# A KNOWLEDGE BASED APPROACH TO TRACEABILITY AND MAINTENANCE OF REQUIREMENTS FOR INFORMATION SYSTEMS

*Mykola Tkachuk*[+], *Volodymyr Sokol*[*], *Heinrich C. Mayr*[*], *Mikhail Godlevsky*[+]

[+] National Technical University "Kharkiv Polytechnic Institute", 61002, Frunze Str., 21, Kharkiv Ukraine
[*] IWAS, University of Klagenfurt, A-9020, Universitaetsstr. 65-67, Klagenfurt, Austria
tka@kpi.kharkov.ua, vlad@ifit.uni-klu.ac.at, mayr@ifit.uni-klu.ac.at, god_asu@kpi.kharkov.ua

**Abstract**

Стаття презентує орієнтований на обробку знань підхід до аналізу та управління вимогами щодо інформаційних систем , який інтегрує в собі концепції, розроблені в різних предметних областях, а саме: в системах підтримки користувачів; в системах, що базовані на використанні прецедентів; а також на методах моделювання систем керування технологічними процесами. Цей підхід дозволяє накопичувати як  первинні так і подальші системні вимоги в структурованій та доступній формі, а також робить прозорими процеси прийняття проектних рішень та розв'язання проблемних сітуацій.

The paper presents a knowledge based approach to requirements analysis and management which integrates concepts developed in different domains, namely that of user support systems, case based reasoning, and process control system modeling. By this approach, initial and changing requirements, design decisions and problem situations during production become transparent and accessible in a systematic fashion. I.e., it leads to a comprehensive knowledge base of all relevant aspects of an information system.

## 1. Introduction

In the last decade many approaches have been introduced for collecting and organizing knowledge and experience gained during requirements engineering phase of a project (from elicitation via analysis to specification and change). Although these approaches provide guidelines and tips for capturing requirements re the information system to be developed, there is a lack of a knowledge based framework for handling decisions made by analysts during this phase. Such information, however, is not only very useful but even necessary in the subsequent phases, when choices between realization alternatives have to be done during implementation or changes have to be made in the course of maintenance or reengineering projects. Consequently, transparency of the requirements engineering stage will help to build more robust and maintainable information systems.

CASSAM (Computer Assisted Software Maintenance and Support) is a framework for the support and maintenance of information systems [1,2]. It consists of the three main parts:

- Software Configuration Management,
- Software Maintenance,
- Software Support.

The CASSAM approach is based on case-based reasoning (CBR), a methodology used in the domain of artificial intelligence for problem solving and learning. CBR tries to apply experience and knowledge from previously successfully solved situations to solve a given new situation. Various helpdesk systems were developed by utilizing CBR (HOMER [3,4], etc.). Clearly, software system requirements support and maintenance (S&M) fits these conditions and, therefore, is a promising domain for the CBR-based CASSAM approach. This paper extends the CASSAM software support model by adding some new features proposed for space-oriented and trajectory-based (SOTB) approaches to system requirements engineering. In this way we try to take into account some environmental facets of requirements engineering process, and to elaborate an appropriate geometrical interpretation for modeling and investigation issues in this domain.

The paper is structured as follows: *Section 2* focuses on some existing CBR and SOTB approaches to system requirements. In *Section 3* we discuss the appropriateness of the CASSAM concept for an effective requirements support. *Sections 4* and 5 introduce a modeling concept for the Multi-dimensional Information Space (MDIS), which provides an integrated database for SOTB approaches. *Section 5* addresses some implementation issues regarding the first version of a special CASE subsystem for MDIS maintenance and supervising. *Section 6* concludes with a short outlook on future work to be done within that field.

## 2. Related work

The CASSAM concept, as was mentioned before, is an approach developed by Institute of Business Informatics and Application Systems at the University of Klagenfurt in the 90ies and dedicated to support and maintenance systems. Its main goal is to provide a framework for supporting the productive phase of technical and information systems. Based on this concept, a product "Unicontrol Helpdesk" has been developed by the Austrian company Software Trading. The main elements of the rather complex CASSAM reference model are the notions *symptom*, *diagnosis* and *therapy* which are related to so-called *reference objects*. Symptoms may be aggregated and related to observations in concrete situations which come with *user reference objects* as instances of reference objects. The case based reasoning process is supported by *rules* and *weights* helping to find the best fitting cases. For a first cross analysis in a given case, *case prototypes* are derived and continuously adapted from sets of "similar" ones thus accelerating the solution search process in an on-line Helpdesk/Hotline situation. Whereas CASSAM focuses on the productive phase we now also want to support the development process as well by capturing knowledge about design designs, implementation problems etc..

Most related work to this goal was done by the Department of Computer Science at the University of Kaiserslautern [5] in the context of the INRECA project. One of the main results of this project is a methodology developed for creating CBR applications. As proof of concept the help desk system HOMER [3,4] was developed. In our work, however, we try not to limit ourselves to CBR applications but aim at establishing a framework for supporting the whole lifecycle of information system.

