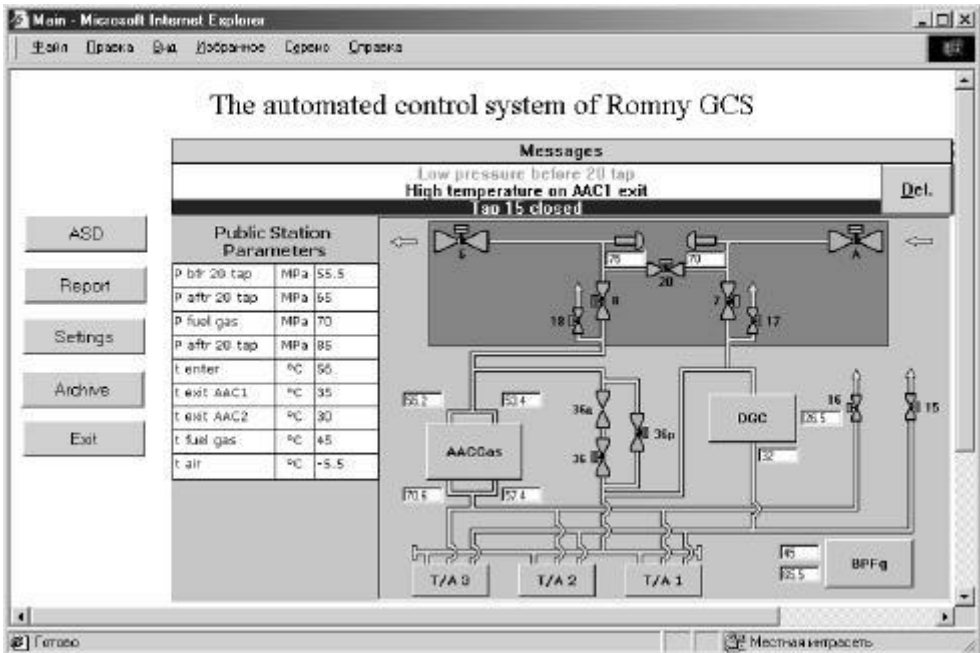


Fig. 4 GCS equipment mnemonic scheme visualized in MS IE browser

In the next section we will present our first vision of one possible approach how we can enhance the traditional SM and SR processes (at least, mostly used today in real-life software praxis), which is not more enough if we are dealing with complex distributed LIHS similar to the one type considered in this section.

4. Knowledge Based Maintenance Environment (KBME): Conceptual Model Specification

4.1 KBME Metaphor

We offer to consider the maintenance process not as (or not only as) a lot of some kind transforming procedures needed for program architecture and / or for data structures changes, but as an evolution process of system functionality, system operation quality. Not only the data structures and program modules have to be changed, but also about system development and operation knowledge have to be re-organized and processed continuously. This knowledge in most LIHS is presented fragmentally and not completely, in diverse forms such as

- some system user guides and system manuals (documentation),
- in business logic of running application programs,
- in local databases, system log files,
- in form of staff members skills and experience etc.

Our main idea is to collect all these kinds of such information, to re-structure it, and to provide it in an appropriate form (mostly visualized) for permanent computer-aided maintenance process. Such knowledge - based structure can be used as some kind of *spatial information environment*, in which an appropriate LIHS has to be placed as a *core-system*. Let us use the term *Knowledge-Based Maintenance Environment (KBME)* for this purpose. Taking into account our conclusions made in *Section 2*, we have to assume that KBME - model can be specified and designed based on some *ontological basis*. We propose to consider it as *a multi-dimensional knowledge-based specification (a set of system's knowledge slices)*, which represent an appropriate LIHS in following main *five projections*:

Technological (business) processes projection – it describes all technological processes, which are carried out and managed by appropriate LIHS, and can be considered as Retrospective Technological Parameters Repository (**RTP_R**) (it's first simplified model is presented as retrospective database in Section 3)

Software/Data Resource Topology projection – it has to define a legacy (current) software and data resources topology taking into account appropriate technological process structure. It describes existing software architecture solutions, related data structures (separate flat files, local databases, Web-data resources etc.) and their interfaces. It has to be constructed as Software/Data resources Topology Repository (**SDT_R**).

