2002

Modelling Tools for Life Cycle of Pedagogic Information System at Tu Kosice, Slovakia

Zdenek Havlice
Technical University of Koslice, zdenek.havlice@tuke.sk

Gabriela Janockova
Technical University of Kosice, gabriela.janockova@tuke.sk

Pavol Sutor
Technical University of Kosice, sutor@pobox.sk

Jan Genci
Technical University of Kosice, jan.senci@tuke.sk

MODELLING TOOLS FOR LIFE CYCLE OF PEDAGOGIC INFORMATION SYSTEM AT TU KOŠICE, SLOVAKIA

Zdeněk Havlice

Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic
phone: +421 55 602 2210
Zdenek.Havlice@tuke.sk

Gabriela Janočková

Institute of Computer Technology, Technical University of Košice, B. Němcovej 3, 042 00 Košice, Slovak Republic
phone: +421 55 602 5131
Gabriela.Janockova@tuke.sk

Pavol Šutor

Institute of Computer Technology, Technical University of Košice, B. Němcovej 3, 042 00 Košice, Slovak Republic
phone: +421 55 602 5136
sutor@pobox.sk

Ján Genči

Institute of Computer Technology, Technical University of Košice, B. Němcovej 3, 042 00 Košice, Slovak Republic
phone: +421 55 602 5138
Jan.Genci@tuke.sk

ABSTRACT

The contribution presents a brief characteristic of structural and object models used for development and maintenance of information systems. It also describes utilisation of these models for generation of pedagogic information system IS STUDENT at the Technical University in Košice. It describes possibility of application of UML language (Unified modelling language) in life cycle of pedagogic information system. On practical examples it demonstrates the importance of modelling of such a system.

1 INTRODUCTION

Technical University in Košice belongs to one of few Slovak universities running unified pedagogic information system - IS Student at full-university base. At the moment 36 database clients are installed at study departments and residential halls. The web applications are used by more than half of teachers

and students representing more than 700 teachers and 8,000 students. IS Student is based on modern technologies (SQL, client/server architecture, WWW interface); however, it still has some problems during its operation. Drawback of the system is that it has not encompassed all detail specific requirements of individual users during its initial development and its function was only extended gradually. This negatively affected internal system architecture resulted in hindered integration of some requirements defined by users. The main effort of our group (five people), besides maintenance of the system, is to develop a new pedagogic information system (PIS) of higher quality by using our five years experience with the system and experience with processing of user requirements. The development of a new pedagogic information system started at the Technical University of Kosice in January 2001 and we plan to finish it in December 2002. An important role in the development of a new system plays modelling and relationships among models that is the main and efficient tool of life cycle of programme systems (we consider information systems). The reasons for using the system modelling tools are [3]:

- Simplification of communication among individual teams participating in project and mainly communication with user
- Early error detection and elimination
- Possibility of automation for transition from some models or their parts to implementation (generation of scripts, source texts and patterns based on models)
- Maintenance of system documentation at higher abstract level based on models
- Use of models for administration and maintenance of running system

2 SYSTEM DEVELOPMENT METHODOLOGY

The life cycle of programme systems is not only characterised by phases and their contents but also by mutual succession of phases, conditions of phase termination, used methods, tools and methodologies. These properties of life cycle define the so called model of life cycle. The choice of life cycle depends not only on system properties (its extend, complication, safety requirements etc.) but also on properties of objects sharing life cycle (organisation, working team, time and finance capacities etc.).

The SwLC models define succession of individual SwLC phases and they are used for planning, control and realisation of pre-project and project works, administration and maintenance of programme system. The most frequently used SwCL systems are: water fall model, prototype model, investigator model and spiral model.