CBR is an AI approach to problem solving which is based on the suggestion that similar problems have similar solutions. Thus, the idea of CBR is to acquire experience from solved problems, to model and store this knowledge in a so-called case base, and to adapt and reuse it for a new case. In order to increase the effectiveness of the solution finding information about the appropriateness of derived solutions is stored as well. The CBR process is sketched in figure 1.

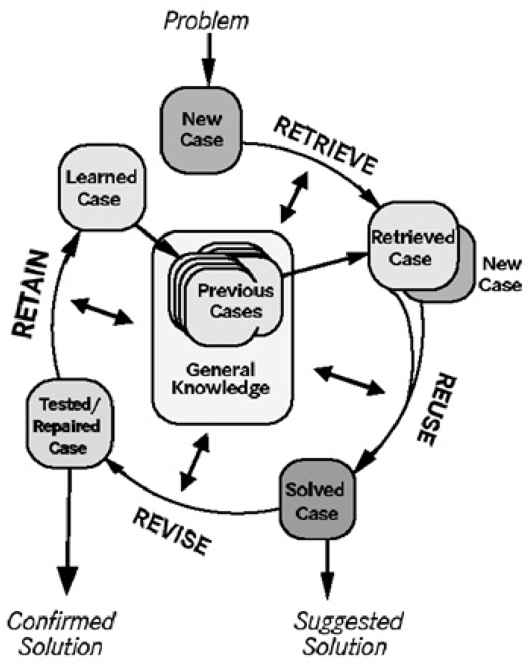


Figure 1: CBR Cycle (according to [4])

We may distinguish four main steps in problem solving using CBR: *Retrieve*, *Reuse*, *Revise* and *Retain* (the so-called R4-Cycle). During Retrieve the most similar case(s) is (are) extracted from the case base. During Reuse information and knowledge from retrieved cased is used to form a solution for the new case to be solved (called *solved case*). On the Revise stage the applicability of a new solution is tested (e.g., on real world objects) and repaired, if possible, if the application of the case fails. Finally, during the Retain stage the new experience and knowledge gained from problem solving is stored in the case base.

SOTB approaches to software system design and system evaluation are generally well-established in the domain of software engineering and applied informatics. One of the earliest proposed concepts is the so-called "*space of programs*" [6] stretched by the three dimensions "Contents", "Dimension" and "Complexity" of an application domain (see Fig. 2). "Dimension" leads to the development of *program libraries*, "Complexity" leads to the development of *closed program systems*, and "Contents" supports the creation of *open program systems*. The limits of this approach are evident, since an evaluated program system is considered as a simple point, without taking into account its structure and the system's dynamics within the time dimension.

[7] proposes a geometric interpretation of the software development process by introducing a *trajectory*, which is build sequentially in three 3-dimensional subspaces. In contrast to the previous model, an appropriate software system itself is considered as an aggregate of data structures, algorithms and information technologies (see Fig. 3). However, it is obvious that the factors of evolution in time, and system environment are not covered by this space model.

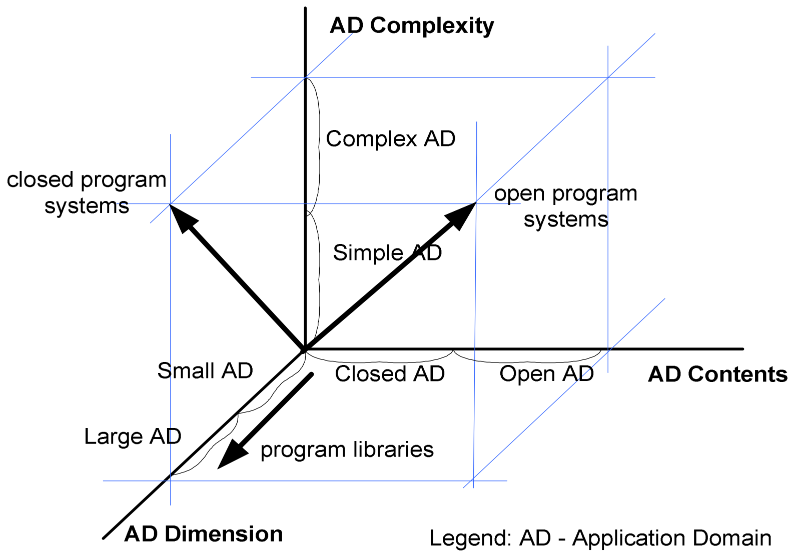


Figure 2: The “space of programs” [6]