System User Profiles projection - it is a collection of user information profiles presented in legacy system, containing all their rights on information access and control statements, which have to be executed in order to provide observation and management of technological processes in LIHS. This facility has to be designed as a User Information Profiles Repository (**UIP_R**)

System Conflicts projection – it is a repository of system case study descriptions, where problem situations occurred during time of system operation have to be collected, their re-solving issues, and necessary system changes related to them have to be depicted, documented and analyzed. This kind of system information has to be captured in a Problem Case Studies Repository (**PCS_R**)

Problem Solutions projection – it includes all information about prospectively (target) solutions applicable/available for actually existing software/hardware problems occurred in legacy system, especially in form of reusable patterns of different types: architectural, technological processes, software, hardware, document patterns etc. It should be constituted in form of Target/Pattern Solutions Repository (**TPS_R**).

The collection of all these Repositories can be considered as a target **KBME**, and it could be given as:

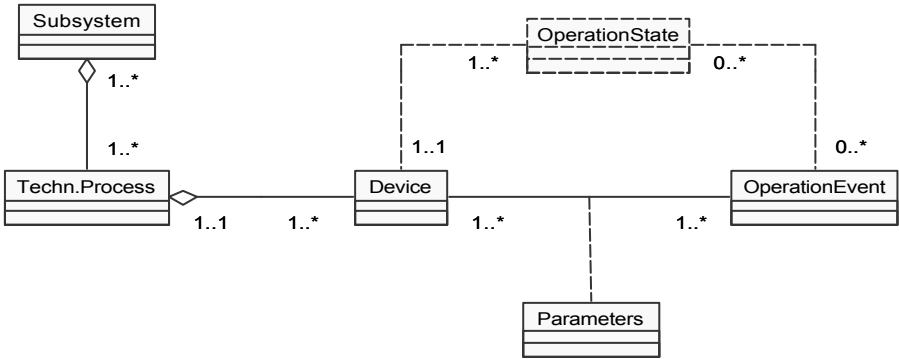
KBME RTP_R SDT_R UIP_R PCS_R TPS_R

All these repositories are finally understood by us as full-functionality implemented databases using object-oriented approach and appropriate information technologies. They allow us to create *current multi - dimensional description* of LIHS in **KBME** and *to modify continuously this one according to system maintenance and reengineering needs* through system description adding, deleting and updating in an appropriate repositories including in **KBME**.

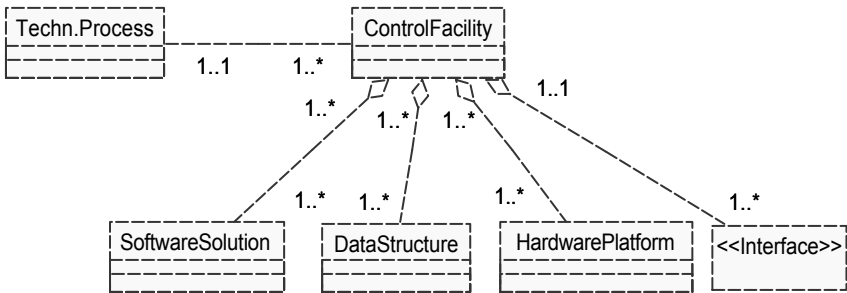
4.2. Local Ontology Specifications

Let us consider the possible specifications for each separate part of KBME conceptual model. Of course, we can now define only our first our proposal for it due to actually quite

