**Stojan Russev**
**Jaroslava Kniežová**
**Roman Russev**
University of Economics
Faculty of Economic Informatics
Bratislava, Slovakia

# Advanced Customer Oriented Development of Software (ACES)

**Summary**
Object oriented development languages and event driven programming, distributed or centralized data processing with thick or rich clients are used at present in information systems (IS) development. Using of existing methodologies and methods in IS designing does not always mean reaching of needed project solution quality. In some cases it is more effective to use combination of structured and object oriented tools or new methodology, which is built according the newest information technologies. One of such methodologies is Advanced Customer Oriented Development of Software (ACES), which is oriented on IS of economic organization development and application of the newest information technologies. ACES was established by the authors at Faculty of Economic Informatics of University of Economics in Bratislava.

**Key words**
ACES Methodology, object software engineering

## Methodology Main Characteristic

ACES Methodology aims to solve problem areas of IS designing, which are built using of very modern information technologies and its main characteristic is:

- project team member is lead through all phases of IS development up to the effective and high quality design,
- less time taking design solution with clearly, useful and effective modelling,
- it based on object oriented approach and standard conventions of UML language are reflected,
- platform universality is reached, i.e. contains unification for modelling abilities not only for one user applications and environment of distributed IS, but for the Internet environment and event driven programming as well,
- methodology life cycle not only helps to organize and optimize work of development team members but also reflects user needs and requirements in that way, so that solution areas were understandable for common users as well,
- new life cycle elements are clearly defined, modification frame for existing elements from the UML language and elements used without modification are described
- present trends of automated IS development are reflected and it is lead to model optimization in view of maximization of automated source code generation possibilities.

ACES methodology provides solution for these problem areas:

- inconsistency of object oriented design and relational databases
- not clearly designing of IS structure,
- absents of methodical based design of presentation tier and user interface,
- problems in internet application design,
- needs of validate cycle implementation, which includes design solution results evaluation based on pre-implementation application prototypes

## Life Cycle of ACES Methodology

ACES life cycle parts IS development into progress of phases, stages and steps, which are needed to be done for successfully designing, implementing and modifying of IS. Life cycle of ACES methodology does not cover information strategy definition, because this is task for the management of economic object, not for the IS development team. In ACES IS creation begins after defining complete solution of needed information in company and working out the task giving for IS development.

ACES life cycle consists of following phases and stages:

1. ***Analysis and Conceptualization Phase***, which includes following stages:

   a. Analysis of Economic Object
   b. Analysis of Existing IS
   c. Basic System Requirements Definition
   d. Definition of Variants of Possible Solutions on Conceptual Level

e. Best Variant Choosing and Analysis of Solution Feasibility
f. Organization Strategy Definition and Plan for Design Phase
g. Definition of Complete Finance Calculation and Brief Time Project Plan
h. Documentation of Analysis and Conceptualization Phase

2. ***Design Phase***, which includes:
   a. Design of Common IS Part
   b. Design of IS Use Cases (Services) and Components
   c. Documentation of Design Phase
   d. Defining of Realization Plan

3. ***Project Realization Phase***, which includes following stages:
   a. Creation of Physical Model of Database
   b. Use Cases (Services) and Components Coding
   c. Use Cases (Services) and Components Testing
   d. Integrating and Testing
   e. Design of Organization Work Processes for IS Implementation
   f. Help and Realize Documentation Creation

4. ***Phase of IS Implementation***, which consists of following steps:
   a. Hardware Environment Preparation and Setting
   b. System Software Preparation and Setting
   c. Fulfilling the Database With Real Data
   d. Beta Version Deployment

5. ***Operation Phase***

In IS development according to mentioned Life Cycle it is supposed to realize validate cycle, which is executed in three life cycle phases and consists of following phases and stages:

1. ***Phase of*** Validation ***Process Preparation,*** which is realized after Analysis and Conceptualization Phase and consists of:
   a. Evaluation Commission Definition
   b. Design of Evaluation Processes
   c. Design of Organization Processes of Changes Executions
   d. Work Results Documenting
2. ***Results Evaluation Phase***, which is realized in life cycle phases: *Design, Project Realization* and *IS Implementation*, it consists of following stages:
   a. Executing of Monitoring Activities
   b. Considering of Requirements Fulfil
   c. Description of Optimization Possibilities and Next Proceeding
3. ***Changes Execution Phase,*** consists of following stages:
   a. Collecting and Analyzing of Change Requirements
   b. Solutions Designing
   c. Realization of Changes Requirements
   d. Documentation Corrections

ACES methodology brings significant changes in comparison to nowadays used methodologies, especially in phases *Analysis and Conceptualization* and *Design*, and therefore the attention will be pointed at these phases of ACES life cycle in next writing.

**1. *Analysis and Conceptualization*** Phase

In this phase the needed activities are divided into stages, in which the exactly defined steps and models are realized.