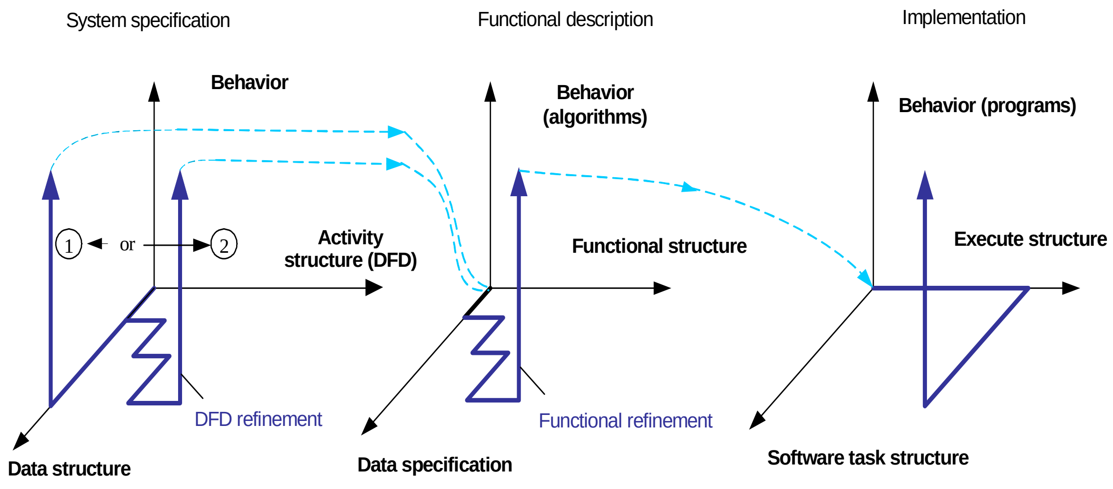


Figure 3: Aggregate of three 3-dimensional “development spaces” introduced in [7]

[8] tries to eliminate those deficiencies. In this paper some questions of system requirements traceability are discussed in relation to the planning of a so-called prospective software reengineering. For this purpose, an appropriate requirements trace is constructed in a 3-dimensional space: *system architecture–Environment-Time* (see Fig. 4). However [8] does not provide any detailed interpretation of the given space dimensions so that it is hard to define metrics within this space, etc.

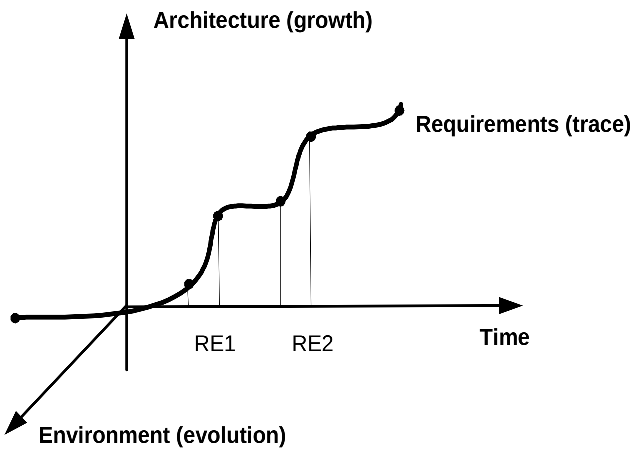


Figure 4: Three-dimensional system requirements tracing [Neu02]

This investigation shows that all the mentioned approaches are not sufficient to deal with essential facets of information systems. Nevertheless, they may be used as a basis for the extension related to new SOTB approaches in order to capture also system requirements engineering aspects which are an important issue in nowadays informatics.

## 3. Extending the CASSAM concept to requirements maintenance

As has been pointed out, the main goal of the CASSAM approach was to support the maintenance and support of productive software systems. The process supported by CASSAM is outlined in Fig. 5. However, the model presented in [2] may be used as a basis for extending it in a way such that also system requirements may be dealt with including different versions, their change and even their prediction (preventive maintenance). The corresponding process is depicted in Fig. 6.

