ACADEMY OF SCIENCE
OF THE
CZECH REPUBLIC

INSTITUTE OF INFORMATION
THEORY AND AUTOMATION

Jindřich Bůcha, Miroslav Kárný, Petr Nedoma, Josef Bohm, Jiří Rojíček

DESIGNER 2000

FEASIBILITY STUDY

VERSION 1.0

NO. 1918 SEPTEMBER 1998

Contents

ABBREVIATIONS **4**

1 INTRODUCTION **5**

2 CURRENT STATE **7**

2.1 Basic terms **7**

2.2 Overall description and evaluation **8**

2.3 Detailed data flow **11**
2.3.1 Structure and description of steps 11
2.3.2 Characteristics of steps 13
2.3.3 Data Specifications 16

2.4 Expertise and experts **24**
2.4.1 Problem-oriented expertise examples 25
2.4.2 Computational complexity 27

2.5 Problems and tasks **27**

3 USERS **30**

4 GOAL AND REQUIREMENTS **31**

5 PROPOSAL **37**

5.1 DESIGNER range **37**

5.2 Architecture of the system **38**

Main functions **42**

5.4 Inputs and outputs **44**

5.5 Data model **46**

5.6 Necessary hardware and software **46**
5.6.1 OO CASE and UML 46
5.6.2 Rational Rose 47
5.6.3 MATLAB 47

5.7 Proposed criterion and method for the evaluation of variants **49**

5.8 Proposed variants **49**

6 SCHEDULE OF THE PROJECT CONTINUATION 50

6.1 Stages of Project 50
6.1.1 Prototype 50
6.1.2 Experience gained from the prototype 51

7 ESTIMATE OF THE COST 52

8 ORGANIZATION OF THE PROJECT 53

8.1 Standards 54
8.1.1 Terminology Stabilization 55

9 CONCLUSION 55

REFERENCES 57

INDEX 58

ABBREVIATIONS

A&D	Analysis and design
ABET	Adaptive Bayesian Estimation Toolbox
API	Application Programming Interface
ARX	Auto-Regression models with eXternal inputs
AS	Adaptive Systems (department)
CAD	Computer Aided Design
CASE	Computer Aided Software Engineering
DB	Data Base
GA	Grant Agency
GUI	Graphical User Interface
ISO	International Organization of Standards
IT	Information Technology
KB	Knowledge Base
LQG	Linear models, Quadratic criterion, Gaussian disturbance
LQ	Linear models, Quadratic criterion
MIMO	Multiple Input Multiple Output
MISO	Multiple Input Single Output
OO	Object-Oriented
PID	Proportional, Integral, Derivative
PLC	Programmable Logic Controller
RR	Rational Rose
SISO	Single Input Single Output
UML	Universal Modeling Language
ÚTIA	Institute for Information Theory and Automation

1 INTRODUCTION

This Feasibility Study (Study) is the first part of the **DESIGNER 2000 Project (**abbreviated **DESIGNER** or **Project)** that is going to build the design tool with the same name.

The overall Project's **goal** *is to improve the accessibility and usage of algorithmic and software CAD tools for the systematic probabilistic* ***design*** *of adaptive systems*. It is solved in the Department of Adaptive Systems (AS), Institute of Information Theory and Automation, Czech Academy of Sciences. It is financed from the grant "Adaptive Systems: Theory, algorithms and software practice (GA AV ČR No. 102/97/0118)" [9]. DESIGNER will support the design of LQG adaptive controllers (Linear models, Quadratic Criterion, Gaussian disturbance), multidimensional case (MIMO - Multiple Input Multiple Output).[1]

The Project relies on *availability of necessary theory, algorithms, software and expertise*. The Bayesian theory [16] integrates the whole design, beginning from data handling, through estimation of the model of controlled system, to the control design. The approach is oriented mainly on ARX (auto-regression models with external inputs) model.[2] It is a continuation of the previous project, the DESIGNER [13]. The **tools** are available in AS and are provided by ABET (Adaptive Bayesian Estimation Toolbox) software [10][11].

The **Study** is going to

1) clear basic properties of the DESIGNER,
2) including the possibility to implement it,
3) evaluate whether the overall goal is attainable without any important negative side-effects,
4) support the decision whether to continue with the Project. To reach it, this Study
5) evaluates the current state of reached solution (Chapter 2),
6) identifies users (Chapter 3),
7) evaluates users' needs; refines the goal; refines functional requirements, including data-collection requirements and data-processing requirements; data correctness and relevance (they should be up-to-date); users' interface; performance; reliability; personal, organizational, economical and technical support; and safety requirements (Chapter 4),
8) proposes basic architecture of the DESIGNER, data content and DESIGNER's relationships to its environment; main functions; inputs and outputs; a rough data model and main necessary hardware and software properties (Chapter 5.1-5.6),
9) proposes criteria and method for the evaluation of proposed variants (Chapter 5.7),
10) presents proposed variants with theirs organizational, personal, technological and economical consequences and recommends the most suitable variant (Chapter 5.8),
11) proposes the schedule of the Project continuation (Chapter 6),

[1] We decided that other tasks, like prediction, fault detection, fusion of information sources are not to be solved, now.

[2] The common misunderstanding is: As the approach is based on regression, similar results can be obtained by other regression-oriented approaches, too.

12) estimates the cost of the Project and profit from the DESIGNER (Chapter 7),

13) proposes the organization of the Project continuation (Chapter 8).

The Study also documents the results of preliminary analysis and design that has been done. This whole approach is based on [18]. The Project is using an *object-oriented analysis, design*, and **UML** (Universal Modeling Language) approach [17].

The grant, covering Project has also an *educational part*. This has a close relation to the other part of the grant, to our design part. As Project is now more advanced, compared to the development of the educational part, and as some unification between these two parts is desirable, the educational part will be built in a similar way, as the design part. It means, there are no requirements on Designer originating from the educational part of the grant.

When one of the authors[3], unfamiliar with previous development, begun to study the approach of the AS Department to the advanced adaptive control, he early met the **objection** that *the corresponding tools are rarely used* [9].[4] But, then he realized that this objection had been risen already several years ago [6]. This situation persists in spite of the fact that a significant work was done in this direction. This lead to the conclusion that the *whole problem could be a rather complex one*, that a very careful solution is necessary and that ***all-available means should be used***, to succeed.

The **roles of the authors** differ. Jindřich Bůcha should aid the other authors, who are at the same time expert users, to create their system[5]. It means he is responsible for the overall Project, its methodological level, consistency, and inclusion of fresh ordinary-user-oriented viewpoint and elucidation of the other authors' knowledge. The others are both responsible for the correctness and completeness of the task and problem description or are playing the role of expert users and providing the initial set of user requirements. On top of it Miroslav Kárný is a spiritual leader of the Project, Petr Nedoma is specialized on ABET and estimation parts of the design, Josef Bohm and Jiří Rojíček are oriented on criteria setting and control design.

This Study is available in *electronic form*. We thought about environment, which it should be created in. There were two possibilities, i.e. MS Word and LaTex. The advantages of MS Word are:

1) the possibility to use directly all tools of MS Office, especially Excel table processor, presentation tool Power Point, organizational Outlook and link to HTML and Internet,

2) the easier connection to other tools, like CASE and possibility to directly overtake pictures and other results from these tools,

3) the proximity to the expected user environment,

4) it has some useful functions that LaTex hasn't, e.g., grammar corrector, better manipulation with pictures,

5) it is used by official bodies, like GA AV ČR, GA ČR, EU, NATO,

6) the existing knowledge of the first writer of the Study.

The advantage of LaTex is in its

[3] Jindřich Bůcha

[4] Outside AS they are not used at all.

[5] User should be maximally involved in the design of his system.

1) prevailing usage in the Institute and AS Department,
2) reliability for the processing of larger documents,
3) orientation on mathematical texts and symbolic references.

The disadvantage of LaTex is in its not up-to-date support. (We tried rtf2latex and html2latex and found that many important especially new document features are not supported.) According to aim to minimize the necessary overall workload, we have chosen MS Word.

Acknowledgement
This project is partially supported by the grant "Adaptive Systems: Theory, algorithms and software practice" GA AV ČR No. 102/97/0118. We also thank to all that helped us, especially to Luděk Berec, Pavel Ettler, Jaromír Fessl, and Rudolf Kulhavý.

2 CURRENT STATE

In this Chapter we are describing and evaluating the current state of the DESIGNER. We provide the results of the preliminary analysis here. They are transformed into requirements and constitute a basis for the preliminary proposal of the problem solution. The described entities are structured and named in such a way to maximally comply with the proposal (of Chapter 5). In this Chapter, there are also listed relevant problems.

2.1 Basic terms

We're beginning by introducing basic terms that we'll consistently use in the Study and Project: We feel it is very necessary to stabilize these terms, as they, at least some of them, have not been used consistently.

Controlled System is the part of external world, that part, a user[6] wishes to control.

Signal is a type of data generated by or influencing the controlled system or controller. It can have measurable values.

Input is a signal, which can be directly manipulated, influencing the controlled system.

Output is a signal, which can be influenced by the user indirectly only.

Disturbance is a signal influencing the controlled system, which cannot be influenced by user even indirectly. It may be both measurable and non-measurable.

SISO is a controlled system with single input and single output.

MISO is a controlled system with multiple input and single output.

MIMO is a controlled system with multiple input and multiple output.

Multidimensional system is MISO or MIMO.

Model is a description of simplified relationships between signals. It is taken from controlled system viewpoint.

[6] We distinguish between user, in an ordinary meaning, and User, in a meaning of design entity.

Controller is a mechanism that can observe outputs and produce inputs of the controlled system.

Control is the overall activity of the controller.

Adaptive controller is a controller that includes estimation and/or tracking of characteristics of the controlled system.

Control problem is a user's state when he wants to implement on controlled system his aims, under given restrictions and with available knowledge.

(Control) design is a process of finding controller that solves the control problem.

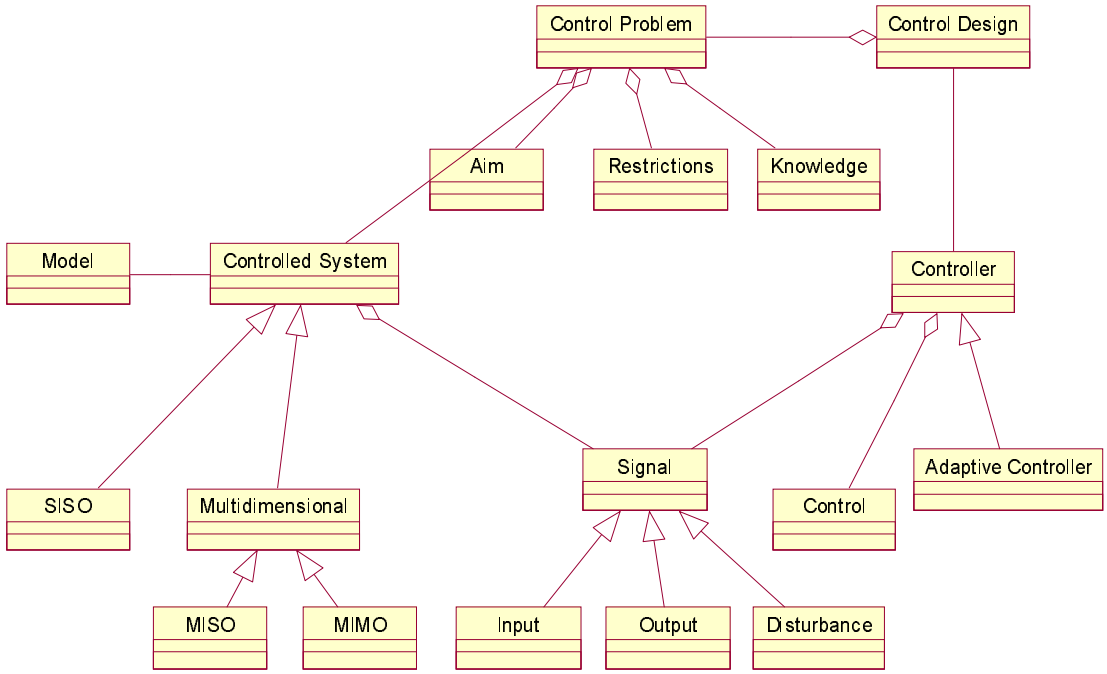

Figure 1: Relations among Basic Terms

The relations among these terms are illustrated in Figure 1. We use UML (Universal Modeling Language) graphical language in our figures. The short explanation of used **language symbols** follows: Named *rectangle*, with two additional lines inside, means a class, e.g. Controller. The class describes its objects. Named *man icon* means an active (external) class, e.g. User (see e.g. Figure 2). Controlled System is considered active as the man icon indicates (see e.g. Figure 2). (Named) *line* connecting two classes means an association relation, e.g. Interaction (see e.g. Figure 2). It is used for a data flow, event and other relations. (Named) *arrow with a diamond* head means that a class is a part of another class. For example, Signal is part of Controller. (Named) *arrow with a triangle* head means a class-subclass relation. For example, Input is a subclass of Signal.

2.2 Overall description and evaluation

We begin the initial exposition of the DESIGNER by the describing its **historical context**. The ancestors of the DESIGNER are **previous projects**, Xlab, DESIGNER (Dlab) [13][14], ABET [10][11][12], ABET97 [15]. The driving force of DESIGNER development was both the development of algorithms and technical development. For example, the important reason for

the Dlab development was that there were technical restrictions related to Dlab, i.e. matrix size limited maximally to 64 kB and work under DOS. The reason for transition to ABET was availability of MATLAB. The reason for transition to ABET97 was the availability of new tools in MATLAB version 5 and the need to reorganize ABET, concerning development of DESIGNER 2000. There are other differences between these ancestors, but they are not important from the user's point of view. They are important from the implementation point of view, as these ancestors are founded in different implementation environments.

The **basis for DESIGNER** is the set of ABET97 classes and functions and so called covering functions. **Covering functions** are mostly non-overlapping functions, covering all ABET functionality. They should hide some of ABET details from a user.