### Analysis of Economic Object Stage

By the analysis of economic object the level of which the future IS will reflect the company specifics, cover its main activities, respect customs and abilities of its employees is influenced. Results of this stage are important material for understanding of company working by the development team, for finding out the basic of solved tasks and information needs on different management levels.

In *Analysis of Economic Object* Stage the following steps have to be realized:

- **modelling of company organization structure**
  UML and object oriented methodologies does not support modelling of organization structures. In the structured approach the organization diagrams are used. These diagrams are recommended to use in ACES. In case, that IS being built is in smaller company with one level management or is going to cover only small part of organization structure, it is possible to skip creating of organization diagrams.
- **modelling of activities and working tasks**
  Tasks of IS being built are dependent on realized activities and problems, which are needed to be solved. Therefore modelling of realized processes and activities is important

base for conception designing and for following project solution of IS as well. On correct understanding of process IS inputs, data processing and outputs depend and how it would be rationalised as well.
Modelling of activities and solved tasks is realized in structure approach using the hierarchical function diagrams. These diagrams are not included in UML. They are suitable for modelling of activity decomposition but they do not show needed progress of process steps in realization. On the other side, UML offers very suitable for this purpose sequence diagrams, state diagrams or activity diagrams. Using these diagrams is recommended in modelling mentioned information.

- **modelling of work responsibility**
  Work responsibility represents the last defined information type, which must be visualized in Analysis of economic object. These have important role for lately defining of IS users, roles and privileges in working with IS. It is needed to define which organization unit is responsible for which activity. Here Use Case diagram from UML is possible to use or the Relational Matrix from structural approach as well. Better view provides Relational Matrix.

### Analysis of Existing IS Stage

Analysis of existing IS should give information about how much of existing automation is, about information environment, which are employees used to, about the needed inputs and outputs, data processing and so on. The database models of existing database are especially important. From existing database the data will be transferred to the new one. The modules, components and classes of the old IS may be used in the new IS as well.

Models from this stage are not used directly in creation of the new IS and therefore in ACES modelling techniques and tools may be used according to the chosen approach to IS development in previous automation and only supplement the existing documentation. Modelling of automated processing should join with before created models for the existing IS. In case the existing IS is well documented, it is even not needed to realize this stage.

### Definition of Basic System Requirements Stage

Basic system requirements represent the requirements on software and hardware environment, work mode, inputs and outputs and needed data processing. The defining of it is based on given IS task and analysis results. As usually IS task is not made by the informatics, the development team should the task specify and make up already defined requirements so as it would be possible clearly specify all next needed works.

Requirements modelling is in relation with the new IS and therefore the UML notation should be used. Requirements modelled for example by the Use Case diagrams present suitable solution. The right visualizing brings project starting in high quality and in view on iteration possibility completeness control as well. It is good to add short word comments to the diagrams, which enables ease understanding for the customer.

The IS requirements correspond with the information strategy, i.e. with the aims in information area. There together with the requirements it is important to define IS aim in this stage.

### Definition of Possible Solutions Variants on Conceptual Level Stage

Conception of IS represent important milestone in IS project. It has to contain way of IS solution, i.e. basic characteristic of IS being built as: architecture, database solution, user groups, most important inputs, outputs and so on.

According to ACES the conception is created based on analysis in shortened way (contains requirements, environment description, basic structure, cost estimates, advantages and shortages of particular solution variants) in more variants and the solution provider recommends one of them. Variants should contain different technologic solutions with supposed costs and time of realization.

### Stage of Suitable variant choosing and preliminary analysis of solution feasibility.

After mutually agreement with the customer the chosen variant is worked out and completed with the missing models. These models are the basis for the Design Phase and are used for preliminary analysis of solution feasibility and definition of validation cycle as well.

The important step in this stage is to create prototypes of input-output screens. Based on them user can definitely decide about the functionality and developed user interface.

Another step is to design brief IS model. IS structure design in object oriented development is one of problem area. Suitable way of structure modelling is defined in *Design* Phase.

Another model of conception is brief database design. ACES suppose IS working with relational database, therefore the tools of structured ap-

proach should be used, i.e. using Entity Relationship Diagram (ERD) to creation of conceptual database model.

In ACES it is not thought about using Class diagrams in IS conception.

## 2. *Design* Phase

Project works in design phase come out from started solution in IS conception.

IS solution in ACES is iterative process, what means, that team member returns to the previous defined parts so as to specify them in more details or improvement according to the need. Returning can be realized from other phases as well, for example from phase Project Realization.

### Common IS Part Design Stage

The common IS part design contains design of parts, which are used by more Use Cases and IS components. Therefore the common part design has to be realized before designing other IS parts and provide compatibility between them. The common part design contains design in all three tiers of IS. In application tier it is needed to design IS structure, administration and user rights to the use cases and components, or other IS parts. In presentation tier the common mask and screen design has to be defined and user rights to screens as well. In data tier logical database model has to be designed, code lists and code systems are needed to define.