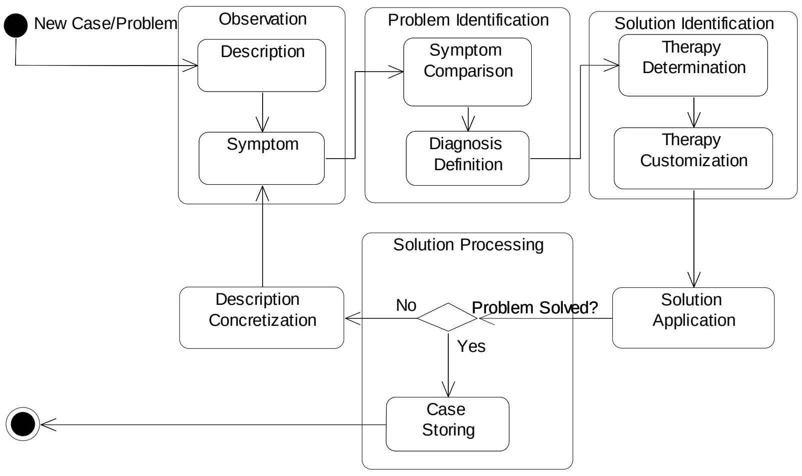


Figure 5: Sketch of the CASSAM support process model

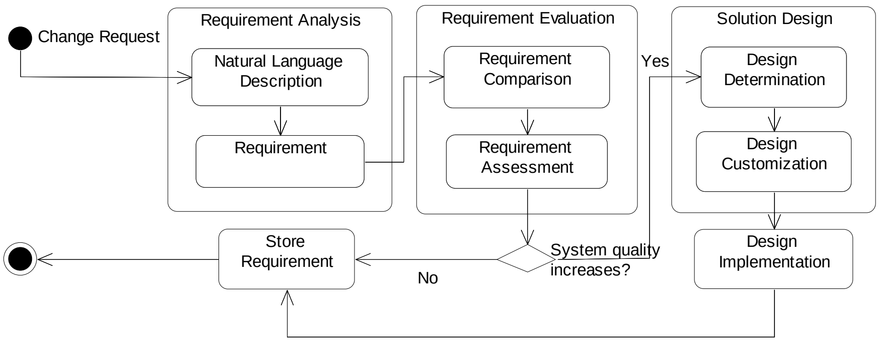


Figure 6: Sketch of the requirements maintenance process

The requirements maintenance process starts with a new change request, which comes usually in natural language and thus should be formalized. After that transformation the new requirement is compared to the content of the knowledge base of requirements. After that, the new requirement is assessed in order to define whether its implementation could forward the system into a better state (see next chapter) or not. In both cases the new requirement is stored together with the assessment result for future use, and in the former case the development process is continued by design and implementation.

The main difference between these two processes is that in the case of support process we evaluate usefulness of a solution after applying it, however in the requirements maintenance process we have to assess utility of a new requirement prior to its further development. Thus we need a model to make such assessment available. Such kind of model will be discussed in the next section.

The next step is to customize CASSAM model in order to provide it ability to handle requirements maintenance. Accordingly to changes in process model, the sketched CASSAM model for requirements maintenance process is presented in Fig.7.

In compliance with the process model, we have a Natural Language Description (NLDescription) class, which represents requirement in natural language form. The task of extraction information from natural language could addressed to such projects as NIBA [14]. The extracted information is stored in Requirement class. The Utility class represents evaluation of requirement's usefulness to the system in development. Such evaluation may be done using

Evaluation Model, which is based on Goal(s) of Information Systems. Each Requirement has a relation to other requirements in a knowledge base, which is expressed as a Similarity class. Similarity assessment may be done comparing Utilities and NLDescriptions of requirements. The Design Pattern and Implementation classes are used to reflect solutions for requirements. It could be the case that Implementation causes other requirements to appear.

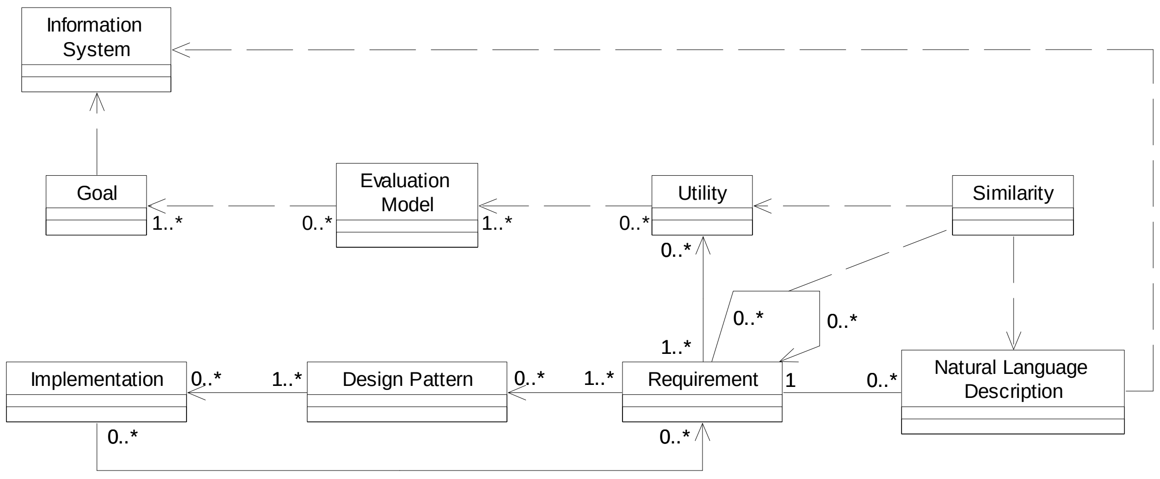


Figure 7: Sketch of the CASSAM model for the requirements maintenance process

Thus we described the process model for requirements maintenance and provided a sketch of a CASSAM model customization for handling this process as well.

## 4. Trajectory-based approach for system requirements tracing and maintenance

We proceed based on the following assumption: For both, the design of a new system and for the reengineering of a legacy system, it is necessary to collect, to analyse and to use information about the respective system's design requirements, about the parameters of the system functionality, and about some prospective solutions for the system's modification. All this information usually is included in non-structured and often fragmental form in such sources as:

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

In order to integrate all information captured from these resources within an appropriate storage structure supporting analysis and exploration, the metaphore of *Multi-dimensional Information Space* (MDIS) was proposed in [9] for the domoain of information process control systems. MDIS comes with the following five system description projections:

(1) *Technical Subsystems–Technological Processes*: defines the modular structure of an appropriate information process control system,

(2) *Technological Processes–Control Facilities*: describes the infrastructure of the soft- and hardware infrastructure, which is used for process control functions,

(3) *Control Facilities–User Information Profiles*: captures the information needs and requests of all system user groups,

(4) *User Information Profile–System Conflicts*: focuses on problem situations and errors, which arise during system operation,

(5) *System Conflicts–Solution Patterns*: depicts possible solutions for system problems.