	Advantages	**Drawbacks**	**Application**
Waterfall model	Termination of one phase and start of following one is precisely defined (results of terminated phase are considered as 100% correct and they represent starting point for the following phase)	1 Model is documentation-driven - the specifications exist only in "not executable" form, usually on paper, the client therefore cannot really understand what the product itself will be like 2 Duration of some phases of SwLC can be very long until these phases have to be finished for all developed system. The beginning of the next phase depends on termination of previous one. It may cause undesirable time lags	By solution of often repeated similar programme systems which are not very extensive
Prototype model	Possibility to test some properties of future system on prototype resulting in early detection of possible shortages and faults in analysis and suggestions	1 Fast production of prototype may induce a false imagination of customer about fast termination of the development of the whole system (so rapidly as prototype) 2 Problem may also arise by absence of suitable analytic documentation of project system at higher abstraction level	For formation of less extensive systems

Investiga-tor model	Permanent formation and testing of functioning programme system providing instant possibility to remove shortages	Several time reworked of complete programme system may be financially and time demanding	For solution of not very extensive systems with no experience with implementation so far
Spiral model	Development of programme system occurs along spiral, it cyclically passes through all phases. After termination of one part of the system development it passes to another one	1 Gradual extending of application in other parts requires proper choice of pilot project, consistent modularity and prediction of properties of future modules 2 It is very important to sustain updated project documentation	For solution of not very extensive systems with no experience with implementation so far

Tab. 1: Advantages and drawbacks of SwLC models [3].

PIS has already been developed for several years. Based on requirements of users continually new modules have been implemented into it. Its development is best represented by spiral model.

3 ACTUAL VALID MODELS FOR ANALYSIS AND PROPOSAL OF PROGRAMME SYSTEMS

Currently structured proposal techniques of the system include tools and procedures for formation of essential model of information system [6]:

1 **Function model** expressed by DFD (data flow diagram) and mini specification of elementary functions

2 **Data model** expressed by ERD (entity - relationship diagram) and conceptual data scheme

3 **State system model** expressed by STD (state-transition diagrams)

and for implementation model of the system:

4 **Model of programme system architecture** expressed by Structure Chart and module specifications - data structure diagrams, data navigation diagrams.

All these models are independent. They are mutually linked and by use of relationships among them it is possible to find inconsistencies or errors in analysis or system synthesis. Data Dictionary, containing all additional data about the system, is also an integral part of the proposal.

Data Flow Diagram (DFD) serves as a graphic tool of proposal and display of function model of system. It describes system from the viewpoint of running processes and provided functions and services. Synonyms for DFD are also terms like bubble diagram, process model, function model and work/flow diagram. The system model expressed by DFD diagrams has a tree (hierarchic) structure. At the top of hierarchy - DFD level 0 only one, the so called context diagram, occurs (it contains entire system represented by one function). It represents boundaries of the system and all sources and destination places (terminators) of data. If we continue in an internal structure of this individual function we passcome to subsequent lower level. This level contains basic functions of the system (subsystem) and their relationships expressed by data flows and data stores. The lowest level contains elementary functions, which are not necessary to decompose.

Entity Relationship Diagram (ERD) is a graphic tool used for to expressing of data model (data objects preserved in the system and relationship between them). The model elements comprise entities, relations, attributes and keys. Primary keys serve for unequivocal identification of entities. Implementation by relation database system is natural for implementation of relation model. In the relation database system the table responds to entity, attributes are columns and relation is realised by

the so called foreign keys. Foreign keys comprise references for further occurrence of the same or other entity.

Several representations of entity - relation diagrams and corresponding methods exist for relation data modelling.

State - Transition Diagram (STD) can be used for modelling of system behaviour, which depends on time. These diagrams originate from formal mathematical instrument - final state automata. It is a hypothetical machine, which may occur in one of the final number of states and the state may be changed by some event. The state diagram serves for graphic representation of behaviour of final state automata.

Structure chart (SCD) is a tree diagram with evaluated edges and its elements consist of nodes (module, pre-defined module, selection, iteration), edges (sequence of module processing), evaluation of edges (data flow, control flow). The root of the SCD tree is a node of part of the system whose structure is described. The diagram of modular structure models in detail modular structure of the system. Non-leaf (non-terminal) nodes represent higher activity abstraction, leaf nodes correspond to particular modules providing required activity. The edges are evaluated by transmitted or accessed information

Data structure diagram (DSD) is used for representation of hierarchic data structure.