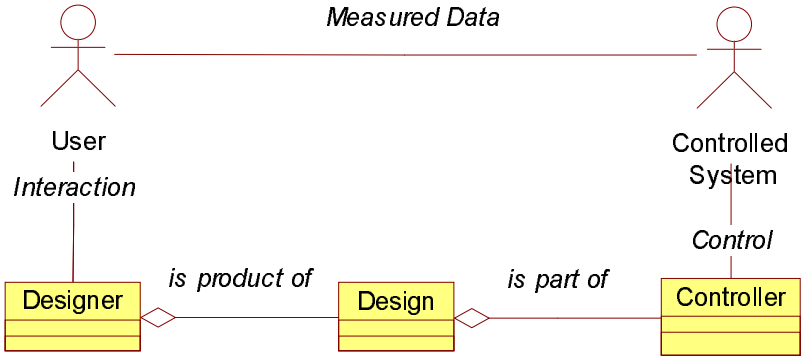

Figure 2: Context Diagram of Design

On the **first level of description**, the overall **context** of the design is shown in Figure 2. There is the class Designer, i.e. the main and only one process, the external entities (actors) User and Controlled System and the relations (data flows) Measured Data, Interaction and Control. The product of Designer is Design[7]; the part of Design is Controller. All these entities are described in more details in the Table 1.

Table 1: Main Entities of the Current Design

No	Entity	Type of Entity	Description
1	**Designer**	main process	It is a set of programming tools, i.e. subroutines, classes, objects, helps, case studies, demos and documentation of ABET and other tools of MATLAB. They allow the User to produce Design and at the end to create and run a program Controller. It can produce Inputs of the Controlled Systems. It is a kind of user environment. It is oriented to a *well-informed user*.
2	**User**	external entity	It is User in sense of Chapter 3.
3	**Controlled System**	external entity	See Chapter 2.1. At present, it is only simulated one or a laboratory pilot plant.
4	**Measured Data**	data flow	It is streams of signal values measured on Controlled System. It is used for the estimation of Controlled System model. Currently, simulated or pilot plant generated

[7] We will use the "design" for the whole process and the "Design" for product of this process.

No	Entity	Type of Entity	Description
5	**Interaction**	data flows	They are data flows between User and Designer consisting, e.g. of Design Specification and Design Response.
6	**Control**	data flows	These are data flows between Controlled System and Controller consisting of Measured Data and Control Action.
7	**Design**	data set	It is a product of the User and Designer's interactive cooperation, i.e. the program describing the controllers and data file specifying optional parameters of the controller.
8	**Controller**	program, process	It is a Designer's product, which is used for the control validation and can be used for routine control.

In this figure and table, everything is clear, well defined and well structured (we hope). The reality was not so bright. It *has had to be cleared, refined, named, defined and structured* in the course of the Project. This more precise picture is used for the description of current state.

The work on DESIGNER has been developing for many years without systematic computer-aided analysis. In this way the *necessary **tools** have been developed*. The existence of these tools is an *advantage*. The *disadvantage* is that these tools have been created in a relatively independent way. They have created their own concepts, terminology and language. This is documented later by Table 14: Synonyms and Table 3: Data Specifications in column Data Names. It is not easy to unify it as it means overcoming subjective standpoints and other obstacles. This unification is a long-term process consisting from many small partial steps, resulting in **many small Project's changes**. It is suitable to support this by some tool. (Our proposal is an OO CASE.)

That has direct relation to the **state of the documentation and tools.** It is not quite satisfactory from the viewpoint of our Project. The documentation has not been done for the user external to AS. It has been done for program creators, with assumption that everybody can ask about everything. It is in state that

1) it hasn't been tested by external users,
2) it hasn't been fully completed, made consistent and correct,
3) it hasn't been automatically updated,
4) the DESIGNER has been under permanent development.

There are errors in software. Some are already known. There is a **quality-of-programs** procedure, but it is not strictly (consistently) realized. It consists from walkthrough through programs and from presentation of programs showing they are working. Tests of programs are not very tough. The final tests are let to the user.

To conclude: To use the previous DESIGNER has meant to program in its language and system. It is not easy, as the problems are not simple, system and language is neither stabilized, nor well documented, nor very understandable, nor very reliable, nor fully implemented. The **requirements** resulting from this analysis are clear: *Make it easy, stabilized…*

2.3 Detailed data flow

2.3.1 Structure and description of steps

On the **next level of description**, there are the following **basic steps** in the design. The steps are devised with assumption that the User will be able to control the design after each step. For example, he will decide whether to continue or to return to previous step(s).

S1 Data Handling

S2 Prior Knowledge Processing

S3 Structure Estimation

S4 Forgetting Estimation

S5 Parameter Estimation

S6 Control Design

S7 Control Validation

These steps are described in more details in the following. The description is not regular. This reflects the sate of the analysis: It is only rough, shallow and not systematic one. On the other hand, we want to describe all, even partial results that could be used in the following stages of the Project.

The steps are described on a more detailed level in following. Some steps contain **substeps**[8]: If no substeps are listed, then the corresponding steps have "no structure" (from users point of view). Some substeps are not described if their meaning is self-explaining. Some substeps are supported by ABET. Some are not. The step *structure should be hidden* to the User.

S1 **Data Handling** provides the information obtained by handling (measured) data.

- Designing Data
  - **Selecting Sampling Period** (not supported in ABET)
- Getting Data
  - **Importing Data**: There is a possibility to import data from a file with ABET function, but there are mainly historical reasons justifying it, as in MATLAB there are efficient tools for this
  - **Simulating Data**
- Examining Data
- Plotting Data (not supported in ABET, supported by MATLAB)
- Preprocessing data
  - **Selecting Data Ranges**: It is often the case that the whole data isn't suitable for

[8] Descriptions associated with these steps shall be refined and/or extended in the next stage of the Project in such a way that they could be used as texts in the User's Help.

estimation, due to various undesired features (missing or "bad" data, outbursts of disturbances, level changes etc.), so that only portions of data can be used. For multivariable data, it is often advantageous to start by working with just some of input and output signals [7]. This substep is not supported by ABET, but it could be done by Expert User with the System Identification Toolbox. This is for the Expert User, only. This can be done on a level of presenting data in graphical form to let to the User possibility to check if data is a proper one.

- **Detrending** data involves removing the mean values or linear trends from the signals [7]. This is for the Expert User, only (not supported in ABET).
- **Normalization**
- **Outliers elimination**
- **Prefiltering** – to enhance or suppress various frequencies.
- **Resampling**
- **Change Delay**
- **Incremental Mode**

S2 **Prior Knowledge Processing** computes initial parameters distribution from prior knowledge.

S3 **Structure Estimation** searches the best model structure of the Controlled System, i.e. the qualitative dependency of outputs on delayed inputs and outputs that leads to the maximum likelihood.

S4 **Forgetting Estimation** selects the optimum forgetting technique.

S5 **Parameter Estimation** computes parameter distribution from initial parameter distribution, forgetting and handled data. Parameter distribution together with the model structure settles the description of Controlled System model.

S6 **Control Design** builds and sets up the Controller in order to reach the desired closed loop behavior. It involves mapping of the goal and restrictions into criterion; penalization is set so that the goal and restrictions required by the User are satisfied.

- Control Specification – gathering a specification of control problem, i.e., how it is to be controlled
- Control design, i.e., LQ control synthesis. LQ synthesis is based on a linear model and quadratic criterion. The approach used here uses as a linear regression model of the form:

$$y(t) = \sum_{i=i_0}^{n} b(i)u(t-i) + \sum_{i=1}^{m} a(i)y(t-i) = parms * z(t)$$

$$z(t) = [u(t), x(t-1)]$$

$$x(t) = [u(t), u(t-1), ..., u(t-n+1), y(t), ..., y(t-m+1)]$$

The quadratic criterion is described by

$$J = \min_{u(t0+1),\ldots u(t0+T)} E\{ \sum_{t=t0+1}^{t0+T} (q(y)(y(t) - w(t))^2 + q(u)(u(t) - u0(t))^2 \},$$

where u is input, y is output, t is discrete time, t0 is delay, and T is time horizon.

The optimization is based on the solution of the corresponding Riccati equation. Square root base iterations are used for the solution. The LQ synthesis generates an optimal control law in the form:

$$u(t) = -L\,x(t)$$

It is full SISO optimization for variable w(t) and u0(t) with multirate property. Additionally to control law L, Lw, Lu0 are provided in a matrix {clext} that respects the penalization of the set-point value and reference input in the criterion.

In case of adaptive controllers, it is done in each regulating step, otherwise only once.

- Plots Specification
- Write to Disk
- Controller in C++ code
- Control

S7 **Control Validation** - validation of Control Design; it checks the set up of the Controller either on a real system or on some user-provided model.

- Control

From the point of view of relation to real Controlled Process, there are two parts of the design, i.e. off-line and on-line parts. Into the *off-line part*, there belongs all, what can be prepared off-line, i.e. all steps. Into the *on-line part* there belongs all, what can be processed on-line, i.e. substep Control. The design is *mainly an off-line* process. Controller can run either in range of design or outside of this range and either off-line, with simulated data, or on-line, with real data. In an off-line case, the User should provide a model of Controlled System that would generate simulated data.

2.3.2 Characteristics of steps

Each step will *end numerically successfully,* i.e. there are no problems with convergence, numerical stability etc.

Steps S3, S5, S6 are followed by an **intermediate steps** where the *user can change the results of previous step* and influence the following design in such a way.

There is the difference in **stability of particular steps**: The steps (substeps, aspects) concerning Structure and Parameter Estimation, Control Validation, Data Handling are stabilized. Step Synthesis and multidimensional cases are not stabilized so much, i.e. there has not been gained extensive experience and these steps and case may be further refined.

Steps' inputs and outputs overview and other *characteristics* are provided nowhere. We collected this information in the following table (Table 2). We are using here commonly used names for M-files. They are sometimes changed, little bit, to correspond more to the standards. Names for data are not so stabilized and we are using user-oriented names as we have collected them from documentation, program comments etc. For each step there are listed the corresponding main M-files. From around 150 of ABET M-files (170 KB of source code) only several are directly corresponding to the steps. For each step, with the exception of Data Handling, there shall be at least one covering function. Now, there are ready four such functions: Eststr, Estpar, Estfrg and

Prior[9]. The last two are in preparation.

Table 2: Steps Characteristics

No	(Sub) Step No	Step or Substep Name	Responsibility	M-file	Input from System	Input from User	Output
1	1	**S1 Data Handling**	Nedoma			Data	Prior Information, Data
2	1.1	**Designing data**		Chnconst; Sigconst	Channel Meanings [, Channel Labels[, Name]]; Signal Sources	Goal, Prior Knowledge, Restrictions	
3	1.1.1	**Selecting Sampling Period**	Rojíček				
4	1.2	**Getting Data**					
5	1.2.1	**Importing Data**		Dflconst, MATLAB tools	No of channels, File name, Precision		
6	1.2.2	**Simulating Data**		Simcon1, …10	Models, Covariance Matrix, Distribution; [No of channels[, Initial Value]]; [Means]; Table; Input Channels, Function, Arguments; Pointer to permanent matrix, Model		
7	1.3	**Examining Data**					
8	1.4	**Plotting Data**					
9	1.5	**Preprocessing Data**		Preconst, Preset, Pre2chn	Maximum Delay, Sampling Frequency, Delay, Limits, Scaling, Incremental Mode, Filtering, Any task		
10	1.5.1	**Selecting Data Ranges**		Preset	See Preprocessing Data above		
11	1.5.2	**Detrending**					
12	1.5.3	**Normalization**		Preset	See Preprocessing Data above		
13	1.5.4	**Elimination of outliers**					
14	1.5.5	**Prefiltering**		Fltconst	Parameters, State		
15	1.5.6	**Resampling**		Preset	See Preprocessing Data above		
16	1.5.7	**Change Delay**		Preset	See Preprocessing Data above		
17	1.5.8	**Incremental**		Preset	See Preprocessing Data above		

[9] We use here names with capitals. In ABET, these program names are used with lower case letters.

No	(Sub) Step No	Step or Substep Name	Responsibility	M-file	Input from System	Input from User	Output
		Mode					
18	2	**Prior Knowledge processing**					
19	3	**Structure Estimation**	Berec, Bohm, Kárný	Eststr	Data, Output channel, Regressor, Prior Information, Workspace	Output Channel, Regressor	Regressor
20	4	**Forgetting Estimation**					
21	5	**Parameter Estimation**	Kárný, Nedoma	Estpar, Frgconst	Data, Output Channels, Channels and Delays, Forgetting, Prior Information; Forgetting Type, Model, Forgetting	Data (2x), Forgetting	Model, Parameters ?, Eth
22	6	**Control Design**	Rojíček	Ulimits	Model, Output Channels, Set-Point,	Limits on Inputs Expectations, Limits on Inputs[, Guess of Penalization Matrix Qu, Guess of Penalization Matrix Qy]	Penalization Matrix Qu, Penalization Matrix Qy, Used Set-Point
23	6.1	**Control Specification**				Design Specification, i.e. 1) Control Specification, 2) Data Specification, 3) Prior Knowledge, 4) Prior Model Structure, 5) Expert Specification	
24	6.2	**Synthesis**		Syncons	Output Channels, Input Channels, Horizon, Penalization, No of Iterations, Mode of Synthesis		
25	6.3	**Plots Specification**					
26	6.4	**Write to Disk**					
27	6.5	**Controller in C++ code**	Nedoma				
28	6.6	**Control**		Syststart, Regconst	Output channel, Input Channels, Mode of Regulation, Limits on Inputs, Limits on Inputs Differences, Channel with Set-Point, Channel with Input Reference		

No	(Sub) Step No	Step or Substep Name	Responsibility	M-file	Input from System	Input from User	Output
29	7	**Control Validation**	Bohm			Mode of Run, Sampling Period, Plot Specification	Plot

The table corresponds to our knowledge and our possibilities. It was reworked many times, it is not fully up-to-date and it reflects the current state of insight, in this way. The problem is that the "current state" is "changing". For each step, we are listing all inputs and outputs. In the column "Inputs from System" we are listing input data produced by the system or stored by the system after they were inputted from the User and used in some previous step. There are specially underlined (by listing in the special column) user's first inputs to steps.

The current DESIGNER **processing** has two main parts. The declarative part, where ABET objects and relations among them are established and the implementation part, where all steps are really run and results presented. The relations are established either by making some created objects parameters of some m-functions or by attaching tasks specified by some objects to channels; in such a way the objects of classes ide, mod, pri, sim, sta, syn, reg can be attached.