These projections have to be specified carefully by taking into account the given system domain. They may be presented in the form of a detailed UML class-diagrams. Fig. 8 shows an example of such a diagram, for more details the reader is referred to [9].

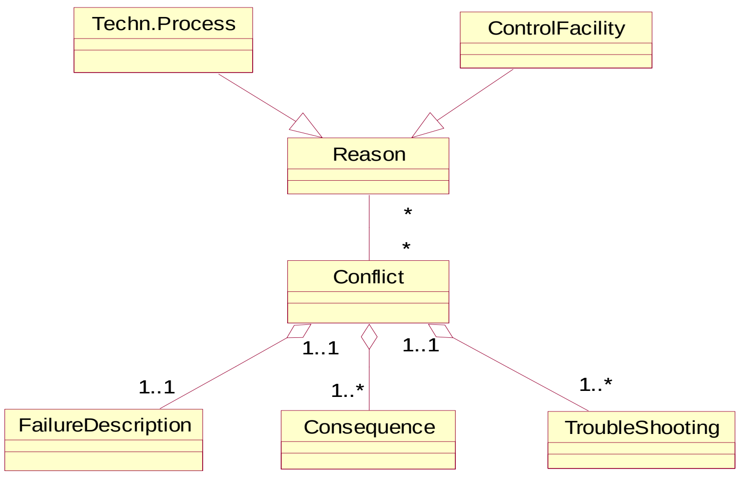


Figure 8: UML class diagram specifying MDIS projection (5)

This approach is now extended by the time dimension as depicted in Fig. 8 which represents the superposition of five 3-dimentional subspaces corresponding to the afore mentioned projections and each of them extended by the time axis. Consequently, at each instant *t* of time *T* (*T* corresponding to the system life cycle) the dynamic information model of a (process control) system can be broken down to the coordinates of the subspaces, e.g. to a plane in subspace *User Information Profiles–System Conflicts – Time.*

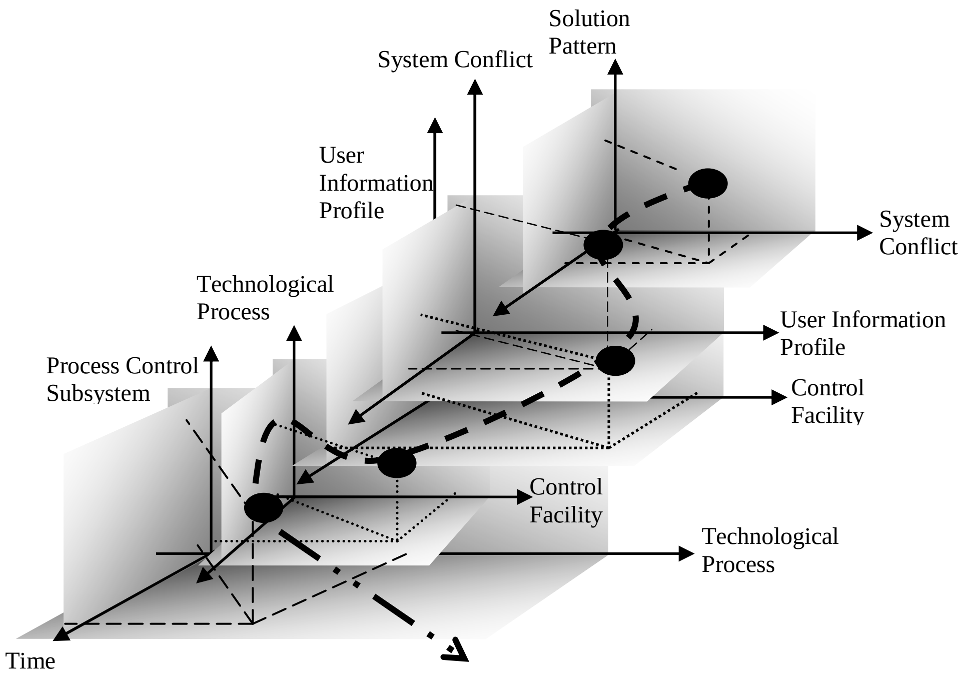


Figure 9: Spatial interpretation of the MDIS model w.r.t requirements traceability and evaluation issues

## 5. Formal discussion and extension

The collection of these planes $P_j^{(t)}, j \in [1,5]$ which are built at the each of the five subsets can be considered as a *hypersurface* $\Omega^{(t)} \Rightarrow \bigcup_{j=1}^{5} P_j^{(t)}$ in the MDIS. Such a hypersurface includes a lot of possible *phase paths (trajectories)* which represent possible evolution scenarios for the system to be modeled (one of such phase trajectory is shown in Fig. 9 as the bold dotted line). The metrics in MDIS can be defined as a space function $d(a,b)$ for two different points $a$ and $b$ in MDIS, where $d(a,b) \geq 0$, $d(a,b) = 0$, if $a = b$, $d(a,b) = d(b,a)$ and $d(a,b) \leq d(a,c) + d(c,b)$. Thus, the $D_E$ (*Euclidean distance*) between any two points in the MDIS can be culculated by $D_E = \sqrt{(a_i - b_i)^2}$. Based on that metric it is possible to build the *nearness criterion* for different system trajectories, and to elaborate some algorithms and procedures for the movement from a current system trajectory to a target system trajectory having appropriate parameters in one or more of the subspaces (1)-(5), e.g., in subspace *System Conflicts–Solution Patterns-Time.*