The system modelled by **classes** and **objects** has several assets:

- It removes divergence between data model (DM) and function model (FM)
- Object can be closer to reality as analytic terms of data entities and functions. The verification of model correctness is simpler.
- Straightforward transition from objects in analytic models to objects in implementation by languages of object-oriented programming.

According to trend of modelling language unification result of methodology unification coming out from works of G. Booch, I. Jacobson, J. Rumbaugh - Unified Modelling Language - is perspective.

Unified Modelling Language (UML) is a standard modelling language for software - language for visualisation, specification, construction and documentation of software systems. It represents a set of the best modelling tools whose quality was tested by a long-time practise during suggestion of complex systems.

Advantage of UML [4]**:**

1 It provides easily operating, visual modelling language by which it is possible to develop and change model sense
2 It enables system modelling (and not software modelling) by use of object oriented (OO) concepts
3 It provides extending and specialised mechanisms for extension of concept cores
4 It is independent on actual programming languages and development processes
5 It provides formal base for understanding of modelling languages
6 It supports grow of OO instruments
7 It supports the highest level of concept development, e.g. collaboration, frames, models, components
8 It integrates the best practices
9 It generates modelling language acceptable for human and machines

UML offers 9 types of diagrams: Use case, Class, Object, Sequence, Collaboration, Statechart, Activity, Component and Deployment.

Use case diagram

It shows system from the viewpoint of external users. It is composed of Actor - it is a user using SW interface for function processing. A person may represent more Actors and vice versa, more persons may represent one Actor. The actor may be not only user of the system but also the system, which activates some case of use (e.g. real time). Next entity is Use Case - it may be defined as a unit of system functioning from the user point of view. Between Actor and Use Case an association is formed. There exist dependence relationships (extends, uses or others) between Use Cases and inheritance relationships.

Class diagram

Class diagram is basic structural, analytic and proposing object model. The diagram records classes as basic units and it also records inheritance relationships, realisation relationship and interfaces. The attributes and operation of classes have been defined..

Object diagram

Object is a class instance, e.g. defined object with specific values of properties and behaviour. In UML language the object icon is depicted by rectangle similarly to class icon, the only difference is that the object name is underlined. The name of particular instance is on the left of colon, the class name is on the right of colon.

State diagram

It records behaviour of an object in time. The state consists of attribute values in the given time and the behaviour is defined by change of state based on external event. This diagram would have all possible states of objects. Further diagrams can better record the behaviour of the whole system. State diagrams are useful for displaying control mechanisms and user interfaces.

Interaction diagram

This part of model defines behaviour of the system or its parts. Objects and their mutual interactions occur in diagrams. One of two displays is represented by the **sequence diagram**. On the time axis it records sequence in which objects communicate to each other by sending messages (by activation of methods). The second way of interaction modelling is the **collaboration diagram**. Messages among objects are here recorded in lists comprising evaluation of oriented edges linking collaborating objects. The first diagram shows better the time relations, the second one shows better the mutual linkage of objects.

Activity diagram

Activity diagram in UML language resembles flow charts (besides sequences, branching and cycles it also enables to model parallel activities and their synchronisation). It records steps, decisions and sequence branches.

Component diagram

The component diagram shows components, interfaces and relationships. The component is implementation building element used for application assembling. It means it may be a module (on source level, relative module) but it also may be an entire library or application, which are parts of the system. The approach to the component may be realised by its interface, e.g. by a set of operations. The relationship between the component and its interface is called realisation. A component may use services of another one.

Deployment diagram

Deployment diagram in UML language shows the physical architecture of system. It is possible to display computers and devices, to depict their mutual connection and also to depict software installed

at a device. Each computer is drawn as a cube and mutual linkages to other computers are lines connecting the cubes.

4 USE OF MODELS BY DEVELOPERS AND USERS

During previous period structured methodology was used for development of PIS. Process models of subsystems "Applicant" and "Student" were generated by Power Designer tool (Fig. 1).