The current DESIGNER **user's interface** can be characterized

1) by being *oriented on particular DESIGNER entities*, like classes, m-functions, their documentation, examples of their utilization, by not being oriented on the overall design,
2) by *laborious* and difficult access to necessary knowledge, i.e., the User does not know and is not told by the system what knowledge he[10] is supposed to have, where to find it,
3) by prerequisites that the User must satisfy, e.g. he must know MATLAB to understand examples,
4) by not paying attention to some basic user's comfort, e.g. there aren't variables' descriptions in plotted diagrams,
5) by not paying attention to strictly systematic approach to communication with him, e.g. the same dialog proceeds in different ways. It is understandable as the only User has been the member of AS Department, till now. It has been easy for this user to ask and get necessary information.

System Identification Toolbox [7] uses multiple models.

The **algorithms** are hidden in the ABET software, they are not accessible to the ordinary users, they are available to expert user.

2.3.3 Data Specifications

The following table describes hopefully all data that could be used in DESIGNER. The table is "sparse" in a sense that there should be more information. But, in this state it shows what needs to be done: It should be filled, e.g. in Description column there should be at least data description from User point of view and description how data is represented. It will be done in

[10] In such situations, we use references to male gender without any prejudice to female gender. You can freely replace gender according to your taste.

the next Analysis and Design stage.

This table is based on existing documentation both explicitly written and implicitly contained in programs. We tried to do our best. But, it is far from being finished and far from being ideal. It was created gradually and only gradually have we realized what it should contain. Due to this, but not only to this, it may not be fully consistent and it may not cover everything. It is meant as working tool that should be farther gradually refined. In column "Description", there should be a meaning and representation of data. We propose the level of that Description to be such that shall fulfill needs of the user, when he is asking for the Help. In case of need it is possible to rearrange the table, e.g. to sort it in another way.

Table 3: Data Specifications

No	Data Name	Data Name(s) in ABET	Output of	Input to	Data Description
1	**Allowed Computational Time**		User	GUI	The maximum time which the User allows to spend on computer.
2	**Alternative Model**	Alt		Estfrg, estpar	
3	**Any Task**	any task	Expert User	Preset	Task structure, i.e. any program; it isn't listed in ABET UG
4	**Arguments**	arg1, arg2, arg3	Expert User	Simcon5	Arguments stored inside the object; i=1..3
5	**Auxiliary Output Signals**		User	Chncons t	Specification of auxiliary outputs; matrix of channel numbers. These are of two types: 1) supporting measured outputs and 2) external disturbances. It is part of Data Specification.
6	**Control Law**		Syncons	Expert User	
7	**Covarianc e Matrix**	COVAR	Expert User	Simcon1 ; Simcon3	[Simcon1]: Noise covariance matrix (default is eye(..)); [Simcon3]: covariance matrix. It may be entered as: scalar => COVAR * eye(n, n), vector => diag(COVAR), matrix (default is 1)
8	**Criterion Structure**	horizont	Expert User	Syncons	Synthesis horizon specifies what actions represent the criterion, it is in the form of a character string, it consists of a sequence of letters corresponding to penalizations of outputs (abbreviated by P or 1), inputs (abbreviated by Q or 2) and minimizations (abbr. as M or 3); example: PQM is a valid sequence, similarly (PM, QM, PPQQM, PPQQMM ...); the horizont should contain at least one minimization (usually at the end). [K:] It has relation to complexity considerations.
9	**Cut Off Frequency**	cut off	User	Prior	[Prior]: [output, [frequency for each input]]; maximum frequency, which can be processed by each output for each input. It is part of Prior Knowledge.

No	Data Name	Data Name(s) in ABET	Output of	Input to	Data Description
10	**Data**	Data, data	User	Eststr, Estpar, Estfrg, User, Prior	Series of input and output data of modeled system. In step S5 input data may be either modeled or directly measured in environment ([Eststr]: matrix of measurements stored columnwise; [Estpar]: if scalar ='number of channels' is used, workspace is not cleared and encapsulated object is returned as Eth; [Prior]: data matrix (dynamic measurements)). Data can be measured in real (non-discrete) times. It is part of Data Specification. There is a gradual transition from Prior Knowledge to Data; some Prior Knowledge is expressed by Data, e.g. knowledge about channel relations (valid equations), simulation models etc.
11	**Data Set Name**		User	Dflconst	Data set name, where Data are stored. It is part of Data Specification.
12	**Data Set Structure**		User	Dflconst	Structure of data set, where Data are stored. It is part of Data Specification.
13	**Data Sources**	filename	User	Dflconst	Matrix of data sources types {"data set", "simulation model", "expert knowledge"} [:K]. Knowledge about particular channels is conveyed to Chnconst via Sigconst. [Dflconst]: DOS filename = (name). It is part of Data Specification.
14	**Delay**	Delay	Expert User	Preset	Number of simulation steps
15	**Distribution**	Distribution; noiseType	Expert User	Simcon1; Simcon3	Distribution of noise: <0 - no change, 0 – zero, 1 - normal distribution, 2 - exponential distribution. 3 - uniform distribution, 4 - lognormal distribution, 5 - Cauchy distribution; if distribution is a scalar, all elements will have the same distribution given by it; (default is 1)
16	**Dominant Time Constant**	time constant	User	Prior	[Prior]: [output, [time constant from each input]]; for each input/output pair specification of dominating (max) time constant. It is part of Prior Knowledge.
17	**Encoded Prior Knowledge**	pri	User	Eststr, Prior	
18	**Eth**	Eth	Modconst	Ulimits, Regcons t	[N]: Time dependent average values of parameters; represented by matrix with row corresponding to time, column corresponding to parameter; it has a Global character, it isn't passed as function input parameter.
19	**Expert**		User	GUI	In some situations, there can be different opinions on solution of this situation among experts. The User can choose which Expert he prefers.
20	**Filtering**	Filtering	Expert User	Preset	Object flt class

No	Data Name	Data Name(s) in ABET	Output of	Input to	Data Description
21	**Forgetting**	frg	Estfrg	Estpar, Frgconst	Specification, whether estimate parameters with forgetting ([Estpar]: frg - exponential forgetting, default is 1, [G, 13]: forgetting factor). It is specified by Estfrg.
22	**Forgetting Type**	type	Expert User	Frgconst	1 - exponential, 2 - directional, 3 - tracking
23	**Frequency Characteristics**	amplitude	User	Prior	[Prior]: [output, [amplitude, omega, shift, input]]; for each input/output pair specification of response (amplitude and shift) to unity sinusoid of omega frequency. It is part of Prior Knowledge.
24	**Function**	fun	Expert User	Simcon5	Function call (character string) to be evaluated
25	**Functional Dependency**	Selection Mask	User	Strset	Describes what entities of maximum Model Structure are relevant to the output. It is assigned to each output after maximum Model Structure is build. [Strset:] prior dependency guess for a channel, values 1: is present, 2: is probably present, 3: is probably not present, 4: is not present. It is part of Prior Knowledge.
26	**Guess of Penalization Matrix Qu**	qin	Estpar	Ulimits	Guess used to speed up processing; if the guess is good then the computation is quicker and vice-versa.
27	**Guess of Penalization Matrix Qy**	pin	Estpar	Ulimits	Guess used to speed up processing; if the guess is good then the computation is quicker and vice-versa.
28	**Channel**	channel	Expert User	Pre2chn	channel connect with pre-processing (1)
29	**Channels**	Channels	Expert User	Pre2chn	Class structure, (last)
30	**Incremental Mode**	incremental mode	Expert User	Preset	'on' | 'off'
31	**Initial Model**	Ini	Prior	Prior, Estfrg, Estpar	
32	**Initial Value**	val	Expert User	Simcon2	Initial value (default 0)
33	**Input Signals**	chanu, uchn, inputs	User	Chncons t, Syncons , Regcons t, Simcon5	Input channels, i.e. matrix of potential input channels; Input channel ([Syncons]: chanu, [Regconst]: uchn), [Simcon5]: Input channels (default 0). It is part of Data Specification.

No	Data Name	Data Name(s) in ABET	Output of	Input to	Data Description
34	**Inputs Differences Satisfaction**		Syncons [f]	User	These are probabilities that Inputs will satisfy Limits on Inputs Differences. It is part of Design Response, i.e. Level of Restrictions Satisfaction.
35	**Inputs Satisfaction**		Syncons [f]	User	These are probabilities that Inputs will satisfy Limits on Inputs. It is part of Design Response, i.e. Level of Restrictions Satisfaction.
36	**Limits**	Limits	Expert User	Preset	[minimum, maximum]
37	**Limits on Inputs**	umax, User's Requirements, constraint: limits for input	User	Regconst, Ulimits	[K:] Required ranges (min, max) of inputs; [R]: (u_min, u_max)
38	**Limits on Inputs Differences**	udmax	User	Regconst	[K:] Required ranges (min, max) of inputs' changes; it will be very often required
39	**Limits on Inputs Expectations**	Requirements Expectations	User	Ulimits	Probability level with which the User expects that limits required for Inputs and their Differences will be adhered to. Default can be e.g. 0.9. [R]: Required probabilities for achievements of requirements. Constraint expectation (probability).
40	**Limits on Outputs**		User	Syncons [f]	It is part of Goal.
41	**Limits on Outputs Differences**		User	GUI	It is part of Goal.
42	**Limits on Outputs Expectations**		User	GUI	Probability level with which the User expects that limits required for Outputs and their Differences will be adhered to. Default can be e.g. 0.9. It is part of Goal.
43	**Limits on Overshoot**		User	GUI	It is part of Goal.
44	**Limits on Steady State Values**		User	GUI	It is part of Goal.
45	**Loss**				Is calculated to ranges [:K]
46	**Max Control Period Factor**	m	GUI		
47	**Max Input Delay**		User	GUI	Maximal input delay used for specification of Model Structure.
48	**Max Output Delay**		User	GUI	Maximal output delay used for specification of Model Structure.

No	Data Name	Data Name(s) in ABET	Output of	Input to	Data Description
49	**Maximum delay**	MAXDELAY	Expert User	Preconst	Maximum delay planned
50	**Means**	means,	Expert User	Simcon3	Noise offset (the number of channels is deduced from the argument length) (default is 0)
51	**Min Control Period Factor**		GUI		
52	**Min Input Delay**		User	GUI	Minimal input delay used for specification of Model Structure.
53	**Min Output Delay**		User	GUI	Minimal output delay used for specification of Model Structure.
54	**Mode of Regulation**	mode	User	Regconst	'pid' (1), 'arx' (2), 'Arx' (3)
55	**Mode of Run**	Mode of Run		S5	Specification of mode which is to be run: and data from the modeled system
56	**Mode of Synthesis**	mode	Expert User	Syncons	The way how the task is used, it enables to call the synthesis either only once in the initialization stage (Mode of Synthesis =1, non-adaptive) or its call is controlled in the simulation experiment by a channel to which it is attached (Mode of Synthesis =2, adaptive). [B:] it will be internal parameter.
57	**Model**	Model, I, Ide	Expert User, Estpar, Prior	Simcon10, Estpar, Frgconst, Ulimits	Model is a structure described below. It can be used for different purposes: as initial, prior, alternative ARX, identified, default models. The MISO ARX model fields are: Output Channel, Model Structure, Output Noise Variance, Mean of Coefficients Distribution, Encoded Covariance of the Coefficients, Number of Degrees of Freedom. Markov chain. Object containing vector of parameters and their covariance matrix, plus some other information (not interesting for user). Output of Estpar. , ([Frgconst]: identification, [G, 13]: identification object); [R]: identification results (structure IDE, and only identified structure and pdf of identified system parameters are needed).
58	**Model Structure**	Str, ord, chtd, maxreg	Eststr	Eststr, Estpar, Estfrg, Prior, Ulimits	Model Structure is represented by object corresponding to the two rows matrix, where in the first row there are channel numbers, in the second row there are delays corresponding to the logical channels from the first row; e.g. [1 2 2; 1 0 1]; ([R; G, 2]: Str, [Eststr]: ord - Model Structure order or description; [Estpar]: chtd - list of channels and delays in Model Structure, [G, 1]: maxreg). Channel with number 0 with delay 1 means the presence of offset. In simplified Model Structure representation, there are used Min Input Delay, Max Input Delay, Min Output Delay, Max Output Delay. In this case, all delays between min and max are assumed in Prior Model Structure.

No	Data Name	Data Name(s) in ABET	Output of	Input to	Data Description
59	**Models**	Models	Expert User	Simcon1	ARX model or array of pointers to models for individual channels
60	**No of Channels**	Nochannels; nochn	GUI	Dflconst, Simcon2	[Dflconst,]: Number of channels recorded (1); [Simcon2]: number of channels (default 1)
61	**No of Iterations**	nstep	Expert User	Syncons	Number of iterations
62	**Object Name**	Name	Expert User	all object constructors	Object name used in previous Dlab version; in ABET it is from historical reasons.
63	**Output Channels**	Ychn, y, chany	User	Chncons t, Eststr, Estpar, Ulimits, Syncons , Regcons t	output channels [Eststr]: Output channel; [Estpar]: the channel for which the estimation is done ([G, 17; Eststr, Estpar, Regconst]:Ychn, [G, 1]:y,[Syncons]: chany). It is part of Data Specification.
64	**Outputs Differences Satisfaction**		Syncons [f]	User	These are probabilities that Outputs will satisfy Limits on Outputs Differences. It is part of Design Response, i.e. Level of Goal Satisfaction.
65	**Outputs Satisfaction**		Syncons [f]	User	These are probabilities that Outputs will satisfy Limits on Outputs. It is part of Design Response, i.e. Level of Goal Satisfaction.
66	**Parameter Range**	limits	User	Prior	[Prior]: [output, [channel, delay, min, max]]; for each output and each pair of limits (min, max) of parameter (specified by channel and delay)
67	**Parameter s**	Param, a	Estpar	, Fltconst	[R]: Vector of parameters (it is also part of Ide – for historical reasons), Param; [Fltconst]: a
68	**Parameters Available to User**		User	Syncons [f]	Parameters which are available to User during regulation (diagonal Penalization Matrix Qu and Qy). They could be specified. It is part of Miscellaneous.
69	**Partial Model Forgetting Factors**	frgs	User	estfrg	Can be specified in MATLAB form, e.g. as 0.8:0.01:1.
70	**Penalizatio n Matrix Qu**	Penalizati on Matrix Qu, upen	Ulimits	Regcons t	Input penalization matrix
71	**Penalizatio n Matrix Qy**	Penalizati on Matrix Py, ypen	Ulimits	Regcons t	Output penalization matrix; [R]: Py
72	**Plot**	Plot	Plots Specific ation	User	Plots corresponding to Plot Specification