Another important facet of the spatial interpretation of our approach to system requirements traceability and maintenance is the adequate view on the dynamics issues of requirements evolution at the each “system point“ placed at the appropriate system trajectory. We refer to the modeling metaphor for requirements engineering (RE) processes given by Pohl [10] which considers an RE-process to establish a *trajectory* within the 3-dimensional space stretched by the three co-ordinates (see Fig. 10):

- *Representation:* indicates the degree of formality of the representation of given requirements (informal, semi-formal, formal),
- *Specification:* defines the degree of requirements completeness (initial, fair, full),
- *Agreement:* indicates to which degree the stakeholders agree from their points of view to the requirements (partial, common).

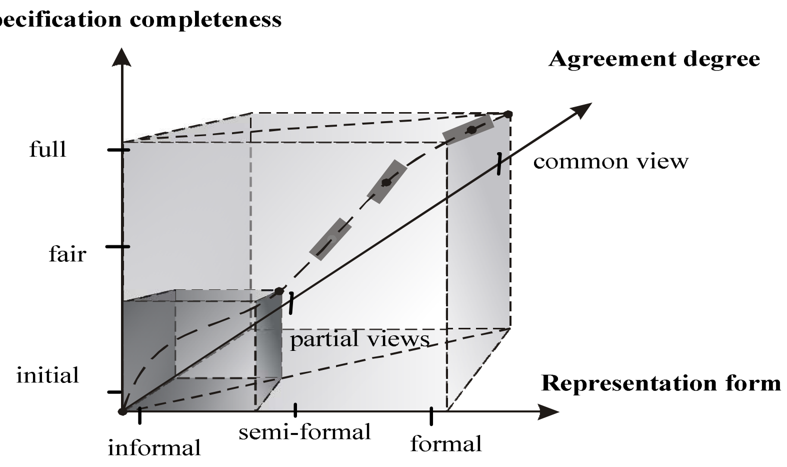


Figure 10: RE process dimensions according to [10]

Our idea is now to combine Pohl’s approach with ours in order to facilitate system requirements processing by moving along some pre-defined paths selected in MDIS.

## 6. Feasibility study and some implementation issues for MDIS

The MDIS framework is based on our experience collected during the last 3 years by performing a number of real projects [11-13]. These projects were aimed to the development of information handling systems for technological process control (IHS-TPC) at gas-branch enterprises placed in the Kharkiv region. Each such IHS-TPC is implemented as a Web-based SCADA (Supervisory Control and Data Acquisition) system solution [13].

The next step will consist in connecting these separate IHS-TPC by a Web-based regional dispatching information system (RDIS), which will allow:

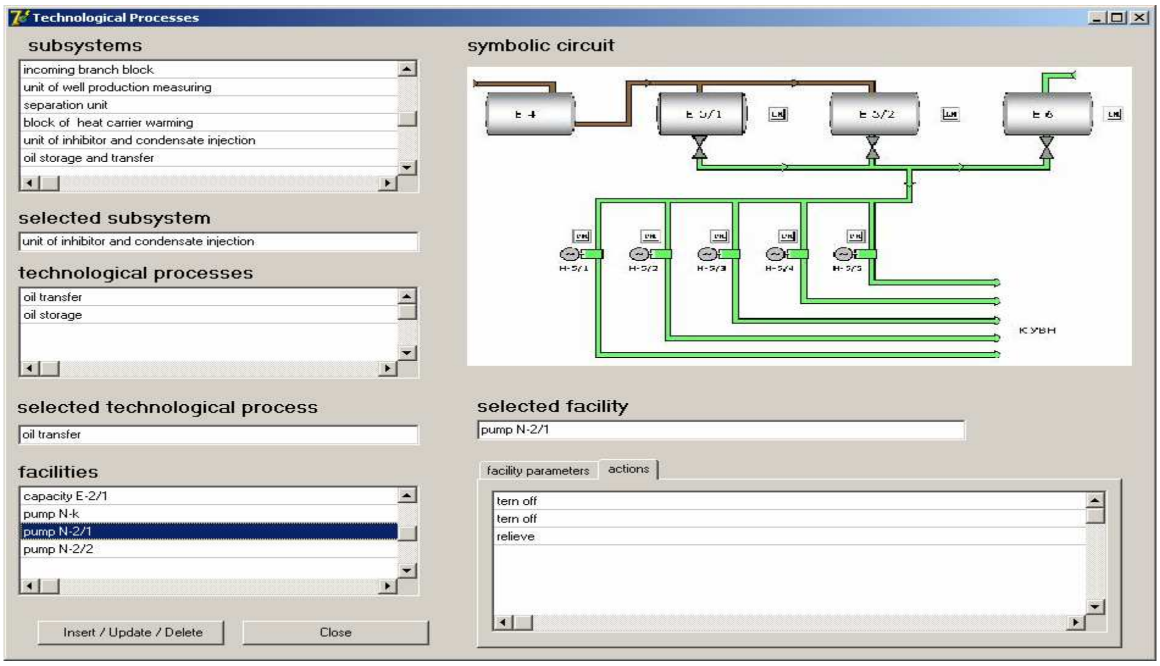
- to capture, to store, and to analyze all the data which are necessary for day-to-day and strategic decisions,
- to support the hot links between system users (e.g. an operator in a gas-extraction node can obtain technical advise information from an expert in the regional management center),
- *to provide continuous and quick access to the required information related to all technological processes and objects of the regional system. Such access will be granted in an authorized mode by each node to any kind of communication devices (laptops, mobiles).*