### IS structure design

IS structure design belongs to the problem areas of object oriented IS development. In object oriented design and UML using, IS structure visualising should by done by Package diagrams, Use Case Diagrams, Component diagrams, Deployment Diagrams and Class diagrams. In packages components can be drawn, use cases and classes, also in components packages, use cases and classes can be drawn, in nodes packages, components, use cases and classes can be drawn. Building elements in other elements leads in large IS to creating not clear and not complete diagrams. On the other side drawing IS structure into more kinds of diagrams causes looseness of projects uniformity and leads after that to mistakes in realisation.

So as to eliminate these shortages new way of visualising of IS structure was in ACES accepted as well as following rules:

- IS structure should be defined in hierarchical diagram, because hierarchical diagram gives more clearness and comprehensibility of solution
- UML notation elements should have firm hierarchy: packages, use cases, components, classes
- nodes in deployment diagrams represent only deployment of particular elements of IS structure and not IS structure
- IS structure diagram does not need to contain all kinds of elements for example use cases may be decomposed into use cases on lower level

After accepting these rules, the IS structure can be visualised using ***IS Structure Diagram*** (Picture 1) not only in Conception but in IS Design as well.

*IS Structure Diagram* modelling has the following advantages:

- IS decomposition is clear and comprehensive, what is especially important in development of large IS,
- classes of application tier are clearly assigned to the use cases, which simplifies project management
- it is possible to read from the model also usages of one class by more than one use case or components (in example on the picture 1 is the class n used by component n as well)
- use case can be marked according to the way of their realization (for example TC - thick client, RC - rich client), what makes next IS designing more uniform
- elements hierarchy is clearly defined and when modelling more levels with the same element (for example with use cases) it is clear, that the element form higher level contains underling elements

The suggested solution of IS structure modelling in ACES is not supplied by UML models, but it complements them in suitable way.

### Design of IS Administration and Management

IS administration has to contain mainly following items:

- design of code lists and code systems
- code lists updating
- security in which is included:
  - user evidence and granting roles
  - evidence of realized database updating, logging
  - database backup
- database reorganization if it is required by the design

Code lists and code systems design is important step for reaching of whole project uniformity. The same code lists are used by more use cases and components and therefore it is needed to design

already at the beginning their structure, code system, way of updating and define persons responsible for this task as well.

IS security is at present provided by the database systems, but in spite of that its design should be not missed in project. The access rights should be defined after IS structure, presentation tier and logical database model design. In design there should be the user groups and their roles defined, i.e. access rights to components or use cases (application tier), screens (presentation tier) and database tables (data tier).

The suggested user groups should by created for organization units or management levels, which are presented in Use Case diagrams by concrete actors. So defining of user groups can by provided according to Use Case diagrams, but also according to Organization diagram.

Other way to define user rights is using Relations matrix (RM). This way is especially effective in case of existence of more organization units and users (user groups). It is needed, that this matrix is created according to the organization diagram and use case diagrams (users). In the matrix the users should correspond with the users from use case and the organization units should correspond with the units from organization diagram.

### Use Case and IS Components Design Stage

The design solutions in this step must correspond with the defined IS common part and may be realized by more teams. Design of presentation and application tier in this stage are very important steps, on which ACES takes big attention.

### Presentation Tier design

In object-oriented methodologies the presentation tier design is not enough covered, which involves notable problems. In ACES methodology the presentation tier design is based on screens prototypes, created in IS conception and is divided into these steps:

- design of menu structure
- design of screen flow
- detailed screen design

### Design of Menu Structure

Menu is hierarchically structured and therefore hierarchical diagrams are suitable for it's modelling. It is possible to use screenshot with down rolled menu in project. This solution is interesting mainly for the customer, who is not forced to imagine the future menu. In case the menu is quite big and often modified according to the logged user, this way of visualising can easy grow into lots of pages in project.

To simple visualising the menu in tabular way hierarchical diagrams provide (picture 2), which can even be oriented in the same direction, as the menu will.

The diagram showing the menu, called as ***Menu Structured Diagram***, has to enable to define the hierarchical relations of lower-level menu items in case the lower level is to be placed separately. This is possible to do when drawing a small triangle next to the upper level menu item, which represents the mark shown next to such item in real menu as well. In this diagram it is possible to draw also the dividing menu items into groups by the dividing line, which is very similar to dividing menu items in real.

### Design of screen control

Modelling of screen control is more difficult at present. Large IS contains lots of screens, the user access to them is used to be realized in very intuitive way and in no case it is hierarchical. Therefore the hierarchical diagrams cannot be used. Trying to draw all screen of one of such IS in one diagram would not be good solution because of the great amount of screens and also because) of the fact that never all screens will have any relation between each other.

Design of presentation tier in ACES is related to particular components and use cases and not to the whole IS. This fact is very important in screen control modelling, because the screen belonging to use case is used here.