No	Data Name	Data Name(s) in ABET	Output of	Input to	Data Description
73	**Plot Specification**	Plot Specification	User	Plots Specification	It is possible to ask for plots of the following variables: required value (set-point), response, input, error
74	**Precision**	precision	User	Dflconst	[Dflconst]: Precision - file type: ascii', 'float', 'integer', 'double', 'composed'
75	**Prior Model**	Pri	Prior	Estfrg, Estpar	Transformed prior knowledge about modeled system provided by the user - [N]: prior parameter distribution ([Eststr]: prior information structure)
76	**Prior Model Structure**	str	User	Eststr	Same as Model Structure
77	**Protocol**		GUI	User	Protocol of all system interactions with the User. It is addressed to the expert User, more or less, to support the diagnosis of errors. It is part of Design Response.
78	**Range of Signal**	range	User	Prior	Usual ranges of values for all signals [:K]. [Prior]: maximum of absolute values for each signal. It is part of Prior Knowledge.
79	**Range of Signal of Differences**		User		Usual ranges of differences for all signals [:K]; it is expected that only an educated user will provide this knowledge. It is part of Prior Knowledge.
80	**Riccati Matrix**		Syncons	Expert User	
81	**Sampling Frequency**	sampling frequency	Expert User	Preset	number of simulation steps
82	**Sampling Period**	Sampling Period		User	Sampling period for data from the real system in case of the corresponding mode
83	**Scaling**	Scaling	Expert User	Preset	[add factor, multiply by]
84	**Second Level Forgetting Factor**	frgup	User	Estfrg	
85	**Set-Point**	Set-point	User	Ulimits	Desired values of the controlled output
86	**Signal Description**	des	User	Eststr, Estpar, Estfrg, Prior	[PN:] Character string expressing that corresponding signal is ‘o’ controlled output, ‘i’ input, ‘a’ auxiliary output, '-' not used in processing.
87	**Signal Labels**	chnlab	User	Chncons t	([Chnconst]: chnlab). It is part of Prior Knowledge. They are labels (denotations) of channels shortly describing real (e.g. physical or chemical) meanings of channels including scales known to (specified by) the User. It is part of Data Specification.

No	Data Name	Data Name(s) in ABET	Output of	Input to	Data Description
88	**Signal Meanings**	Ide, description	GUI	Chnconst	[G, 10]: description; [Chnconst]: channel description (o, i, a) | number of channels (default 2)). The values must be translated from GUI.
89	**Signal Sources**	S1, S2, … S10	Expert User	Sigconst	Signal Sources
90	**Signal with Input Reference**	chanu0	User	Regconst	It is part of Data Specification.
91	**Signal with Set-Point**	chanw	User	Regconst	It is part of Data Specification.
92	**State**	z		Fltconst	
93	**Static Gain**	measurement, gain	User	Prior	[Prior]: [measurement (,level)], [Prior1]: gain (SISO case); [N]: measured values on channels in infinite time and corresponding probabilistic levels of credibility. It is part of Prior Knowledge.
94	**Steady State Signal**		User	Prior	It is part of Prior Knowledge.
95	**Table**	table, tabi	Expert User	Simcon4	[N]: specification of step function represented by matrix, in which the first row contains times, in which step function changes, and the second row contains changed function values; [Simcon4]: table, [G, 32]: tabi
96	**Type of Problem**		User	GUI	It can be: "regulation problem" or "servo problem". It will be used by GUI to ask for Set-point. It is part of Goal.
97	**Used Set-Point**	Used Set-Point	Ulimits(?)		Used set-point (will be specified as function by one parameter).
98	**Workspace**	workspace	Expert User	Eststr	The way of work with Encapsulated Workspace.

2.4 Expertise and experts

We distinguish two types of expertise: general and special. There is a **general expertise** obtainable from the experts specified in Table 4.

Table 4: General Expertise

No	Expert	Expertise/Note	Complexity
1	Berec	he left on October 15, but can cooperate	Structure Estimation
2	Bohm	Documentation, S3, S7	Prior Knowledge Processing
3	Kárný	Theory	Control Design
4	Kulhavý	theory	

5	Nedoma	software, architecture	
6	Rojíček	Documentation	Control Design
7	Tesař	task types	

2.4.1 Problem-oriented expertise examples

There is a **special expertise**, providing knowledge *how to solve* mainly the design problems, obtainable from the following experts[11]. We call this a problem-oriented expertise. Its sample is gathered in Table 5.

Table 5: Sample of Problem-Oriented Expertise.

No	Name	Rule	Source
1	Skeptical	The standpoint is little bit skeptical (conservative): There is a huge amount of possibilities to solve. It is impossible to solve all.	Bohm
2	Structure is Good	Only sometimes can experts specify who knows how to do it, that the structure is good.	Bohm
3	Able to Generalize	It is not sure that the experts will be able to generalize their specific knowledge, i.e. their ability to solve specific problems. The reason is that experts have experience with simple cases, only.	Bohm
4	Expert Adviser	Question is whether there must be or need not be an expert adviser to support the user.	Bohm
5	Real Data	The experience shows that it is very hard to obtain any reasonable results if real data are used. The reason is that the level of approximation is not taken into account in the whole approach. (But, Kárný does not agree.)	Bohm
6	Conflicting Knowledge	There are rules that Bohm would use, but Kárný would not. (This is not a problem, as conflicting knowledge can be used in KB.)	Kárný
7	Control is Dependent	Success of control is 90% dependent on the model. Estimation in 5 – 10% produces structures that he would refuse.	Bohm
8	Output is Independent	If the output is independent of inputs, i.e. in Model Structure there aren't parts corresponding to inputs, then the reason can be that there is no Prior Knowledge specified. There will be 80% of cases with independent output if there is no Prior Knowledge specified. The rest, 20%, is caused by the low robustness of controller. (Kárný: Rojíček tests Robustness. Bohm knows / feels something, not all. There are packages in MATLAB testing robustness. But theoretically, they should not fit with our approach.)	Bohm
9	No input Influences Output	If there is no input which influences some output, then either inputs are not correctly specified or you have bad Data.	Kárný
10	Incomplete Prior Knowledge	The estimated structure will correspond to Prior Knowledge. The problem with Prior Knowledge is that the User can provide incomplete Prior Knowledge.	Bohm

[11] There is also a tendency not to widen the extent of ABET project at present (in the first Project prototype part).

No	Name	Rule	Source
11	Only One Structure	If there is only one structure resulting from Structure Estimation then do not take it. (But, Kárný does not agree.)	Bohm
12	More Structures	If there are more structures resulting from Structure Estimation then take that which has the best physical interpretation, i.e., it hasn't got sparse Model Structure, i.e., there are no zero coefficients in transmission polynom; only after that choose that structure with the greatest probability.	Bohm
13	Flat Parameters Distributions	Bad Parameter Estimation can be recognized by the flat Parameters Distributions. In that case, knowledge of the Controlled System can be used. The flat distribution means that probabilities of the particular Parameters variants differ on level of 1%. The extents of difference of these variants are not important.	Bohm
14	Result Should Be Certain	If there is any result of the Design, then it should be very certain, e.g. 99%. What the certainty is the User should specify	Bohm
15	Offset is High	Penalize input differences or specify Limits on Input better	Kárný
16	Not Satisfied with Distribution of results	If you are not satisfied with the distribution, try to gather more data.	Kárný
17	Oscillations	If there are oscillations then choose higher penalizations	Rojíček
18	Wrong Data	The DESIGNER should protect the design from using wrong data. After each step, the check of results should be done with the aim not to allow to continue and to use this data if they are wrong. Each step can yield wrong data even if it ended numerically correctly. Experts: Kárný, Nedoma. (Kárný prefers logging off corresponding information.)	Bohm
19	Outliers	Apply after S1: If there are outliers in Data then delete them. Kárný: the "delete" is not correct, better is to space out some average values.	Bohm, Kárný
20	Confidence of Parameter Is Wide	Apply after S5: If the confidence interval of a parameter (derived from corresponding covariance matrix – Rojíček) is wide compared to value of the parameter then the model is not trusty and delete it. It is also the criterion for Data and Prior Information. This should protect against 80% of negative results in step S7.	Bohm
21	Gain Is too High	Apply after S7: If the gain is too high do not use the proposed controller.	Bohm
22	Output Compared to Measured	A good test to find whether the model is good is to take a close look at the model's output compared to the measured one on a data set that was not used for the fit ("Validation Data"). (Kárný: This would be a mixing of approaches, this is for the Expert User, only).	Ljung, 1-1
23	What Model Couldn't	It is valuable to look at what the model could not reproduce in the data ("the residuals"). This should not be correlated with other available information, such as the system's input. (Kárný: This would be a mixing of approaches, this is for the Expert User, only).	Ljung, 1-1

No	Name	Rule	Source
24	Main Loop	If there is a space in Inputs (Differences) Satisfaction, i.e. Inputs (Differences) Satisfaction > Limits on Inputs (Differences) Expectations, then this space can be "exchanged" for Outputs (Differences) Satisfaction by increasing Limits on Inputs (Differences) Expectations, and vice-versa, if there is a space in Outputs (Differences) Satisfaction, i.e. Outputs (Differences) Satisfaction > Limits on Outputs (Differences) Expectations, then this space can be "exchanged" for Inputs (Differences) Satisfaction by decreasing Limits on Inputs (Differences) Expectations. If there is no space either in Inputs (Differences) Satisfaction or in Outputs (Differences) Satisfaction, then try to gather more Data. In multidimensional case, it is let up to the User to choose which Expectations to change.	Kárný
25	Loop after Validation	Main Loop Rule that is to be applied after Control Validation	Bohm
26	Small Loops	After processing Prior Knowledge, the result can be presented to the User to check it. If the User is not satisfied, then there is an inconsistency in the Design Specification.	Kárný

It is not too large and representative knowledge sample. It is not possible to derive some definite conclusions about KB from it. But, we consider this a seed that will help and inspire experts to generate another knowledge.

2.4.2 Computational complexity

The **computational complexity** is an important aspect of the design. There is knowledge about this topic in AS for each function. It is not documented. As being at least partially of a heuristic character, we consider it in range of expertise. AS has experience with problems of the following dimensionality: 100 – 1000 data items, 5 x 5 (|Inputs| x |Outputs|) or 130 parameters which can take up to 10 min of computer time. A systematic attention and data gathering should be oriented on the problems connected with multidimensionality.

There are two areas of complexity, i.e. the complexity of on-line part of the design and the complexity of off-line part. It should be taken into account in the design. For the off-line part there are the experts specified in Table 4. In the design, there are critical steps of Structure Estimation and especially Control Design. The Control Design step is much more demanding and will depend on the user, how much time he may spend with computer, more or less.

Specification of requirements on the computational complexity description is given in Chapter 8.1 Standards.

2.5 Problems and tasks

Problems and tasks are connected with the Project. They are continuously discovered as the Project advances. They are listed in the Table 6. These problems are projected into requirements, later. Some of them were solved during preparation of this Study. They are marked as "solved". Some have not been solved yet and their deadlines influence the schedule of the Project.

There is also one special problem. We could not solve it: The User could have specified (in Prior Knowledge) that Controlled System is stable. But, *if it is not stable*, we do not know how to solve

it.

Table 6: Open and Closed Problems

No	Step	Stage	Problem - Task	Responsible	Deadline
1			Prepare computation of **Losses** in Control Validation. Rojíček will solve it by reusing results.	Bohm	no problem
2			How long to compute and When to start Control Validation	Bohm	
3			Compute better Satisfactions	Bohm	
4	S5	1	Prepare covering function for Control Validation	Bohm	solved
5	S5		The usage of **PID** controllers is possible for comparison, but only conceptually. It is not implemented. There is a toolbox supporting this kind of design, which seems could be used. Prepare corresponding function. Then parameters of PID controller should be collected together with data and knowledge for the design.	Bohm	
6	S5		**Consider computational complexity** in the case of on-line control and modify the corresponding function.	Bohm	Aug 30
7	S5		Made Regconst operational for all **Modes of Regulation**.	Bohm	
8	S5		Prepare **member function Synset** for accessing Control Law and Riccati Matrix, products of synthesis.	Bohm	ready
9	S5		Make Synconst to accept Parameters Available to User	Bohm	no problem
10			Compare our approach with PID	Bohm [:K]	
11	S1	1	Prepare function for handling of data from different standard data formats.	Bůcha	
12			Specify how to recognize the low quality of design.	Kárný	
13	S5		It must be specified experimentally, which value to take as default for the **level of certainty** of the overall design (90% - 99%).	Kárný?	try
14	S1	1	Prepare covering function for Data Handling	Nedoma	not the 1st stage
15	S1	1	Prepare function for the elimination of **outliers**.	Nedoma	not the 1st stage
16	S2		Experts can provide knowledge that there is a **dependency between some input and output**. Prepare function to support this.	Nedoma	solved
17			Prepare the description of Prior.	Nedoma	solved
18	S2		The **levels of credibility** of Static Gains should be computed from Ranges of Static Gains. If the User insist on one value, the consequences should be derived. The default levels are set on the noise level. Prepare corresponding function.	Nedoma	solved
19	S1		Prepare function for a systematic **filter design**.	Nedoma	->Bohm
20	S2		Prepare function for **specifying the control frequency**.	Nedoma,	

No	Step	Stage	Problem - Task	Responsible	Deadline
				Berec	
21	All		A systematic theoretical and experimental attention should be paid to and data gathering should be oriented on the problems of **computational complexity**: Specify what and how it influences the complexity. Find, possibly, experimentally.	Nedoma, Bohm	will think
22	S2	1	Cover **basic principles**, like the law of matter preservation.	Nedoma, Kárný	solved
23	S2		Now can be processed only the maximum part of **Usual Ranges** in range of Prior Knowledge processing. Prepare function to extend it to cover both maximum and minimum and current processing of maximum deleted from Prior. Also, Usual Ranges of Differences should be solved.	Nedoma, Kárný	solved
24	S1	?	Prepare function for **normalization**.	Nedoma	solved
25	S5	1	**S5 should produce** Inputs Differences Satisfaction, Inputs Satisfaction, Outputs Satisfaction and Outputs Differences Satisfaction.	Rojíček	
26	S4	1	Prepare covering function for Control Design, 3 functions - for each Restriction alone	Rojíček	
27	S4,S5	1	**MIMO -** only for Limits on Inputs	Rojíček	
28		1	Existing **programs** should be further **extended to process**: Limits on Outputs, Limits on Output Differences, Limits on Output Expectations, Limits on Steady State Values.	Rojíček	very open
29	S4		Today, program for Criterion setting, Ulimits, can **process only the simplest** kind of **restrictions**, i.e. Limits on Inputs, only, e.g. not Limits on Input Differences, Limits on Overshoot and not in combination ([R]: it isn't easy). Prepare function for the other restrictions.	Rojíček	
30			Compute no of iterations	Rojíček [:B]	
31			1) Inform and 2) try to persuade the potential users that the offered DESIGNER has a clear advantage for them (prepare several examples showing real advantages of our controllers over classical PID controllers - see problem above).		
32	?		Prepare procedure for a **variable transport delay** processing. It is not available, but it was prepared for an older system, SIC. Prepare the corresponding function.	Schier	solved ?
33			Analyze the possibility to design GUI under C++	Bůcha	
34			Find the numbers of MATLAB version 4 and 5 users	Bůcha	

No	Step	Stage	Problem - Task	Responsible	Deadline
35			Construct prototype	Bůcha	
36			Specify requirements on documentation of experiments with DESIGNER and ABET97	Nedoma	

3 USERS

There are *four types of users*: The user of the first type, **Ordinary User**, should be a professional worker in the field of control engineering. He *need not have knowledge of modern approaches* to automatic control. He should be able to program in MATLAB. He can be a member of AS Department, also. The user of the second type, **Expert User**, should be *an expert on modern approaches* to automatic control. He can be a member of AS Department, also and he can directly use the ABET functions. Of course, there will be a continuo of other users between these two extremes. The third User is **Administrator**. His main responsibility is to distribute the DESIGNER to the Users. The last User is **Developer**. His main responsibility is to develop the DESIGNER. Individual Developers are members of **DESIGNER team**.