Obviously, the degree of complexity of a RDIS is high, so that the MDIS framework combined with the CBR-approach will be useful and needed for both, system requirements engineering and system support and maintenance.

In order to utilize the MDIS model as a real information basis for system requirements traceability and maintenance, a special CASE-system has to be developed. This system should service as a tool for the following tasks to be performed:

- to construct an appropriate data scheme for each of the (1)-(5) MDIS–projections listed in Section 4,
- to furnish these structures with real system data,
- to support the mapping between several coordinates (subspaces) in MDIS.

A first version of such a CASE-tool has been developed. It supports the description of separate information projections in MDIS. Some GUI-examples are shown in Fig. 11 (a), (b).


Technological Processes

subsystems

- incoming branch block
- unit of well production measuring
- separation unit
- block of heat carrier warming
- unit of inhibitor and condensate injection
- oil storage and transfer

selected subsystem

unit of inhibitor and condensate injection

technological processes

- oil transfer
- oil storage

selected technological process

oil transfer

facilities

- capacity E-2/1
- pump N-k
- pump N-2/1
- pump N-2/2

symbolic circuit

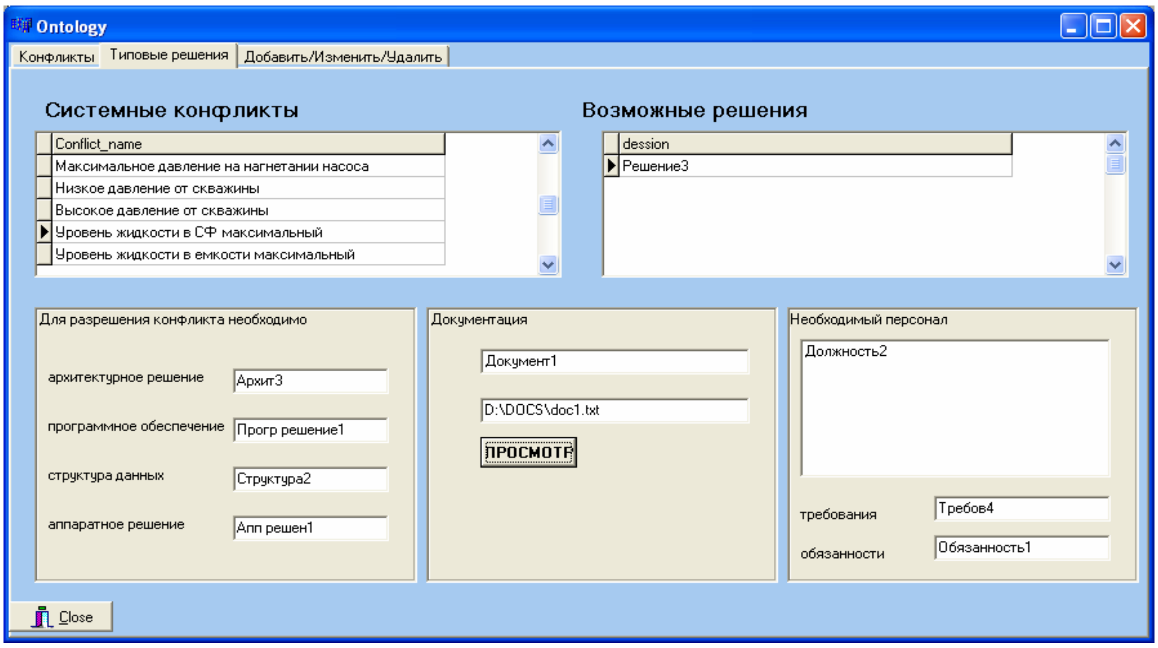
selected facility

pump N-2/1

facility parameters | actions

- tern off
- tern off
- relieve

(a)


Ontology

Системные конфликты

| Conflict_name |
|---|
| Максимальное давление на нагнетании насоса |
| Низкое давление от скважины |
| Высокое давление от скважины |
| Уровень жидкости в СФ максимальный |
| Уровень жидкости в емкости максимальный |

Возможные решения

| dession |
|---|
| Решение3 |

Для разрешения конфликта необходимо

| | |
|---|---|
| архитектурное решение | Архит3 |
| программное обеспечение | Прогр решение1 |
| структура данных | Структура2 |
| аппаратное решение | Апп решен1 |

Документация

Документ1

D:\DOCS\doc1.txt

Необходимый персонал

Должность2

| | |
|---|---|
| требования | Требов4 |
| обязанности | Обязанность1 |

(b)

Figure 11: CASE–tool screen shots (showing multi-language support)

Fig. 11, (a) presents the user’s dialog for the data relating to the MDIS-projection *Technical Subsystems-Technological Processes*. Fig. 11, (b) is given in Russian and illustrates the use case for the MDIS-projection *System Conflicts–Solution Patterns* (*Системные Конфликты-Типовые Решения*).