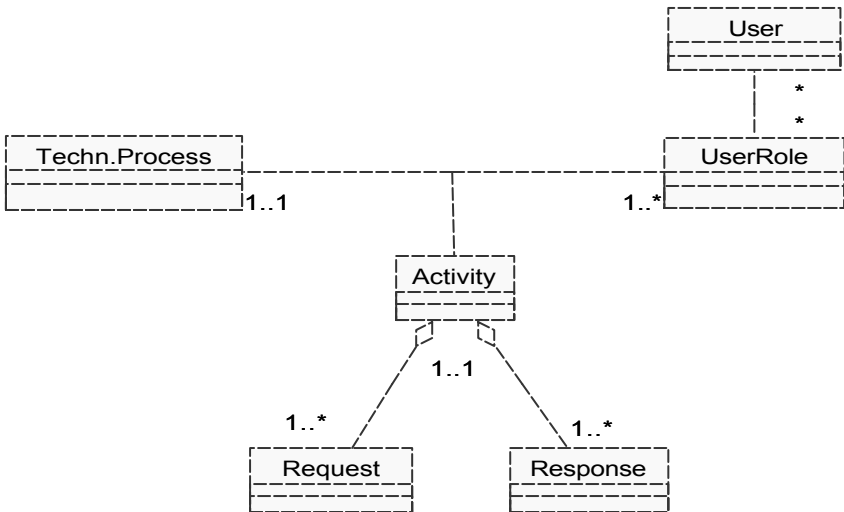
complex aggregation of information sources distributed in large LIHS, which have to be integrated into this model. We will use a “bottom-up” approach for this purpose, and at first we will construct the local ontology for each separate KBDM– projection presented