At the beginning of the Project, the attention will be concentrated on the Ordinary User. To *characterize the Ordinary User* little bit more accurately, we compiled the two lists of terms: in the first list there are terms which needn't be known by the Ordinary User (so much), in the second list terms that should be known (little bit more). The boundary lead between these lists is not meant as a sharp one, but as a gradual one. It means that the terms are ordered from "not needed" to "needed" and the boundary is somehow fuzzy or statistical.

Not needed: Steps of design (Data Handling, Prior Knowledge Processing, Forgetting Estimation, Structure Estimation, Parameter Estimation, Control Design and Control Validation), LQ synthesis, penalization, Q_u, Q_y, Bayesian approach, Probability (theory).

Needed:

Basic terms (see Chapter 2.1), ability to program easier tasks in MATLAB.

Explicitly specified potential users are listed in Table 7:

Table 7: Potential Users

No	Name	Contact	Orientation
1	Dr. Ing. Pavel Ettler	CompuReg, Plzeň, Ettler@CompuReg.cz	user oriented on rolling mills with experience with some AS tools; he used structure estimation and a simple adaptive controller
2	Ing. Jaromír Fessl	Honeywell, Jaromír.Fessl@czech-republic.Honeywell.com, tel.: 61 12 23 72	he is an interested user with some experience with adaptive algorithms
3	Ing. Antonín Lízer	tel. home : 0411-504 012	He applied adaptive control in paper making industry

No	Name	Contact	Orientation
1	Dr. Ing. Pavel Ettler	CompuReg, Plzeň, Ettler@CompuReg.cz	user oriented on rolling mills with experience with some AS tools; he used structure estimation and a simple adaptive controller
4	Dr. Rudolf Kulhavý	Honeywell	he didn't agree with the proposed approach, but changed his mind
5	Ing. František Dušek	Fakulta chemicko technologická, Pardubice	
6	Ing. Hofreiter	FS ČVUT, hofreite@fsid.cvut.cz	recommended the other
7	Ing.Snajdr	Tepelná technika Praha, Dělnická 27, tel: 66710221	
8	Ing.Tmej	Vertex Litomyšl, tel: 0464-651111 or 0464-651371	
9	Doc. Jakeš	FCHT Praha	recommended the other:
10	Ing. Bronec	Fischer Rose Mode, V olšinách 75, Praha 10, tel.: 81 00 26 66	
11	Firma Sidat	Jinonická 80, Praha 5, tel.: 57 21 52 86	
12	Ing. Michal	Honeywell	
13	Doležal	Honeywell	
14	Firma Omron	Praha Vinohrady	

4 GOAL AND REQUIREMENTS

The requirements are listed in the following table. There are two parts of requirements: the first part was generated during the analysis of current state of the problem, the second part was generated during our discussions with potential users. In requirements we also enlist the current, sometimes already boiled ideas how to develop the System, further.

There are requirements that are consequences of other requirements; they are marked in the table column Source, e.g. R45, i.e. see requirement no 45. The requirements' list has not been closed. It will be systematically extended in the following Project stages. It will be checked, of course! It is impossible to accept all requirements. Mostly, we accepted users' requirements. Those not accepted now are mentioned in the table column Comment, together with a corresponding comment. The requirements are classified according to theirs priorities. It would be suitable to judge them more precisely, e.g. to weight them to be able to compare quantitatively design variants.

Table 8: Requirements

No	Name	Description	Priority	Source	Comment
1	Goal	The goal is to improve the accessibility and usage of algorithmic and software tools for systematic probabilistic design of adaptive systems by extending computerized support (CAD).		AS	

No	Name	Description	Priority	Source	Comment
2		Generally, the system like the proposed one is very much needed. (If it worked it would be real success.)		Kulhavý, Ettler	
3		**Kind and Extent of Control**			
4	LQG adaptive controllers	DESIGNER 2000 should be a tool for the design of LQG adaptive controllers.	1	AS	It is expected that the extension to multivariable case will be straightforward for multiple inputs and possible for multiple outputs.
5	MIMO	MIMO		AS	
6	Complexity and uncertainty	For the User, they are the most interesting cases with high complexity and uncertainty. The design problem "30 x 40" is not adjustable "by hand".		Kulhavý	
7	Asynchronous, with variable delay	Their processes are asynchronous, with variable transport delay, linearizeable.		Ettler	[Kárný:] Principally it is not a problem; it is necessary to find whether Bohm has prepared the corresponding tools.
8	Reality	It is necessary to consider that the reality is correlated (not uncorrelated), colored (not white), nonlinear and sometimes even non-stochastic (frozen sensors). Penalization is sometimes step-wise and not smooth (not quadratic) function.		Kulhavý	?
9	Design variants	There should be more design variants evaluated at the same time.		AS	
10		**Organizational**			
11	All available means	All available means should be used to succeed		Bůcha	partially accepted: see some comments in this column
12	Range	It is necessary to specify precisely the range of solved control problems, i.e. the range of DESIGNER 2000 application.		Kulhavý (Bohm)	
13	Other conditions	To expect an improvement from the application of adaptive controller only is not realistic, as there should be improved many other conditions, e.g. noise level		Ettler	[Kárný:] There is no complaining against data with little noise.
14	Existing software	In the first stage there should be used the existing software and should not imply (nearly) any changes.		AS	

No	Name	Description	Priority	Source	Comment
15	First stage	In the first stage, the loop from the user will be supported mainly after all seven design steps.		Kárný	Reason: In the corresponding project time, there will not be enough time and knowledge to make a small loop after each step.
16	Restricted filtering	In the first stage there will be restricted filtering.		Kárný	So called Hebký filter
17	Small loop	It should be possible to make small loop after Structure Estimation step. There is knowledge.		Nedoma	
18	Updating	There should be an updating operation so that the DESIGNER 2000 should be updated also after it has been launched into the routine operation in order to keep pace with the development of tools in the AS department.		Bůcha	
19	Quickly	The System should be ready quickly.	1	Fessl, Kárný	
20	Tuning	It should be useful to support the tuning of control in the course of production.		Kulhavý	
21	Collaborating Users	Some Users will have an important role in the Project: user-friendly language should be designed. Collaborating Users should provide a feedback concerning the suitability of that language.		Jakeš	
22	Up-to-date.	The System should be up-to-date.		R49	
23		**DATA CONTENT REQUIREMENTS**			
24	Knowledge	It is very much necessary to use knowledge to challenge the complexity.		Kulhavý	The tools based solely on theoretical approaches are not sufficient.
25		**FUNCTIONAL REQUIREMENTS**			
26	Configurators	The DESIGNER 2000 should be comparable to existing so called Configurators.		Kulhavý	
27	Similar product	Honeywell has something similar to the DESIGNER 2000. And not only Honeywell, each better firm has similar system.		Fessl	[Kárný:] But, they are not supporting adaptive controllers.
28	Ph.D.	(In the US conditions) the (Ph.D.) designer must not be a part of the design (it is too expensive).	1	Kulhavý	[Kárný:] even in ČR
29	Advanced user	The System should be semi-open to the advanced user.		Fessl	[Kárný:] to a carefully selected extent

No	Name	Description	Priority	Source	Comment
30		**Data Collection**			
31	Connect to existing systems	The DESIGNER 2000 should be able to be connected to existing systems.		Kárný	
32	Description of reality	The user should have the opportunity to import the description of physical and/or chemical reality into the design (so called basic principles).	1	Kulhavý	
33	Data measurements	The System should cover the problematic of data measurements, e.g. drivers and cards for various industrial systems, communication lines RS 232, RS 486, …		Ettler	not accepted: [Kárný:] It isn't technically and economically possible to support data measuring by us
34	Integrated Data handling	Data handling should be integrated into DESIGNER 2000. It is not acceptable for user to use another tool for data handling. This should support the import from standard databases and selection data from large amount of data. For example, Rolling Mills Frýdek-Místek daily collect many 10's MB of data in MS Access, they collect approx. 1000 signals, 30 are signals interesting and 10 signals are used for control; there are different levels of sampling: 1/s is standard, 10/s is for special events, 1/10ms was used during implementation.		Ettler, Fessl	
35		**Data Processing**			
36	Design off-line	Design mainly off-line: The time of Controlled Process, when it is out off production is critical (it is usually very expensive). The design should spent maximally 6 hours of such time. It is advisable to make as much as possible of the design during work time of it.	1	Kulhavý	[Kárný:] It is also recommendable from the point of view of heavy computation, often required.
37	Data size	It is possible to expect as common data in size 100,000 records.		Kulhavý	[Kárný:] but also in size of 200 records
38	Check the range	Check the range of DESIGNER applicability with the User.		Kulhavý	
39	Criteria Setting	It should be useful to support somehow considerably the Criteria Setting.	1	Kulhavý	AS is working on it, but the extent of attention is partially restricted
40	Knowledge contradictory	Knowledge can be contradictory.		AS	

No	Name	Description	Priority	Source	Comment
41	Controller in C++	The result of the design, Controller in C++ code should have the following properties of its source code: encapsulated, reentrant, real-time processing; it should be changeable.		Ettler	
42	Serendipity	It would be nice to support such things like serendipity effects, i.e. for example archiving fortunately good solutions, penalization matrices.		Kulhavý	
43	PLC	It should run as PLC. They are used everywhere and it will be hard to persuade a customer to change its equipment.		Ettler	[Kadlec:] it is broadly used, but we don't know it too much; It applies to controller not to the DESIGNER
44	Normalization	Normalization is necessary. But, it should be done automatically and should be hidden from the User. For the external User, the System should work in physical units. Similarly, all preprocessing should be hidden from the User.		Ettler, Fessl	
45	More measured noises	The System should be able to work with more measured noises. The System should be able to say, “I don’t know”. And, not only at the end in the course of the design.		Ettler	[Kárný:] B. has something
46	Backtrack	If the result of the design is not good, the System should be able to backtrack and find where the problem began.		Ettler	
47		**Data Correctness and Up-to-Date-ness**			
48		**User Interface**			
49	Maximally simplified	The user's utilization of DESIGNER 2000 should be maximally simplified.		AS	
50	Native language	Use a user’s native language.		R49	
51	Language understandable	The language should be much more understandable to the user then was the language of the Designer. The user prefers data-point-of-view to parameters-point-of-view, prefers graphical language to symbolic one.	1	Kulhavý	
52	Bother about information	Bother precisely about the sequence of information required from the user, e.g. whether and when it will be used in the design.		Bůcha	not accepted or partially accepted
53	GUI	The System should use GUI.		Fessl	
54	Forms	User interface should support forms, dialog, and defaults.		Ettler	
55	Document	A very good documentation is the must.		R49	

No	Name	Description	Priority	Source	Comment
	ation				
56	Rules for writing	There should be specified and obeyed rules for writing documentation.		R49	
57	Help	Hypertext type of on-line help.		R49	
58	Index	Index in off-line documentation (on paper).		R49	
59	Define terms	Define every term just once.		R49	
60	Synonyms	Do not use synonyms, but, collect synonyms and refer by them to basic terms.		R49	
61	User's comfort	User's comfort.		R49	
62	Undo	Undo.		R49	
63	GUI	There should be a very good GUI.		R49	
64	Familiar	Familiar (use of good metaphors [GUI 2-7]).		R49	
65	Protocol	Protocol		R49	
66	Occasional user	It will be nice to think about occasional user. Such user does not remember how to use the System and this situation should be supported. The set of used terms must be stable. The User should be informed about "what" and "where" to specify. There should be a semi-detailed documentation.		Fessl	
67		**Performance**			
68	Controller speed	The controller should work on speed of ms (for rolling mills). It did not work in past, but they are starting to use industrial PC and the speed of these PC is increasing.		Ettler	
69		**Reliability**			
70	Quality of software	Quality of software should be very good.		R49	
71	Standards	There will be defined standards concerning quality of software.		R70	
72	Walkthrough	Walkthrough.		R70	
73	ISO 9000	Consider ISO 9000.	1	R70	
74	CASE	Consider use of CASE, testing and versioning tools.		R70	
75	Robustness	The System should be robust.		Fessl	
76		**Personal**			
77		**Economical and Technical Support**			

No	Name	Description	Priority	Source	Comment
78	Show benefits	It is necessary precisely show benefits of the DESIGNER 2000, i.e., what money it brings to the user.		Kulhavý	Not accepted or partially accepted; we show benefits, but not up to the level of financial benefits. We have necessary knowledge. The User should do it himself.
79	Show real examples	It is not sufficient to show "one" application to the user. It is necessary to show real examples.		Kulhavý	
80	Performance contracting	The "performance contracting" is interesting for the user. It means, it is worth to make the "price" of the DESIGNER 2000 dependent on benefits, which it really brings to the User.	1	Kulhavý	
81	MATLAB	It is expensive to use MATLAB. It is very important to the User.		Ettler	Not accepted. [Kárný:] To use MATLAB is a technically justified necessity.
82	Ready products	The Users want ready products.		Jakeš	partially accepted
83	Firm management	To collaborate with User it is necessary to persuade his firm management.		Jakeš	
84		**Safety**			
85	Not destroy	The controller must not destroy the controlled process. The design should not be done on-line; there is a danger of the first step. It must be carefully tested off-line. There must not be a bigger change when going from manual to automatic control. Automatic control must start from the level of manual control. To secure all these things they are checking "force", "limits" etc, they are using many data.		Ettler	

5 PROPOSAL

5.1 DESIGNER range

The **range of solved control problems**[12], i.e. the range of DESIGNER reasonable application

[12] The control problem should satisfy this range, it is a **precondition** of the System's **applicability** and it should be checked with the User (see Chapter.5.3).

is:

1) DESIGNER is *to be used mainly* when the classical *solution has reached its limits*. It means, use DESIGNER when the control problem is currently being solved by set of mutually independent control loops (or when it can be solved in such a way), e.g. there is a similar system or there is a corresponding project, but this solution has reached its limits because it is a too complex problem
2) DESIGNER is *to be used also* when some loops are more complex, e.g. the solved *control problem has long delays or variable delays, there is feedforward, or need for multivariate solution*
3) DESIGNER is *to be used only* when the solved *control problem can be linearized*, e.g. the user knows that it is not necessary to add nonlinear elements to solve the control problem, and when the (economical) importance of the problem is not high, e.g. it means that it is not worth of hiring a Ph.D. level professional to build and/or operate a specific nonlinear model
4) DESIGNER is *not to be used,* when the solved *control problem is a simple one*, e.g. when PID control is sufficient.