Preliminary database models were generated by data modeller of the CASE tool (Fig. 2). Important advantage of Power Designer is generation of source code of modelled database with possibility to work with various types of databases. In our case it represented undoubted advantage because we have worked with database server INGRES 2.0. It also enables to define integrity conditions, constrains and it contains patterns for trigger writing. By correct set up it is able to generate a code. It may implement the code directly into database by ODBC connection. In practise it means that the database developer does not need directly to contact source code and in spite of this he can generate database in full scale. Advantage is also the possibility of Reverse Engineering realisation by means of which it is possible to sustain more simply the consistency between model and database. During development the structured modelling appeared as improper because it did not meet requirements of complexity and description universality. It also represented a strong difference between new object-oriented environments for software generation and model. A possible example may be the realisation of subsystem "Applicant" for study departments of individual faculties in Builder 5.0, which is based on object approach. For basic documentation and acquaintance with the system may be used for example DFD applicant model, however, object-oriented approach already results from own character of environment.

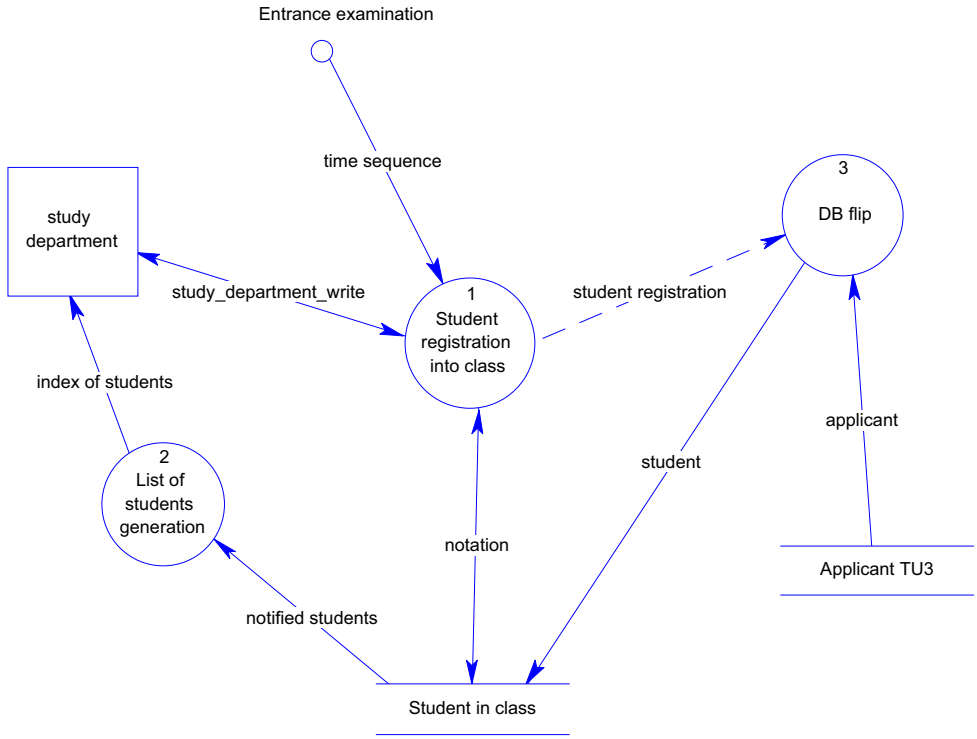

Fig.1: Part of process model of PIS system

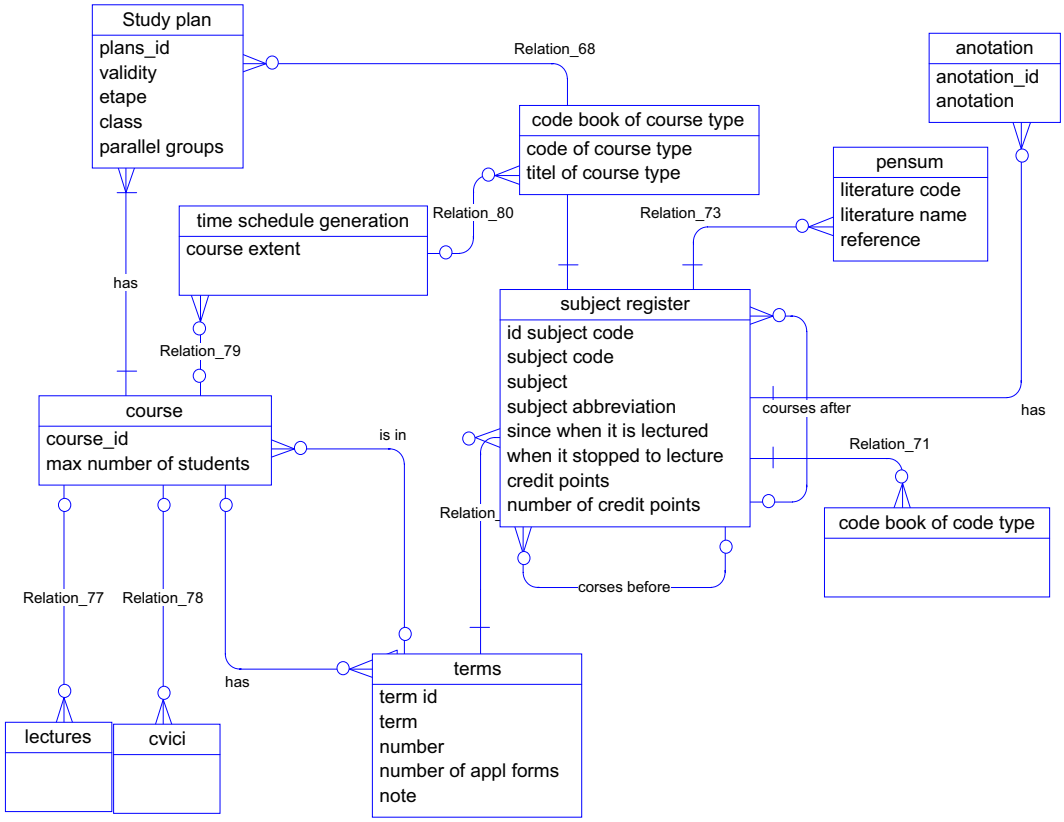

Fig. 2: Part of data conceptual model

With respect to development in modelling technique and transition to a new PIS version at the beginning of this year we approached object-related modelling. The base is composed of UML while individual models are generated and sustained by CASE tool Rational Rose 2000 Enterprise Suite. The majority of the whole model consists of USE CASE (UC) models used for collection of requirements and documentation of individual system parts behaviour. A basic division of USE CASE models and basically division of the whole system is depicted by Package diagram in Fig. 3.

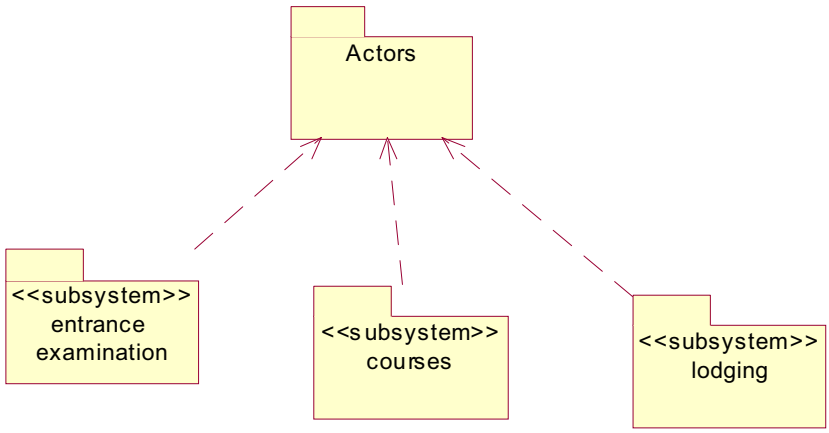

Fig. 3: Package diagram of new PIS version

Currently the subsystem "Entrance examination" (also called "Applicant") is documented in detail because it comprises the base of system and natural data input. As an example we have introduced a part of the main USE CASE diagram, subsystem "Entrance examination" in Fig. 4. The example depicts functions provided to the participants: "Study administrator", "Applicant" and "Administrator".