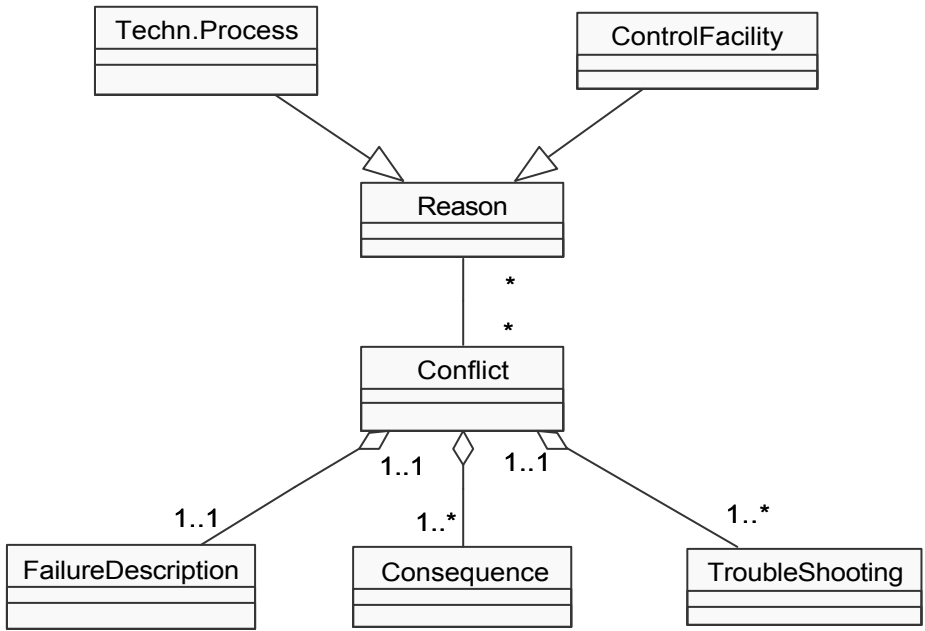
(a) RTP_R specification



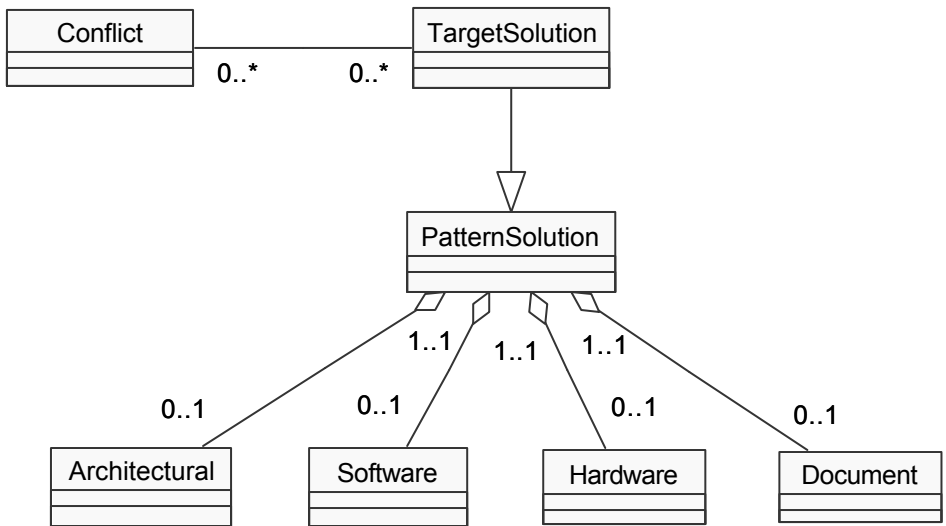
(b) SDT_R specification



(c) UIP_R specification



(d) PCS_R specification



(e) TPS_R specification

Fig. 5 UML class diagrams for **KBME** knowledge model specification

above. We will use for this description the UML - notation, which is actually recognized from many authors as a powerful tool for ontology representation, (e.g. in [GP00], [Ki99]), and we are starting with:

RTP_R ontology specification - it could be given as the set of the following classes and their relationships: *Subsystem* – *TechnologicalProcess* – *Device* – *OperationState* – *OperationEvent* – *Parameter*, and is presented as appropriate class diagram on Fig. 5, (a).

SDT_R ontology specification – it has to combine the following classes and their relationships: *TechnologicalProcess* – *ControlFacility* (*SoftwareSolution*, *DataStructure*, *Hardware Platform*, *Interface*), and is shown in class diagram form on Fig.5, (b)

UIP_R ontology specification - it has to be composed from the following classes and association classes: *Subsystem* - *UserRole* – *Activity* (*Request*, *Response*), the class diagram is depicted on Fig. 5, (c)

PCS_R ontology specification – it could be considered as scheme including the following classes and links: *TechnologicalProcess* – *ControlFacility* – *Reason* - *Conflict* (*Failure*, *Consequence*, *TroubleShooting*), its class diagram refers to Fig. 6, (d)

TPS_R ontology specification – it includes such classes as: *Conflict* – *TargetSolution* - *Pattern* (*Architectural*, *Software*, *DataStructure*, *Hardware*, *Document*), the class diagram is outlined on Fig., 5 (e).

Next, all these local ontologies have to be integrated (taking into account some existing semantic constraints) in the top-level structure, which represents the target knowledge model specification for **KBME**. By this consideration our main idea is to put into this model all kinds of information, which could be served as *knowledge-oriented control engine* for system maintenance process. As a result a system maintenance process in appropriate LIHS could be organized and has to be provided as a *permanent working cycle*, where a legacy (current) system architecture will continuously be changed and improved accordingly to any new system challenges influenced both external and internal factors.

5. Conclusions and Future Works

Let us briefly summarize some results and problems presented and discussed in our report:

We have considered some common software maintenance issues and approaches with expect for understanding its genesis and goals. We have determined two main factor groups influenced this process in domain-independent manner

We have also considered some specific maintenance problems, which are typical for large LIHS (legacy handling information systems), based some on the experience in the real-life maintenance and reengineering project in the field of Ukrainian gas-transport industry branch

The conclusion is made, that for such complex kind of legacy systems the maintenance process has to be provided continuously based on knowledge engineering in an appropriate LIHS.