There are further preconditions of the use, but they are on a different level, they cannot be checked in advance:

1) Good result could be obtained if the data are measurements of *sufficiently excited system* and/or there exists a *sufficient prior knowledge*. But, this can be recognized only after the most of design is performed, sometimes
2) The design *goal* should be at least approximately describable by a *quadratic loss*.

5.2 Architecture of the system

The DESIGNER has the following **main architectural features**:

1) OO CASE as a nucleus of the project,
2) knowledge base as a nucleus of the DESIGNER control,
3) efficient users' communication,
4) use of prior knowledge,
5) extension of the existing algorithmic and software tools if it is necessary to ensure completeness of the system.

1) The **nucleus** is needed to stabilize and standardize somewhere the project terminology, concepts and approaches. The **CASE** is needed to provide for project methods, documentation, dynamics, reliability, implementation support and user involvement. The CASE changes the character of the project from the currently prevailing bottom-up to top-down, too. The **OO CASE** is needed to provide for the solution of project complexity and for developer cooperation via UML. All main features are interdependent. The object-orientation and CASE-orientation influences importantly all the other architectural features. This is why we put this feature on the top of main feature list.

The orientation on OO CASE is extended into a more general use of **CAD** tools to support the project, i.e. use e.g. GUI (Graphical User Interface), MATLAB, expert system shell, testing tools, to further support high quality, correctness, requirements attainment, dynamics of development,

easy future development and administration of the project and product.

Object-oriented CASE tool, as a main Project tool, should support the following functions: object-oriented analysis and design in the full range of standard tools, UML to provide for exchange of project information and cooperation, production of documentation and Help, single user environment, reengineering, testing, version control.[13]DESIGNER environment

2) **KB** is needed to allow using *knowledge about the existing tools and about design process* to integrate these tools into one comprehensible and reliable system. Knowledge is spread in the documentation, examples and helps and among expert designers in the Department. There is general and specific **knowledge** used by the DESIGNER. This more *general* and/or more *stable* one is held in KB, that more *specific* and/or more *variable* one is held in DB, as DB can be more easily actualized. There is technical and problem-oriented knowledge. The **technical knowledge** specifies how to use ABET functions and objects, what are relations among them, etc. It is hidden to the ordinary user and it is implemented by standard programming tools. But, this knowledge is accessible to the expert user and in this case, it is implemented in KB. The second part of knowledge, the **problem-oriented knowledge** specifies how to manage the whole design in order to reach reliably a successful solution. It has mainly a heuristic character. The part of this knowledge is already identified. The extent of the rest of this knowledge is dependent on experiments that will be done with the early versions of the system. This processing knowledge is accessible to the user. In this part of KB, there can be "contradictory" knowledge, dependent on expert who provided it.

For example, specific characteristics of m-functions like their input and output parameter names, their usage of and utilization in other functions, their hierarchical relations, their characteristics of computational complexity, characteristics of information processed by these functions like their location, extent and form, their hierarchical relations are held in DB. The general characteristics like specification of cooperation of these functions, knowledge, how to derive the overall computational complexity, are held in KB.

KB extent is not large, at present. It has about 30 rules. This is also caused by our concentration on the creation of the first prototype, where no rules are considered. The rules are reasonably complex. Examples of problem-oriented knowledge can be found in Chapter 2.4.1.

KB is **object-oriented**. The reasons are: a) There is a reasonably complex hierarchy of objects in the design, with composition and inheritance relations, with needs for encapsulation. For example, our steps and/or covering functions (see this section) are self-contained and the concept of hiding unnecessary information (see section Communication) aims at encapsulation and polymorphism, too. For example, there are send, receive, plot operations of polymorph type. b) The previous approach had some object-oriented features. There are, e.g., the following classes of objects: Project, Step, M-function, Problem, data flows, e.g. Design Specification, Control Action, User and Expert User.

3) Efficient users' **communication** is needed to decrease the complexity of usage and to provide for better users' understanding of the system. It is solved by proposing a comprehensible *communication* language, by *providing* the user with "*exactly*" as much information as he needs, i.e.

[13] From the point of view of the Project, the usage of object-oriented CASE is important. But, it is even more important from the point of view of AS Department development. It should bring the current level of IT project object-oriented methodology into the Department. The current level of IT should support the current world level of control, produced by the Department, also.

hiding as much information as possible from him, providing as much information as possible, if he needs it, providing exact and consistent information and knowing "exactly" what are his needs in various situations.

To implement these concepts, the interaction with the user, GUI, is organized a) in a *cooperation with* several potential *users*, b) in a *top-down* manner, c) with a *hypertext-oriented help,* produced directly from the project CASE depository, d) with a *good documentation,* also produced directly from the project depository, e) with access to the system via *Internet*.

Interactive conversations begin like in the following style: "Do you like to solve control problem?" "Well, it is nice and wise of you to chose the DESIGNER 2000." "To begin with it, we are to

1) check the applicability of the system to your problem,
2) if you feel like, tell something about you and
3) specify your problem.

Let us start by checking whether DESIGNER 2000 suits to your problem. …"

Basic elements of communication language and usage of prior knowledge are given in structured description of Interaction (see Figure 7: Class Diagram of Interface structure, Table 3: Data Specifications, Table 11: Proposed Data Structures). In range of the concept of using knowledge of controlled system it is necessary to stimulate the User to provide as much as possible of his Prior Knowledge and restrict its incompleteness in such a way, and to precise as much as possible his goals.

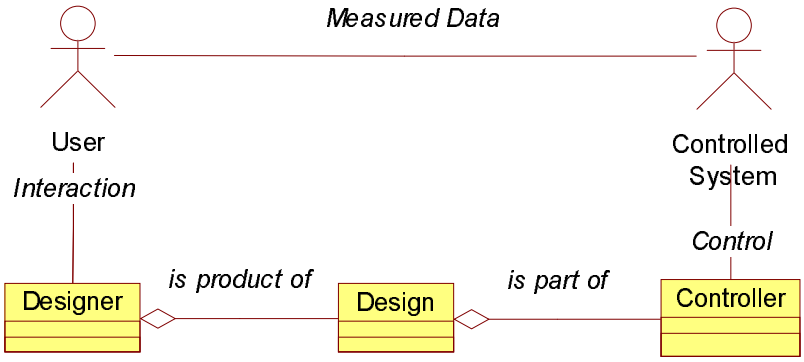

Figure 3: Main Classes diagram

The DESIGNER's environment is shown in Figure 3. The main classes of this environment are Designer, User and Controlled System. In this Figure there are also classes Design and Controller that are parts of the Designer class. They are considered in this Figure because they have a special role and relation to the environment. All these classes and their relevant relations are shortly described in Table 9. (There are classes corresponding to association relations, not to generalization relations. An example is Interaction. For simplicity reasons, they are shown neither in this Figure nor in other similar Figures. They are shown in their own Figure 7: Class Diagram of Interface structure.) Controller can run either off-line or on-line. Figure 3 is the same as Figure 2. We repeat it here for reader's convenience. The difference between the current state of the System and the proposed DESIGNER is in the *contents, meanings and implementations* of classes (entities) and their relations. (Please, compare Table 1: Main Entities of the Current Design and Table 9.)

Table 9: Main Classes and Relations of the Proposed System

No	Class / Relation	Type of OO Entity	Type of Entity	Description
1	**Designer**	class	main process	It is a result of the Project, a set of functions, which allows the User to design a Controller. For the more detailed description, see Figure 4-Figure 7.
2	**User**	actor class	external entity	It is Ordinary and Expert User in sense of Chapter 3.
3	**Controlled System**	actor class	external entity	See Chapter 2.1.
4	**Measured Data**	association relation	data flow	Data measured on Controlled System by User.
5	**Interaction**	association relation	data flow	It is all bi-directional interaction between User and Designer. It contains both Design Specification and Design Response. For more detailed description see description of associated class.
9	**Design**	class	data flow and data set	It is a result of the Designer's activity, the part of Designer.
7	**Controller**	class	data flow and data set	It is a result of the Designer's activity, the part of Design.
8	**Control**	association relation	data flow	It is all bi-directional interaction between Controlled System and Controller. For more detailed description see description of associated class.

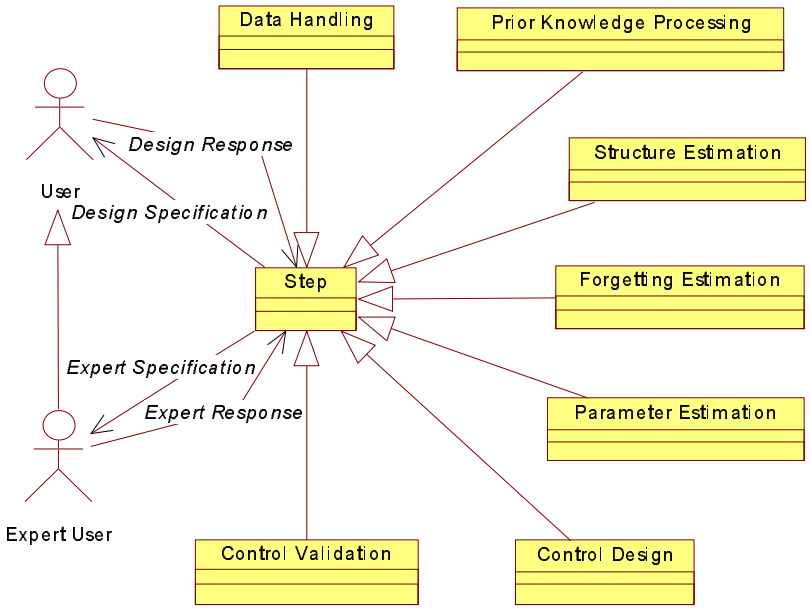

Figure 4: Class diagram of steps' Common Properties

Our task, the DESIGNER 2000 project is solved in range of the grant "Adaptive Systems: Theory, algorithms and software practice" [9]. In the same grant, there are other tasks. One of them, namely, "Support of an efficient computer aided *education* in the field" is very closely related to our Project, especially in the proposed framework. These tasks should share common knowledge, documentation, they could share approach and tools like Help, Examples, Internet, experimental environment etc. This

between these steps, User and Controlled system.

In Figure 6, there is shown the relation of Step to other more detailed functions (classes) and/or relations of these classes to each other, i.e. /esign Control, //, //, /UI, /elp, /utorial, //amples, /ocumentation, Chec/ of /pplicability, Step /b/ect, //// /b/ect and Comple/ity. /he new classes are shortly described in /able //.

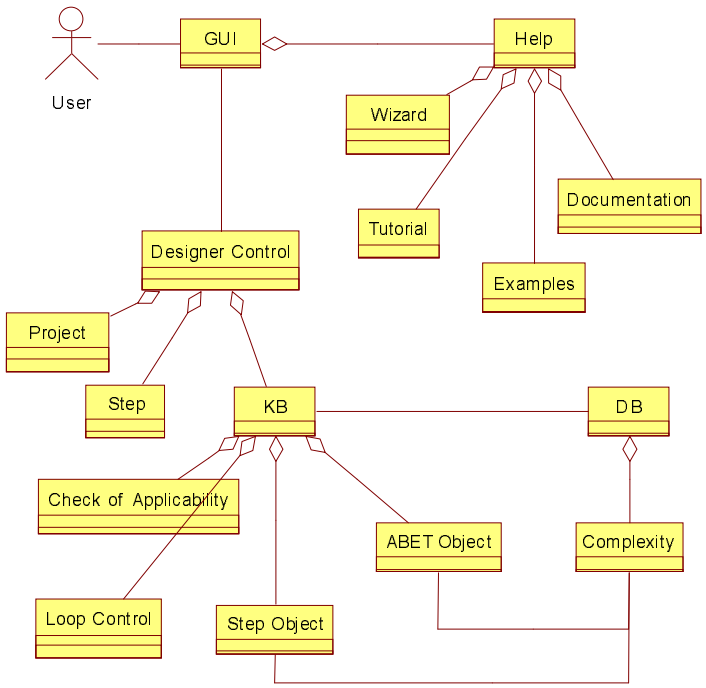

Figure 6: Detailed Class Diagram

Table 10: Details of Detailed Classes

No	Class	Description
1	**ABET Object**	It is a part of KB describing the characteristics and behaviors of ABET Objects.
2	**Complexity**	It is a part of DB describing the complexity characteristics of various processing parts.
3	**DB**	It processes and provides all internal data necessary for the overall control of Design.
4	**Design Control**	It provides the overall control of Design process.
5	**Documentation**	In range of Help, it provides a documentation to guide the User through the work with DESIGNER. OO CASE produces it.
6	**Examples**	In connection with Help, it provides examples illustrating the work with DESIGNER.
7	**GUI**	It provides all processes necessary for implementation of graphical user interface.
8	**Help**	It provides any kind of help. OO CASE supports its production.