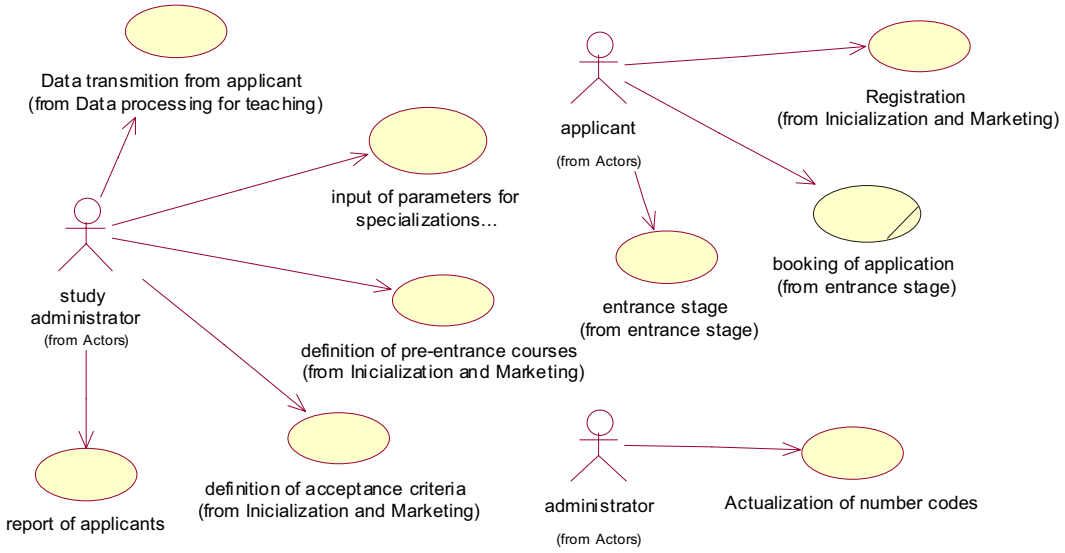

Fig. 4: A part of USE CASE diagram "Entrance examination".

The subsystem contains more than 30 similar diagrams. It also contains description of functions related to system only partially or functions, which are not directly run in the information system, however, their effect on the system function is considerable. In question is theUSE CASE marked by the stereotype "Business". As an example may serve UC "Submission of application form" (Fig. 4).

Although the UC diagrams characterise the system requirements, they do not say anything about the system internal structure. Class diagrams were used to describe internal structure. In the first approximation the basic structure was proposed which gradually evolved into the final stage by study of requirements. Similarly to requirement collections there exists a detail class diagram for subsystem "Entrance examination" . A part of this simplified diagram is depicted in Fig. 5.