In order to elaborate a conceptual solution for this problem the metaphor of Knowledge-Based Maintenance Environment (KBME) was introduced

The first knowledge model specification for KBME is designed and presented.

Obviously our concept represented now is actually rather "crude", and we would like to clarify that we see some very important and difficult questions, which have to be answered in future:

- a) how we have to elaborate some whole version of KBME conceptual knowledge model specification ?
- b) which modelling abstractions should be used in order to integrate and to operate with heterogeneous knowledge and data resources in KBME ?
- c) which software solutions and information technologies will be adequate for KBME effective implementation ?

These points will define our future works in this research domain and in its industrial applications.

Acknowledgements

The authors would like to express the special gratitude to prof. Heinrich C. Mayr from University of Klagenfurt, Austria, for his stimulating discussions of some problems related to the topics presented in this report. We would like also to thank the chief engineer A. Kuzmin from the design firm «PromAvtomatika» (Kharkiv, Ukraine), and to graduate students of National Technical University “KhPI” A. Veydyenyeyv, O. Rudenko, and R. Ismailov for their supporting by designing and implementation of our ideas.

Bibliography

[Br00] Bril R. et al.: Maintaining a legacy: towards support at the architectural level. In: Journal of Software Maintenance: Research and Practice, Volume 12, Issue 3, 2000; p.p. 143-170.

[Bo98] Box, D.: Essential COM. Addison-Wesley-Longman, 1998. – 421 p.

[Ch00] Reengineering Forum Europe 2000 (euroREF: 7th Reengineering Forum). Proceedings (edited by Elliot Chikofsky). Workshop on Web Site Evolution. Zurich, Switzerland, 29. February – 3. March 2000.

[CHP00] Carney, D., Hissam S., Plakosh D.: Complex COTS-based software systems: practical steps for their maintenance. . In: Journal of Software Maintenance: Research and Practice, Volume 12, Issue 6, 2000; p.p. 357-376.

[Cc01] Compressor Controls Corporation, URL:<http://www.cccglobal.com/> (accessed on April, 2001)

- [EV00] Proceedings of the Fourth European Conference on Software Maintenance and Reengineering (edited by J. Ebert and C. Verhoef). IEEE Computer Society Order Number PR00546. Zurich, Switzerland, 2000. - 241 p.
- [FH97] Fong, J., Huang, S.-M.: Information Systems Reengineering. Springer-Verlag, Singapore, 1997. – 250 p.
- [GP99] Granefield S., Purvis, M.: UML as Ontology Modeling Language. In: Proc. Of the IJCAI-99 Workshop on Intelligent Information Integration, Stockholm, Sweden. 1999.
- [KT01] Kuklenko, D., Tkachuk M.: Retrospective Database in Technological Control Systems – Concept and Implementation Issues (*submitted and accepted for the publication in the Proceeding of International Conference “MicroCAD’01”, May 2001, Kharkiv, Ukraine*)
- [Ki99] Kitchenham, B. A. et al.: Towards an ontology of software maintenance. In: Journal of Software Maintenance: Research and Practice, Volume 11, Issue 6, 1999; p.p. 365 – 389.
- [KR00] Knublauch, H., Rose, T.: Round-Trip Engineering of Ontologies for Knowledge – Based Systems. In: Proc. Of SEKE-2000 Twelfth International Conference on Software Engineering and Knowledge Engineering, Chicago, Illinois, USA. 2000; p.p. 239-247.
- [TME00] ReTIS2000: 6th International Conference on Re-Technologies for Information Systems (Preparing to EBusiness). Hrsg. von H. Thoma, H.C. Mayr., A. Erkollar. – Oesterreichische Computer Gesellschaft. Band 132. - 2000. – 183 p.
- [Mi98] Miller, H.: Reengineering legacy software systems. Digital Press, Boston, 1998. – 267 p.p.
- [Mo01] MODBus protocol: URL:// <http://www.win-tech.com/html/mbap.htm> (accessed on April, 2001)
- [OI00] Olsem M.: An Incremental Approach to Software Systems Re-Engineering. In: Journal of Software Maintenance: Research and Practice, Volume 10, Issue 3, 1998; p.p. 365 – 389.
- [Om01] URL:<http://www.omg.org> (accessed in March 2001)
- [Pi97] Pigoski, T.: Practical software maintenance: best practices for managing your software investment, John Willey & Sons, 1997. – 384 p.p.
- [RMS00] Rozgonuk V., Medvedeva L., Schevchuk. V.: Do pitannya pobudovi kompleksnoj sistemi keruvannya NAK “Naftogas Ukraine”. // Naftova ta gasova promislovist. 2000, 3, p.p. 52-53 (in Ukrainian).
- [ST00] Schach, S., Tomer, A.: A Maintenance-oriented Approach To Software Construction. In: Journal of Software Maintenance: Research and Practice, Volume 12, Issue 1, 2000; p.p. 25 – 45.