No	Class	Description
9	**Check of Applicability**	It is a part of KB oriented on checking applicability of DESIGNER.
10	**KB**	It processes and provides all internal knowledge necessary for the overall control of Design.
11	**Step Object**	It is a part of KB describing the characteristics and behaviors of steps.
12	**Tutorial**	In range of Help, it provides tutorial functions to introduce new user into work with DESIGNER.

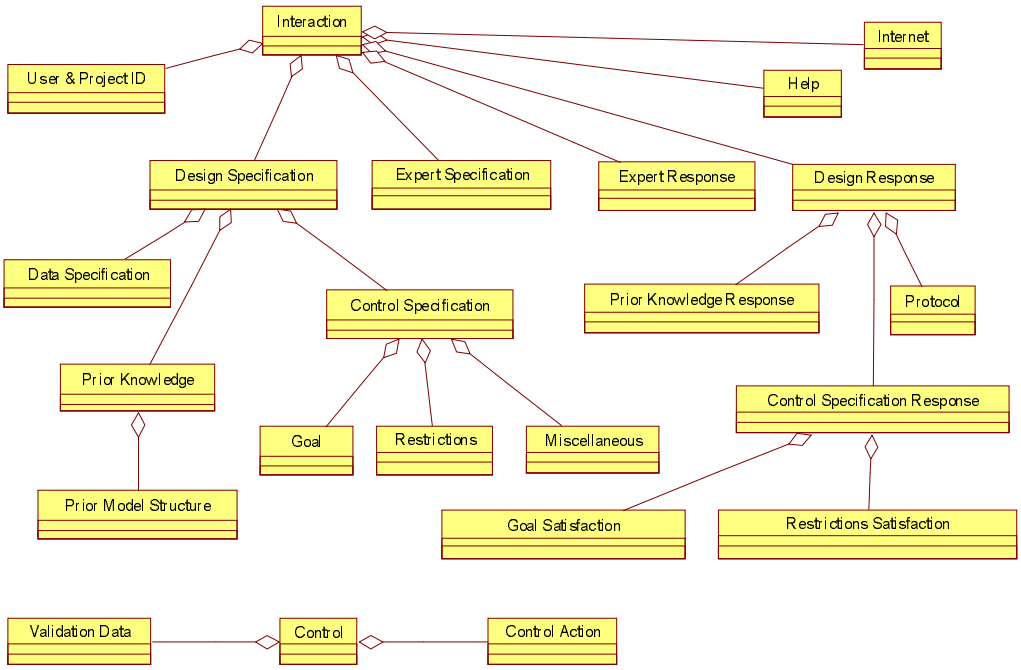

Figure 7: Class Diagram of Interface structure

5.4 Inputs and outputs

In Figure 7, there are shown the basic levels of input/output structure. More detailed description is in Table 11: Proposed Data Structures. The list and partial description of data items occurring in these structures is in Table 3: Data Specifications.

Table 11: Proposed Data Structures

No	Entity	Structure	Description
1	**Measured Data**		
2	**Interaction**	1) Design Specification, 2) Design Response, 3) Expert Specification, 4) Expert Response	
3	**Design**	1) Controller, 2) Design Response, 3) Expert Response	

No	Entity	Structure	Description
4	**Controller**		
5	**Control**	1) Validation Data, 2) Control Action	
6	**Design Specification**	1) Control Specification, 2) Data Specification, 3) Prior Knowledge, 4) Prior Model Structure	It is a product of User's interactive activity, which is using the Designer's, tools, i.e. a program (and data included in this program) for Designer to produce Control Action.
7	**Control Specification**	1) Goal, 2) Restrictions, 3) Miscellaneous	
8	**Design Response**	1) Prior Knowledge Response, 2) Control Specification Response, 3) Protocol	It is all information provided to User during creation of the Design.
9	**Prior Knowledge Response**	the same as Prior Knowledge	
10	**Control Specification Response**	1) Goal Satisfaction, 2) Restrictions Satisfaction, 3) Changeable Specification,	
11	**Goal**	*1) Type of Problem, [for regulation problem:] 2) Limits on Outputs, 3) Limits on Outputs Differences, 4) Limits on Outputs Expectations, [for servo problem:] 5) Set-Point, 6) Quality of Transient Response*	
12	**Restrictions**	1) Limits on Inputs, 2) Limits on Inputs Differences, 3) Limits on Inputs Expectations	
13	**Data Specification**	1) Data Sources, 2) Data Sets, 3) Data, 4) Output Signals, 5) Auxiliary Output Signals, 6) Input Signals, 7) Signal Labels, [for servo problem:] 8) Signal with Set-Point, 9) Signal with Input Reference	
14	**Prior Knowledge**	1) Range of Signal, 2) Range of Signal of Differences, 3) Static Gain, 4) Steady State Signal, 5) Cut Off Frequency, 6) Parameter Range, 7) Frequency Characteristics, 8) Dominant Time Constant, 9) Functional Dependency	What the User knows about her/his control problem and what AS can process.
15	**Prior Model Structure**		
16	**Expert Specification**	*1) Arguments, 2) Function, 3) Table, 4) Means, 5) Initial Value, 6) Model, 7) Covariance Matrix, 8) Distribution, 9) Models, 10) Signal Sources, 11) Any Task, 12) Delay, 13) Filtering, 14) Incremental Mode, 15) Limits, 16) Sampling Frequency, 17) Scaling, 18) Maximum delay, 19) Channel, 20) Channels, 21) Forgetting Type, 22) Workspace, 23) Object Name, 24) Mode of Synthesis, 25) Criterion Structure*	
17	**Miscellaneous**	1) Allowed Computational Time, 2) Parameters Available to User, 3) Expert, 4) Min Control Period Factor, 5) Max Control Period Factor	
18	**Goal Satisfaction**	1) Outputs Satisfaction, 2) Outputs Differences Satisfaction	
19	**Restrictions Satisfaction**	1) Inputs Satisfaction, 2) Inputs Differences Satisfaction	Levels of satisfaction

No	Entity	Structure	Description
20	**Expert Response**	*1) Control Law, 2) Riccati Matrix*	Levels of satisfaction
21	**Quality of Transient Response**	*1) Limits on Overshoot, 2) Limits on Steady State Values*	Quality of response on transient response
22	**Data Sets**	1) Data Set Name, Data Set Structure, 3) Precision	Part of Data Specification.
23	**Validation Data**		It is a stream of values of Controlled System signals used for the validation of the design. These values may be simulated or obtained from the real Control System. In case of simulated values, the User is expected to provide them, e.g. With the help of some model.
24	**Control Action**		It is a behavior signal, which is the input of Controlled System. It is not a routine control, but, more or less, the validation of control.
25	**Handled Data**	1) Data, 2) Output Signals, 3) Auxiliary Output Signals, 4) Input Signals, 5) Signal Labels, [for servo problem:] 6) Signal with Set-Point, 7) Signal with Input Reference	

5.5 Data model

In our object-oriented approach, the data model is spread in the (object-oriented) model. It is described in whole Chapter 5.

5.6 Necessary hardware and software

We consider hardware and software properties from the points of view of the Users.

For *Ordinary or Expert User* we expect a standard equipment, i.e. (plain) PC, MS Windows or HP-Unix workstation, and **special equipment**, i.e. run-time of expert system and MATLAB. PC should be as powerful as demanding designs will be done. The special equipment would cost the User approximately 100.000 Kč.

For *Developer*, we propose on top of previous requirements OO CASE, expert system, e.g. Nexpert Object, C++, MS Office. To connect MATLAB and Nexpert Object API (Application Programming Interface) should be used.

5.6.1 OO CASE and UML

The utilization of OO CASE based on UML is justified by *advantages* it brings to the Project:

1. It will significantly help to *comprehend* a model of the (complex) System. It means, it will support higher design quality and efficiency, and lower time and cost of system design.
2. In will bring a collection of *best engineering practices* that have proven successful on modeling of large and complex systems. For example, it will promote a development process that is use-case driven, architecture centric, and iterative and incremental.
3. It will provide *standardized, graphical, used-oriented language for description of* the Project.
4. It will support *communication* both in the DESIGNER team and outside, e.g. our international communication.
5. It will support *extensibility* of the designed System.
6. It will support the use of *contemporary concepts*, like ActiveX, component-based and portability approaches.
7. It brings *perspective* tool into the Project.

There are obviously *disadvantages,* also:

It is necessary to *invest* money to buy a tool and work to learn a tool.

5.6.2 Rational Rose

In the following table, there are compared available OO CASE tools. It can be seen that the most suitable is **Rational Rose** (RR). This is supported by the standpoint of Prof. Cendelín [3]. Future attention will be directed to this product, mainly.

RR is not an ideal tool. Today it is not stabilized, i.e.

1. it does not support all 100% of UML (no Object Diagram and no Activity Diagram),
2. it is an obvious compromise among many leading software firms interests,
3. it does not include very many consistency controls, now.

These are not serious problems for DESIGNER, they address more Prof. Cendelín's orientation on process control.

Nevertheless, RR is still the most advanced available tool. If we had oriented on the pure project-oriented solution, a classical structure-oriented CASE tool would have satisfied out requirements. But from perspective viewpoints, OO CASE tool is more appropriate. Due to these reasons, RR has been chosen as the best solution (which can still be revised).

5.6.3 MATLAB

MATLAB must be an algorithmic base of DESIGNER. Questionable is which version of MATLAB. The candidate versions are versions 4, 5.1, 5.2.

Advantages of version 4 are:

1. better price
2. errors in version 5.1 and 5.2
3. some users still working in version 4, even in AS Department.

Advantages of version 5.1, especially 5.2, are:

1. speed of development of GUI (it seems that version 5.2 is computationally twice as quick as version 5.1)
2. possibility to use new GUI approaches, like ActiveX
3. and this enable much more sophisticated user-oriented GUI,

4. there are more users of these versions.

Table 12: Evaluation of Unofficial CASE Proposals

No	Supplier	CASE name	Version	School rate	Price	Properties											
						OO A&D (UML)	Requirements Traceability	CORBA 2 IDL	Documentation Generation	MS Word	C++ Code Generation	Round Trip Engineering	Version Control	Hypertext Help	Consistence state and class diag.	Life cycle of objects	Reference
1	AIT	SELECT Enterprise	5.1	0,5	56 000	1	1	1	1	1	1	1					
2	FBL Group	Objectif	2.0b	0,4	62 000	1	?	?	1?	1?	1	1	1	0	1	1	1
3	KOMIX	SELECT Enterprise	5.1	0,5	56 000	1	1	1	1	1	1	1					
4	LBMS	SELECT Enterprise	5.1	0,5	56 000	1	1	1	1	1	1	1					
5	OKSystem	Rational Rose/C++ for Windows	4.0	0,2	20 300	1	?	?	?	1	1	1	?	1			

The existing texts, e.g. documentation, are to be used *to create Documentation and/or Help* for the User. The existing texts are mostly in Latex format, Help should be in hypertext form. Ing. Haindl recommends to use LaTeX2HTML Translator by Nikos Drakos, University of Leeds [http://cbl.leeds.ac.uk/nikos/tex2html/doc/manual]. The Translator does not translate everything, e.g. mathematics.

Another possibility is to use the product distributed by UNICORN that directly translates Word document to Help.

5.7 Proposed criterion and method for the evaluation of variants

The **criterion** is: The minimization of estimated Project cost. The method is estimate the costs of variants and compare them.

5.8 Proposed variants

We have identified the following possible (partial) variants:

1) variant with or without CASE and which CASE

2) variant with or without OO CASE course

3) variant with or without a test tool

4) variants of GUI, e.g. MATLAB, Simulink, another

5) variants of expert system shell

6) variant to base Documentation on existing Latex documentation or on documentation produced by OO CASE.

1) *OO CASE*: The price of OO CASE Rational Rose is 21000 Kč. If OO CASE is not used, we guess that the OO Analysis itself, Design and Implementation substages and stage will take 90% more time then they would take with CASE. We consider this index quite reasonable. Our guess is based on experience with rough analysis made here in the course of this study. Considering the price of one scientist day 4310 Kč, considering guesses from Table 13: Schedule of the Project, we get the cost of not using OO CASE 2.39 MKč. We recommend using OO CASE.[14] It would be reasonable to consider a more expensive OO CASE, too. This is let to the next stage.

2) **OO CASE course**: Self-study is expensive. If we consider the course price 5000 Kč, course duration 2 days, course efficiency 100% (it means that to learn the topic on the same level of understanding without the course one needs 100% more time), then the price of self-study is 8600 Kč. We recommend taking the course. This conclusion can obviously be generalized for the cases of other courses, too, e.g. for the course of test tool.

3) **Test tool**: The price of test tool SQA Suite (Node-locked License, Robot, Manager) is 29000 Kč. If test tool is not used, we guess that the Implementation stage will take 90% more time

[14] We are not taking into account advantages mentioned in note 13. They are harder to calculate.

then they would take with test tool. Considering the price of one scientist day, considering guesses from Table 13: Schedule of the Project, we get the cost of not using test tool 1.19 MKč. We recommend using test tool.

4) **GUI**: There is a possibility to use MATLAB GUI. It is a simple one, e.g. it allows to place MATLAB plots, to use pushbuttons, sliders, pop-up menus, pushdown menus, list-boxes, radio-buttons, check-buttons, edited and fixed texts. Only the left-mouse-button is supported, no double clicking etc. For the first prototype it would be suitable, in A&D stage the more detailed analysis of GUI possibilities will be done.

5) **Expert system shell**: In the mean time the available Nexpert Object will be used. This system is not further supported and upgrade is not possible. In A&D stage the more detailed analysis of expert system shell possibilities will be done.

6) *Base of Documentation* should be decided in A&D stage after a more detailed analysis will be done.

There are no special organizational, personal consequences of these variants. The proposed ones have technological and economical advantages.