The solution is given by the Activity diagram, which is made for concrete IS use case which the user will use. These diagrams are in ACES called ***Screen Control Diagrams***.

In up to this time used Control Flow Diagrams is only way of defining of screen calling. In Activity Diagrams also the conditions of screen calling can be described.

Some screens contents more tabs, which are used to work with some data groups. These tabs are not shown in activity diagram. In this case one group of information would stay not visualised, therefore it has to be completed in Screen Control Diagram. Such diagram, which contains appropriate solution, is shown on picture 3.

### Detailed screen design

Screen design is one part of IS project which should give enough information inside the team as well as outside it, i.e. the customer (user) should capture complete idea about screens design and functionality.

On the other side it is important to include the solution of particular screens, which has been defined by the project members inside the team. This solution should be presented in project in that way so that no important information was lost and the programmers could this realize.

Information which has to be covered in IS project in detailed screen design part can be divided into:

a. screen design - appearance
b. screen data (input and / or output information)
c. screen functions:
   - user's functions (these functions serve the user directly, for example searching)
   - hidden functions (user does not directly call them, but they are needed for execution of the user functions, for example controlling of the user grants for enabling or disabling some of the screen controls)

The first information group presents the solution of the screen appearance. It is important to think about the fact that IS built at present usually have highly dynamic screens according to the user activity. Therefore the screen appearance should be reviewed together with the possibilities of its changing in reaction on user's generated events.

The second information group, which has to be worked in the IS project, are the data, which the screen brings. This means input information, which are filled in by the user on this screen, or output information, which this screen displays to the user. At this moment the origin of the output data is needed to define and the destination of the input data as well. The exact place in the database must be marked as the named attribute of the table.

The last information group presents the functions, which the screen provides. These can be divided into two groups: in the first type belong the functions as the core of IS functionality for the user or functions used for the user comfort and they are often activated by the buttons. The second group contains functions on the background from the user view and the user often does not know about them. Most often these functions are various controls, for example user access or controlling of input data.

### Screen Design

The appearance of the screen must be considered from the two views. The first represents the static view on the screen – what colour, logo, controls placement, etc. the screen has. The second point of view means screen behaviour that is - how the screen appearance is changed according to its interaction with the user.

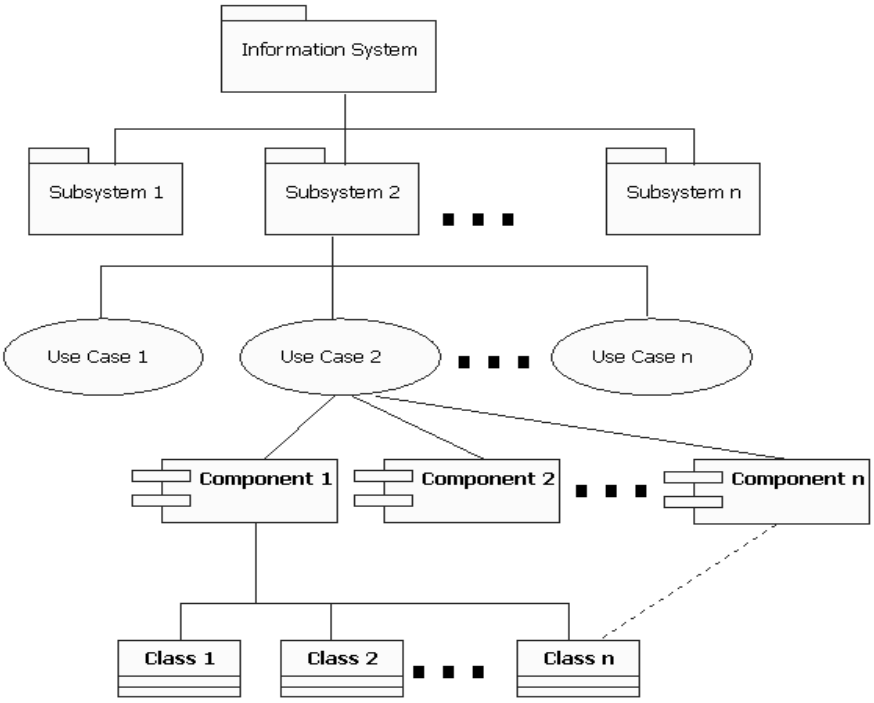
In ACES it is recommended to define the IS screens from the static point of view- static design and the dynamic point of view as the screen dynamics. For exactly defining the screen static, simple picture of the screen (screenshot) should be included in the IS project. The short verbal description is suitable to add as well.