## 7. Conclusions and Future Work

The extension of the CASSAM approach and its combination with the MDIS model form a basis for an in-depth analysis of complex information systems. Initial and changing requirements, design decisions and problem situations during production become transparent and accessible in a systematic fashion. I.e., our approach leads to a comprehensive knowledge base of all relevant aspects of an information system. Nevertheless, further research is necessary to exploit the full power of the case-based reasoning methodology: e.g., further measures as well as weights have to be defined for the comparison of requirements in order to support the evaluation process. Another example is the need for establishing two- and three-dimensional relationships between requirements, design decisions and symptoms/observations as well as their connections to the initial MDIS projections.

## Acknowledgments

We have to thank our students A. Zemlaynoj, T. Zavinskay, and R. Gamsaev of the Department of Information Management Systems at the National Technical University “Kharkiv Polytechnic Institute”, who participated in implementing the first version of the MDIS tool.

## References

1. Mayr, H.C.; Khouri, B.: CASSAM: An Approach to Computer Aided Software Support and Maintenance. In (Györkös, J. et al. Hrsg.): Proc. of the 2nd Conference on Re-Engineering of Information Systems, Bled, 1993, pp. 33-48.
2. Khouri, B.: Ein Konzept für den Einsatz fallbasierter Diagnosemethoden im rechnergestützten Softwaresupport. Dissertation, Johann Kepler University Linz, 1996 (in German).
3. Göker M., Roth-Berghofer Th., Bergmann R., Pantleon T., Traphöner R., Wess S., Wilke W.: The Development of HOMER: A Case-Based CAD/CAM Help-Desk Support Tool. In (Smyth B. & Cunningham P. eds.): Advances in Case-Based Reasoning, Proc. 4th European Workshop on Case-Based Reasoning EWCBR98, Dublin, 1998. LNAI 1488, Springer-Verlag, Berlin, 1998 pp. 346-357.
4. Göker, M., Roth-Berghofer., T.: Development and utilization of a case-based help-desk support system in a corporate environment. In (K. D. Althoff, R. Bergmann and L. K. Branting, eds.) Proc. 3rd International Conference on Case-Based Reasoning, Springer-Verlag, 1999, pp. 132-146.
5. Bergmann, R.: Highlights of the European INRECA Projects. In (Aha &Watson eds.): Proc. 4th. International Conference on Case-Based Reasoning, ICCBR-2001. Vancouver, Canada, Springer-Verlag 2001.
6. Bachmann, P. et al. Programmsysteme: Anwendung, Entwicklung, Fundierung. - Akademie Verlag, Berlin. 1983. (Бахманн П. и др. Программные системы / Пер. с нем. - М.: «Мир». 1988. 287 с).
7. Ward, J.: Strategic planning for information systems. John Wiley & sons, 1997.
8. Neumoin V.: Software Requirements Traceability in Reengineering Perspective. In: “Problems of Programming”, 2002 No 1-2, pp. 91-97 (special issue of Proc. 3rd International Conference “UkrPROG’2002”) .
9. Tkachuk, N.; Kuklenko, D.; Ovasapov, S.; Shekothyhin, K.: Knowledge-based Maintenance Environment for Large Information Handling Systems. Lecture Notes in Informatics (LNI), GI-Edition, Bonn, Volume P-2, 2001, pp. 139–154.
10. Pohl, K.: A Process-centered Requirements Engineering Environment. Dissertation, RWTH Aachen, 1995.
11. Tkachuk, N.; Mayr, H.C.; Kuklenko, D.; Godlevsky, M.: Web-based Process Control Systems: Architectural patterns, Data Models, and Services. Lecture Notes in Computer Science (LNCS 2510), GI Edition, Berlin 2002, pp. 721-729.
12. Tkachuk, N.V.; Kuklenko, D.V.: Usage of the SCADA-Concept for Intellegent Data Engineering in Process Control Systems. “Management Information Systems and Devices”, All-Ukr. Sci .Interdep.Mag 2002, Issue 121, pp. 129-136.
13. Tkachuk, N.; Shekhovtsov, V.; Kuklenko, D.; Mayr, H.C.: An Approach to Efficient Data Handling in Web-based Process Control Systems. In (Hamza, M.H. ed.): Proc. International Conference on Intelligent Systems and Control. Salzburg, Austria, 2003. ACTA Press, pp. 242-247.
14. Günther Fliedl, Christian Kop, Willi Mayerthaler, Heinrich C. Mayr: Das Projekt NIBA Zur automatischen Generierung von Vorentwurfsschemata für die Datenbankentwicklung. Papiere zur Linguistik Nr. 55 (Heft 2, 1996) Gunter Narr Verlag Täbingen (in German).