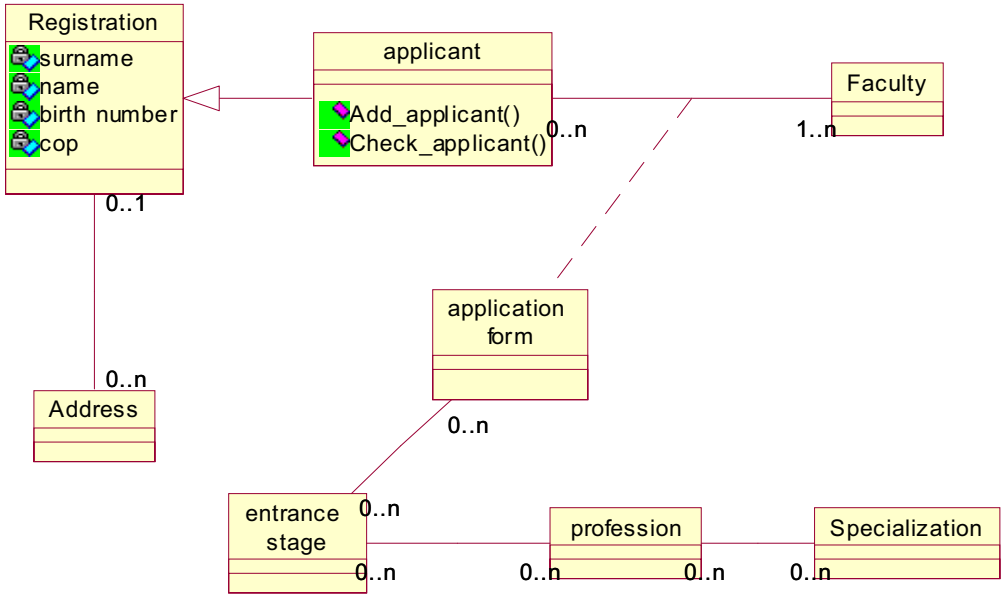

Fig. 5: A part of class diagram of the subsystem "Entrance examination". Note that the diagram is simplified and not all the relations, attributes and methods are depicted.

The scope of subsystem "Entrance examination" was to test the ability of UML language during modelling of system like PIS. The subsystem is one of the most important module enabling data input into the system as well as it is also a logical start of data processing in the entire information system.

Advantage of the developed class diagram is also characterised by a fact, that besides information on internal structure of application, it comprises the logic base for proposal concerning the structure of database data model on which the application is based. One of the module from Rational Rose product is also Data Modeler enabling generation of data models in the so called schemes. The truth is that this module has not been used for database proposal so far because it does not provide so much practical advantage like CASE tools assigned directly for data modelling.

5 CONCLUSION

We had a possibility to understand modelling importance during operation and maintenance of information system "Student" where many problems are related to absence of complex system model. Many errors could surely be found and removed earlier provided that the system was properly documented from the very beginning. During layout and development of new pedagogic information system we have tried to use modelling tools as much as possible. At the beginning of 2001 we approached object-related modelling. It seems that the application of UML - standard modelling language for visualisation, specification, construction and documentation of software systems is most appropriated. The testing of UML language by modelling of subsystem "Entrance examination" proved its ability for further work in the system. We paid a considerable attention to USE CASE models which systematically and intuitively record function requirements on the system. This enables a management of entire process of PIS development. The individual models are processed by CASE tool Rational Rose 2000 Enterprise Suite. The use of other UML diagrams, e.g. activity diagram and interaction diagram is also possible. The modelling of other subsystems of PIS will be processed by analogous procedure.

We assume that the development of a complex PIS model may be used as a reference data for the development of an analogous system by other universities in Slovakia.

REFERENCES

[1] HAVLICE, Z. and H. TELEPOVSKÁ, J. GENČI (1997). Pedagogic information system. Proceedings of Slovak Conference – University Information Systems - UNINFOS 97. Technical University in Košice, Slovakia (in Slovak)

[2] HAVLICE, Z. and V. CHLADNÝ (1999). Object Oriented Analysis and Design of Information Systems. Proceedings of Scientific Conference with International Participation – Computer Engineering & Informatics. Košice - Herľany, pp.102-107

[3] HAVLICE Z. (1998). Modelling and prototyping in project management and development of information systems. Publishing Elfa, Košice, Slovakia (in Slovak)

[4] KASANIC V. (2001). Hypertext presentation of UML and its application in praxis. Student Research Report, Department of Computers and Informatics, Technical University in Košice, Slovakia (in Slovak)

[5] KULAK D. AND E. GUINEY (2000). Use Cases: Requirements in Context. ACM Press, New York

[6] JÍLKOVÁ H. AND I. STANOVSKÁ (1992). Structured processes of analysis and information systems proposal. ESPRIT Ostrava, Czech republic (in Czech)

[7] SCHMULLER J. (2001). Thinking in UML language. First Edition, Grada Praha, Czech Republic (in Czech)