IS developed in nowadays are specific in that way, that their screens have high dynamics in details. For the defining the screen dynamics the screenshot does not cover the needs. Possible solution seems modelling the screen, which is divided into more parts (frames). One part is changing according to the item, which is chosen by the user in the second part. The example of this case is shown on picture 4. Here the screen (form) named *Exams in 7th Term* is modelled. This screen consists of two parts (frames): *Exam List* and *Exam Details*. The *Exam Details* part changes with influence to which item (exam) from the *Exam List* part is chosen by the user.

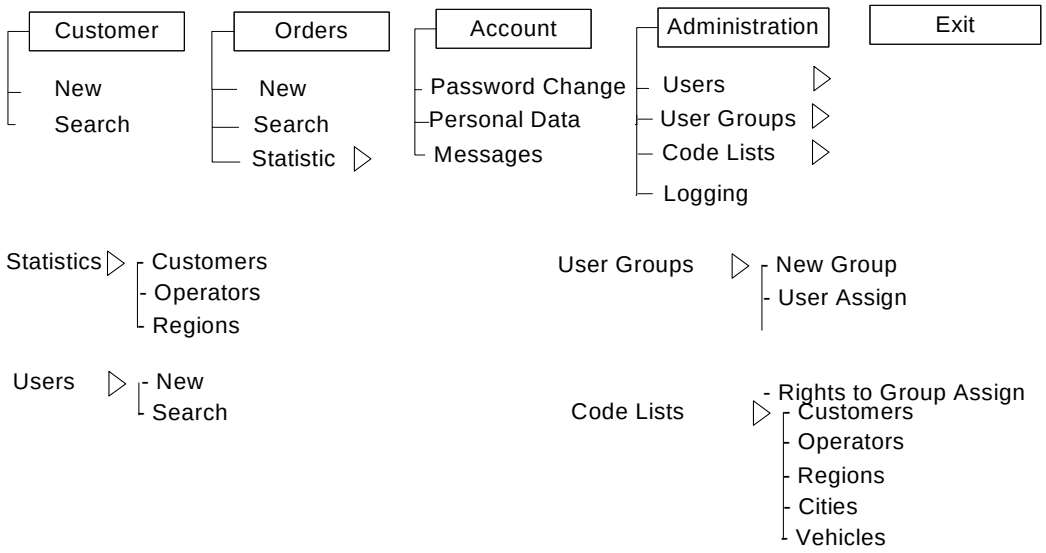
According to the mentioned above it is possible to define the core of the screen dynamics and to model it using defined relation: aim - aim subject. If the screens in IS project will be completed with such diagrams, which clearly show active screen parts and their relations to each other, this models are able to bring much needed information about the screen dynamics. Based on these models the programmers are able to clearly realize functionality providing the dynamics and the customers (users) can obtain good conception about the future screen appearance. There is only small shortage in understanding models for the user - the need to explain the relation *aim -aim subject* and the graphical representation of this relation as well. These diagrams are in ACES named ***Screen Dynamics Diagrams***.

### Screen Data

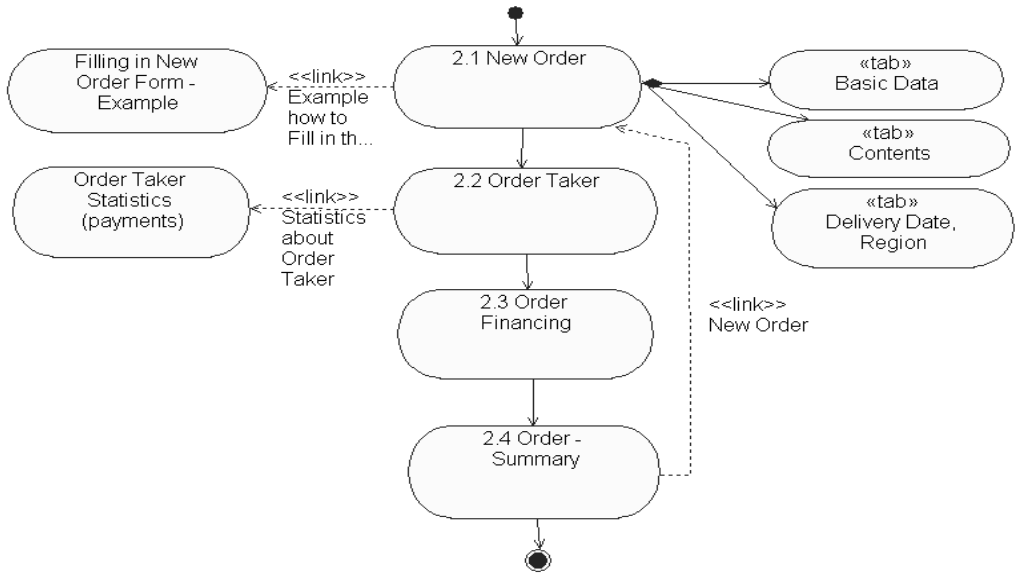
Each screen is used mainly so as to bring the information to the user. This information can be brought to the system or out from it. All controls on the screen are of the two groups. First is the group of controls not connected with database (most often buttons) and group of controls connected to it (not needed in directly way). In detailed screen design it is important to exactly specify where is the destination (or source) place in database for all connected controls. This mean specifying the attribute in table, from which it is to be read or to which it is to be written.