- [Se01] Serck Control Co. URL:<http://www.serck-controls.co.uk/> (accessed on April 2001)
- [St00] Staab, S., Erdmann, M., Maedche, A., Decker, S. An Extensible Approach for Modeling Ontologies in RDF(s). // Hrsg. H. Schmerck, D. Seese, W. Stucky, R. Studer. - Institute für Angewandte Informatik und Formale Beschreibungsverfahren, Universität Karlsruhe (TH), Bericht 401, Oktober 2000. – 10 S.
- [St00a] Studer, R., Decker, S., Fensel, D. Staab, S.: Situation and Perspective of Knowledge Engineering. In: Knowledge Engineering and Agent Technologies. IOS Press, 2000.
- [Te01] Tele-maintenance via the Internet, Siemens AG, 2001 Automation and Drives: URL:http://www.ad.siemens.de/fea/html_76/fhemden.htm (accessed on April 2001)
- [Tk99] Tkachuk N. et al.: Object - Oriented Structural Model and Program Package for Distributed Control System of Air Cooling Facility on Gas-Compressor Station. In: Printed Scientific Works of KhPU, Issue 73, Kharkiv, KhPU, 1999, p.p. 3-7 (in English).
- [Tk00] Tkachuk N. et al.: A Gas - Compressor Station Case Study in Software Re-engineering. In: ReTIS2000: 6-th International Conference on Re-Technologies for Information Systems (Preparing to EBusiness). Hrsg. von H. Thoma, H.C. Mayr., A. Erkollar. – Oesterreichische Computer Gesellschaft. Band 132. - 2000. – p. 142-153.
- [TS00] Tomer A., Schach, S.: The Evolution Tree: A Maintenance-Oriented Software Development Model. In : Proceedings of the Fourth European Conference on Software Maintenance and Reengineering (edited by J. Ebert and C. Verhoef). IEEE Computer Society Order Number PR00546. Zurich, Switzerland, 2000. - 241 p.
- [Vo99] Voas J.: Disposable information systems: the future of software maintenance? In: Journal of Software Maintenance: Research and Practice, Volume 11, Issue 2, 1999; p.p. 143 – 150.
- [Vo00] Voas J.: Disposable COTS-Intensive Software Systems (A Position Statement). In : Proceedings of the Fourth European Conference on Software Maintenance and Reengineering (edited by J. Ebert and C. Verhoef). IEEE Computer Society Order Number PR00546. Zurich, Switzerland, 2000. - 241 p.
- [Wa99] Warren, J.: The renaissance of legacy systems: method support for software-system evolution. Springer Verlag, London, 1999. – 182 p.p. (URL: <http://www.comp.lancs.ac.uk/projects/renaissance/>)
- [Wi01] WinTECH Software, URL: <http://www.win-tech.com/> (accessed on April 2001)