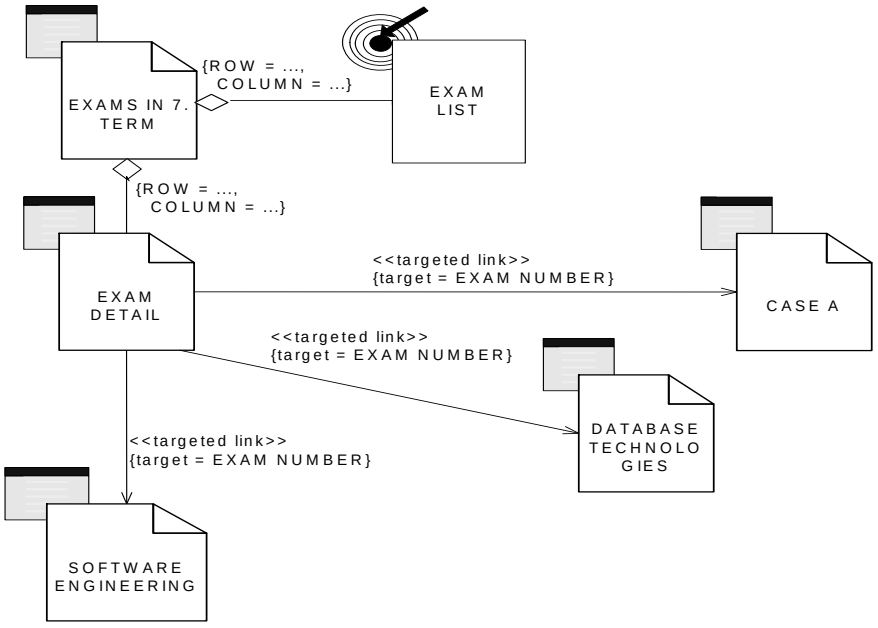

Picture 1 IS Structure Diagram in ACES



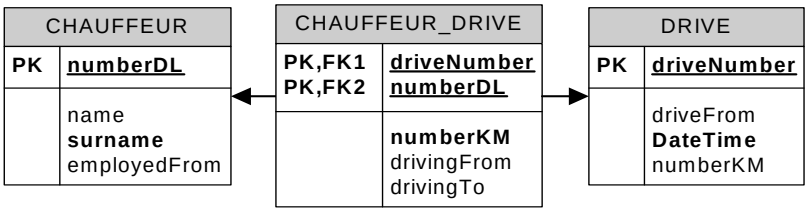

Picture 2 Example of Menu structure Diagram



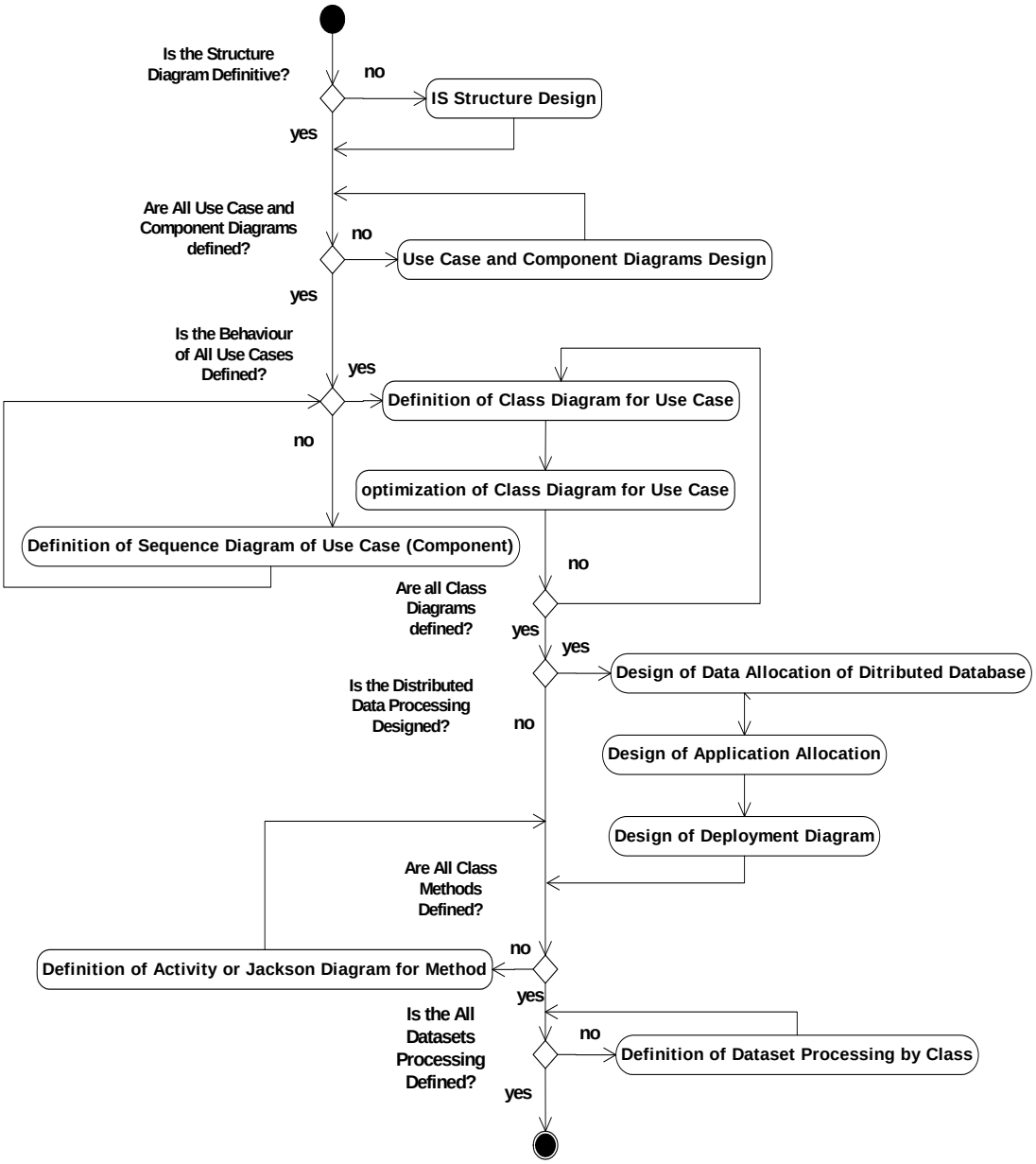

Picture 3 Example of Screen Control Diagram




Picture 4 The dynamics of the Form (screen ) Exams in 7 th Term




Picture5 Example of database model excision




Picture 7 Design of application tier in ACES

The structured approach enables to model database screen connection using table, in which the data items of screen and database attributes corresponds together. This is very tabular and reliable way to define all items (controls) without the possibility to skip one of them. In object-oriented approach the class diagram is created, for modelling the presentation tier of IS. But these diagrams usually do not contents the objects used only for transferring database data.

In ACES methodology it is recommended to use selection of needed database tables for modelling the database screen connectivity. These tables are shown as the undercut of the database logical model. This enables the programmer to work with the screen design directly instead of searching in the logical database model for one table again and again. The database model undercut gives very tabular way for working with screen data. Example of database model undercut is shown on the picture 5 and the needed SQL commands for getting the screen data are shown on the picture 6.

```
SELECT name, surname, numberDL
FROM chauffeur, drive_chauffeur
WHERE chauffeur.numberDL =
drive_chauffeur.numberDL

INSERT INTO drive_chauffeur VALUES
(    ,    )  //numberDL, numberKM
```

Picture 6 SQL command for dataset

The database model undercut with the SQL query represent clearly way of information modelling which gives much needed information and reliability of once modelled information as well. The way of modelling of the screen data and database connectivity in ACES is suitable also for modelling of components or use cases and database connectivity of application tier as well.

### Screen Function

Functions of screen and its dynamics are in relation together because functions provide the dynamics. In dynamics modelling there was important to model screen appearance according to the interaction with the user. In screen function modelling there is important to model all functions which may but do not need to involve the screen changing.

In object-oriented approach the screen represents the object, which has (as the other objects have) its methods, which are in case of screens mentioned functions. For modelling objects and their methods are in object-oriented approach class diagrams used. Because of the needs to show situations when the functions are activated, the events must be modelled. Also the logic order of the function must be modelled. The functions, events and logic have relations to each other and this should be shown in project. The object-oriented approach does not give tools for modelling function logic in screen design, what can be considering as shortage.

Information, needed in screen design can be characterized as events, to them belonging functions and the additional conditions in relations to each other. This means to assign every event to right function, so as the programmer could realize these function as driven to correct events. When the function starts quite often it is used to check the additional conditions assigned to this function.

The solution for modelling these assigning is to use table, which could contain all functions of the screen, events and condition as the table columns. Jackson Structure Diagrams or Activity diagrams are very suitable for modelling the function logic. The diagrams should complete the table composed for every modelled screen.

When defining screen function in IS project, it is needed to show in tabular way the all functions in connection with the events and conditions. The table with this kind of information represents suitable way to clearly define the relation: function - events - conditions (example is shown in tab. 1).

| Number of event | Event | Additional conditions | Action |
|---|---|---|---|
| 2.1 | ADD CHAUFFEUR | **if** the user has the right to add the chauffeurs to the drive | adds raw into the *grid* |
| 2.2 | CHAUFFEUR ASSIGN | **if** the chauffeur was changed in detail **then** „do you want to change ...? “ | loads changes in *grid* |
| 2.3 | REJECT CHANGES | **if** the chauffeur was changed in detail **then** „do you want to reject ...?“ | rejects changes loaded since form opened |
| 2.4 | MARKING CHAUFFEUR IN RAW | | DELETE CHAUFFEUR and DETAIL CHAUFFEUR buttons enabling |
| 2.5 | DELETE CHAUFFEUR | **if** the user has the right to delete chauffeur in drive | deleting the chauffeur from *grid* |
| 2.6 | CHAUFFEUR DETAIL | **if** the user has the rights to view chauffeurs in drive | opening form named frmDetailSof |
| 2.7 | PRINT | **if** the printer is connected | printing the card of chauffeurs in drive |
| 2.8 | OK | **if** any changes were **and** ASSIGN CHAUFFEURS wasn't called **then** „do you want to change? ...“ **if** the data of allowed edit is not over | database updating |
| 2.9 | CANCEL | | closing form without changes |

Tab. 1 Screen Functions

In ACES it is recommended to use table for screen functions definition. Using table is easy to understand for the customer (future user). Modelling the function logic using the Jacksons Structure Dia-

grams or Activity Diagrams should be used for the more complex functions, with logic of more than two - three steps.

### Design of Application Tier

Application tier contains all system processing, which are needed for working out the information tasks of the system. The basic elements are classes included in use cases and components. All these classes should be modelled in IS Structure Diagram with their hierarchical relations. Because of the needs of the next phases of IS development their detailed description of their continuity and behaviour should be completed.

The detailed design of application tier may be according to ACES realized for the subsystems, use cases or components, depending on what it is in concrete case most suitable. Only in case of small IS (with small number of use cases) it can be realized for whole IS. This way the project is getting more tabular and the number of non-effective designs is getting smaller.

In the detailed design of application tier it is important to model:

- classes realizing defined use case and messages they are sending to each other
- class and object inheritance
- method algorithms
- classes deployment
- interoperability of the application tier classes with the data tier

The defined process of application tier modelling in ACES is:

- use case diagrams or component diagrams creation
- creation of sequence diagrams for each use case on the lowest level, or for each component with the description of the main classes and messages functions, which they send, so that it could be possible to get clearly idea about the way of working out the information tasks and participation of the particular classes
- creation of the class diagram for the use case on highest level (or component) together with the relations to the other use case classes and the all classes have to be contained in the class diagram
- optimization of class diagram according to enabling the most possible inheritance
- modelling of component and classes deployment into the nodes using deployment diagram (this step must precede design of the allocation of distributed database)
- modelling of methods algorithms of particular classes using State, Activity or Jackson diagrams. It is not needed to model small intuitive functions or object methods, which consist of small number of command lines and short verbal definition is enough to understand them.
- modelling of application tier classes work with datasets and database as it was explained in design of presentation tier

This process is shown on picture 7. There are contained only those diagrams, which should be mandatory, created in case of large IS. According to the needs the user can create other diagrams as well.

Another requirement is, that the model decomposition levels were defined so as to give all needed information for the realization phase of the project.

As it can be seen in application tier modelling the UML diagrams are used in ACES but in exactly defined continuity. This way number of problems can be solved, which appear in using other object-oriented methodologies.

Data tier definition is concluded in common IS part design, presentation tier design and application tier design. Therefore in ACES methodology it is not marked one step to designing the data tier.

## References

1. Conallen, J. (1999), Modelling Web Application Architectures with UML, Rational Software Corporation, CA.
2. Gornik, D. (2002), UML and Data Modelling Profile, Rational Software Corporation, CA.
3. OMG (2005), Unified Modelling Language: Superstructure, version 2.0, formal/05-07-04, August 2005.
4. OMG (2006), Unified Modelling Language: Infrastructure, version 2.0, formal/05-07-05, March 2006.

## Prof. Ing. Stojan Russev, PhD.

Prof. Ing. Stojan Russev, PhD. is dean of the Faculty of Economic Informatics. He works in areas of software engineering, methodological aspects of information systems development, automation of programming and projecting. He is author of more than 100 studies about the problematic. He is teacher at the Faculty of Economic Informatics and is active in subjects as Software Engineering, Database Technologies and others.

## Ing. Jaroslava Kniežová, PhD.

Ing. Jaroslava Kniežová, PhD. is teacher of Department of Applied Informatics at the Faculty of Economic Informatics. The subject touched by her are: Software Engineering, Database Systems, Database Technologies. She has written more than 20 studies.

## Ing. Roman Russev

Ing. Roman Russev is teacher of Department of Applied Informatics at the Faculty of Economic Informatics and doctoral student at the same faculty. The subject teached by him are: Automated Programming and Projecting B, Case Studies A, Case Studies B. He has became the ORACLE Certified Developer (Oracle Designer, Oracle Developer). He has written 22 studies.