6 SCHEDULE OF THE PROJECT CONTINUATION

6.1 Stages of Project

We divide the Project into four **stages**:

1. **Feasibility Study** that aim is to create the first rough closed concept of the whole System, about its goal, main user requirements, its main functions, consequences, cost, necessary resources, and overall feasibility
2. **Analysis and design** (A&D) that aim is to create the precise design of the System that can be used to implement it
3. **Implementation** that aim is to create the working System that implements the design in a given implementation environment; the part of the implementation is the testing of System's correct behavior, whether the System corresponds to the design and user requirements
4. **Operation** that objective is to provide for the performance, maintenance and further development of the System; the System should be developed in accordance with the development of user needs and with changing environment. A feedback from users will be necessary. This is the reason, why the "**marketing policy**" will be very mild to at least first users. The product will be offered to users "nearly" free off charge.

Data Handling should be covered according to the corresponding specification.

The overall development must be **incremental**. This means that the **parts** of the System could be solved one after another. It is necessary to go through all stages for each part. In the current state, we can identify the first part, i.e. the prototype, only.

6.1.1 Prototype

The first prototype part that objective is to connect and run all steps from the beginning to the end

in *the simplest possible version*. It should include knowledge concerning the main loop, only. It should use the available covering functions. The prototype should include a session protocol, where all inputs (including defaults) and outputs main data structures shall be logged.

It is *not necessary* to cover for example the following features:

1. include *knowledge* concerning *all loops*, especially small ones, build regular KB
2. multidimensionality
3. solve specialties like Help, Documentation, Protocol, Internet
4. solve the *complexity* problems both for off-line and on-line solutions
5. solve Parameters Available to User
6. cover Expert User "requirements"
7. build sophisticated GUI
8. base project on *OO CASE*, testing tools etc.
9. cover servo type of problem
10. external noise.

The covering functions for the first prototype are prepared with respect to these simplifications. The prototype was presented to AS and resulting suggestions were implemented.

6.1.2 Experience gained from the prototype

It is already possible to mention some experience gained directly from the prototype development:

1. It is not efficient to create even that simple prototype without previous analysis. It brings problems:

- It brings necessity to repeat some not well-designed actions
- It brings higher complexity, as some A&D is necessary and it must be done concurrently with implementation (programming)
- It increases uncertainty, as there is practically no (written) specification and as there are no criteria of successful solution.

2. The work with MATLAB is not without problems

- The description of MATLAB functions is neither really clear nor complete
- MATLAB does not have many required and useful functions
- The work with lower versions is problematic, e.g. version 5.1 is much slower then version 5.2 (selection takes 6 sec instead 2.5 sec), there are more errors in version 5.1

3. It shows that much time is spend on testing
4. It justifies our standpoint to the previous development of DESIGNER.

5. On the other hand, it showed that without making the prototype we could not have enough experience to design the whole GUI for DESIGNER.

7 ESTIMATE OF THE COST

In this Chapter we estimate the Project cost and "profit" from the DESIGNER.

It would be suitable to consider the Project's cost (time) in relation to money (time) that has been *spent during the overall previous development* of tools (theory, algorithms, software and expertise). It is hard to estimate it, as it is hard to estimate what all belongs to the previous development and as it is hard to estimate the efficiency of the development. We estimate that this time is higher then 80 and less then 400 researcher-years.

The Project cost is estimated in Table 13. The estimation is very rough, as assumptions that it is based on, are rough too. They are: a) cost of processing one project object, e.g. data item, data structure, program, requirement, is one day, b) some objects are more complex, we estimate e.g. three days for problems, c) we recognized already the most of this objects, we estimate 70%. It gets the ***overall necessary effort estimate on level of 4.3 person-years***. If we consider possible estimation error in range +200% and –66%, we get the minimal and maximal overall necessary effort estimate 1.4 and 12.9 person-years.

Table 13: Schedule of the Project[15]

No	Stage - Substage	Capacity Bu [d]		Other Capacity [d]		Deadline
		Done	Sched.	Done	Sched.	
1	Feasibility Study	130	9	36	0	
1.1	Study	90		5		30-08-98
1.1.1	Study itself	80		4		
1.1.2	Translation into Latex	10		1		
1.2	1st Prototype Substage	20	5	30	0	13-08-98
1.2.1	Covering Functions			30		30-07-98
1.2.2	GUI	20	5			13-08-98
1.2.3	Various other tasks			?	?	
1.3	FS Presentation	30	4	2	0	
1.3.1	Control '98	8	4	1		04-09-98
1.3.2	NATO SfP	5		1		
1.3.3	RIP	12				11-06-98
1.3.4	AS	2				26-01-98
1.3.5	GA ČR	3				

[15] A careful reader will notice that the capacities of stages (and substages) do not correspond to sums of capacities of corresponding substages. It is given by the construction of these numbers: They are our estimations. They do not directly correspond to sums.

No	Stage - Substage	Capacity Bu [d]		Other Capacity [d]		Deadline
2	Analysis and Design	5	254	0	245	25-08-99
2.1	Feedback from User on FS		5		3	
2.2	Analysis	5	111	0	99	
2.2.1	GUI		4			
2.2.2	OO CASE	5	16	0	8	
2.2.2.1	Books	1	2		2	
2.2.2.2	Books Study	1	6		6	
2.2.2.3	Presentation to AS		2			
2.2.2.4	Purchase of CASE	1	2			
2.2.2.4	Course		4			
2.2.3	ISO 9000		3		3	
2.2.4	OO Analysis Itself		88		88	
2.3	Design		88		88	
2.4	Prototyping	0	50	0	55	
2.4.1	KB & Design Control		4			
2.4.2	KB & GUI		4			
2.4.3	KB & DB		3			
2.4.4	Covering Functions				30	
2.4.5	ABET Development				15	
2.4.6	Documentation		2			
2.4.7	Help		2			
2.4.8	Outlook Utilization		4			
2.4.9	PID Comparison		1		10	
2.4.10	API	1	10			
2.4.11	ES acquisition	2	15			
2.4.12	CASE & MATLAB	1	5			
3	Implementation		177		177	04-05-00
	All together	135	440	36	422	04-05-00

8 ORGANIZATION OF THE PROJECT

On the Project, there are and will be participating more people. For example there are the following potential "programmers": Josef Bohm, Markéta Valečková, Hong Gao, Natalia Khailova, Petr Nedoma, Lenka Pavelková and Jiří Rojíček. There are implicit Project **standards**

described in [12].

8.1 Standards

The part of new **revised ABET97 documentation**, reused in DESIGNER, will be done in a new style as oriented on a new type of the User. It should adhere to the following standards:

- The documentation should be truly *understandable, precise and complete.*
- There should be a content, index, abbreviations and the used synonyms.
- All (not self-evident) terms should be *defined* and this definition should be graphically *underlined* (**bold**). Important information should be graphically underlined (*italic*). There should be the user-oriented data items' and functions' descriptions and the descriptions of items' *representations*. For *functions*, there should be lists of inputs and outputs, their description, short description of algorithm and reference to the theoretical background.
- Before utilized, e.g. in the prototype, a description of covering function should be presented to the DESIGNER team for discussion and approval
- There will be used the *user names* from this Study.
- All *ABET objects' items* will be described in corresponding m-functions as comments.
- An OO CASE and its depository will be a **core of the Project**.[16] The parts of previous

As we propose to buy a single user version of OO CASE[17], it is necessary to *organize the access to the repository* for all, which needs it. We start to use the OO CASE type of description already in this Study to produce a basic set of OO figures.

8.1.1 Terminology Stabilization

In the previous DESIGNER, there were not stabilized terms. It resulted in the high occurrence of synonyms. In the DESIGNER, synonyms will be suppressed. The User will have the possibility to use them, but indirectly. For example, to use them via index, where synonyms will be listed with references to standard terms. The seed of this approach to synonyms is the following table. It will be extended and used in the Index and Help.

Table 14: Synonyms

No	**Term**	Synonym**(s)**
1	Controller	Regulator
2	Criterion	Control Strategy
3	Criterion Setting	Optimality tailoring
4	Data Handling	A priori Information Processing
5	Model structure	Regressor
6	Modeled Data	Simulated Data
7	Object Member	Object attribute value; object field
8	Parameter Estimation	Parameter Identification
9	Protocol	Log
10	Signal	Channel, logical channel, logical processing channel
11	Structure Estimation	**Structure Identification**

9 CONCLUSION

1. It is ***feasible to continue*** with the Project.
2. The ***estimate of overall necessary effort is on level of 4.3 person-years***.
3. There is relatively high **uncertainty** of the estimate; i.e. the minimal and maximal overall necessary effort estimate is 1.4 and 12.9 person-years.
4. Generally, the system like the proposed one is very much needed. If it worked it would be real success.
5. The Study confirmed the Project's assumption about tools: These tools can be used to build the DESIGNER.

[17] The reason is that it is necessary to gain some experience with the tool before implementing it on a broader scale.

6. One of the important Project's assumptions was the existence of experience and knowledge about the utilization of Bayesian methods in analysis and design of controllers. This assumption was confirmed only partially. There is same knowledge that can be elucidated, but it is not clear now if any significant knowledge can be gathered and further research should be made in this direction.
7. There was an opinion that as Bayesian approach is based on the least square method, similar results can be obtained in other ways, too. That is not the case.

REFERENCES

[1] Bůcha J., Kárný M., Nedoma P., Bohm J., Rojíček J. (1998). DESIGNER 2000 Project. Pre-prints of Control 98 Conference, Swansea, submitted.

[2] Bůcha J., Kárný M., Nedoma P., Bohm J., Rojíček J. (1998). Overview of DESIGNER 2000 Project. Pre-prints of Process Control 98 Conference, ŘÍP 1998, University of Pardubice, Kouty nad Desnou, p. 65-70.

[3] Cendelín J. (1998) Object oriented CASE systems, CSKI lecture.

[4] Fowler M., Scott K. (1997) "UML Distilled", Addison-Wesley.

[5] Kárný M., Halousková A., Bohm J., Kulhavý R. and Nedoma P. (1985) "Design of linear quadratic adaptive control: Theory and algorithms for practice", Kybernetika, vol. 21, Supplement to No. 3, 4, 5, 6.

[6] Kárný M., Halousková A. (1991) "Preliminary Tuning of Selftuners", in [19].

[7] Ljung L. (1995) "System Identification Toolbox, User's Guide", The Math Works, Natick, Mass.

[8] MATLAB (1996) "Using MATLAB version 5", The Math Works, Natick, Mass.

[9] Nedoma P. (1996) "Adaptive Systems: Theory, algorithms and software practice" GA AV ČR No. 102/97/0118, 1997-1999.

[10] Nedoma P., Kárný M., Bohm J. (1996) "ABET, A Tutorial", ÚTIA, ČAV, Praha.

[11] Nedoma P., Kárný M., Bohm J. (1996) "ABET, User's Guide", ÚTIA, ČAV, Praha.

[12] Nedoma P. (1996) "ALIEN, User's Guide", ÚTIA, ČAV, Praha.

[13] Nedoma P., Kárný M., Bohm J. (1996) "DESIGNER: Preliminary Tuning Adaptive Controllers", Proc. Of the 2nd Process Control, ŘÍP 1996, University of Pardubice, Horní Bečva.

[14] Nedoma P., Kárný M., and Bohm J. (1996) "Project DESIGNER", Proc. of the spring school Adaptive and Predictive Control, Oxford University, Oxford.

[15] Nedoma P., (1998) "ABET97, Adaptive Bayesian Estimation Toolbox, A Tutorial", Technical Report, ÚTIA, AV ČR, Prague, Czech Republic.

[16] Peterka V. (1981) "Bayesian system identification", in Trends and Progress in System Identification, P. Eykhoff, Ed., pp. 239-304. Pergamon Press, Oxford.

[17] Rational Rose (1996) "UML Tutorial", Rational Rose/C++ Demo (CD ROM), Rational Software Corp., Santa Clara, CA.

[18] Standardy Státního informačního systému České republiky (1996) Úřad pro Státní informační systém, Praha.

[19] Warwick K., Kárný M., Halousková A. (Eds.) (1991) "Advanced Methods in Adaptive Control for Industrial Applications", Lecture Notes in Control and Information Sciences 158, Springer-Verlag, Berlin.

INDEX

A

a priori information processing **55**
A&D **3**
ABET **3**
adaptive controller **7**
administrator **29**
algorithms **15**
analysis and design **50**
API **3**
ARX **3**
AS **3**

B

basic steps **10**
basis for DESIGNER **8**
bold **54**

C

CAD **3**,**37**
CASE **3**,**37**
communication **38**
computational complexity **26**
computational complexity description **54**
context **8**
control **7**,**9**
control design **7**,**11**
control problem **7**
control strategy **55**
control validation **12**
controlled system **6**,**8**
controller **7**,**9**,**55**
core of the Project **54**
covering functions **8**
criterion **49**,**55**
criterion setting **55**

D

data handling **10**,**55**
DB **3**
design **4**,**9**
Designer **8**
DESIGNER **4**
DESIGNER team **29**
detrending **11**
developer **29**
disturbance **6**

E

education task **41**
expert system shell **50**
expert user **29**

F

first level of description **8**
forgetting estimation **11**

G

GA **3**
general expertise **23**
goal **4**
GUI **3**,**50**

H

hiding information **39**
historical context **7**

C

change delay **11**
Channel, **55**

I

implementation **50**
importing data **10**
incremental **50**
incremental mode **11**
input **6**
interaction **9**
intermediate steps **12**
ISO **3**
IT **3**
italic **54**

K

KB **3**,**38**
knowledge **38**

L

language symbols **7**
log **55**
logical channel, **55**
logical processing channel **55**
LQ **3**
LQG **3**

M

main architectural features **37**
many small Project's changes **9**
marketing policy **50**
measured data **8**
MIMO **3**,**6**
MISO **3**,**6**
model **6**
model structure **55**
modeled data **55**
multidimensional system **6**

N

next level of description **10**
normalization **11**
nucleus **37**

O

object attribute value **55**
object field **55**
object member **55**
objection **5**
object-oriented **38**
object-oriented CASE tool **38**
OO **3**
OO CASE **37**
OO CASE course **49**
operation **50**
optimality tailoring **55**
ordinary user **29**
outliers elimination **11**
output **6**

P

parameter estimation **11**,**55**
parameter identification **55**
parts **50**
PID **3**
PLC **3**
prefiltering **11**
previous projects **7**
prior knowledge processing **11**
problem-oriented knowledge **38**
processing **15**
Project **4**
protocol **55**
providing information **39**

Q

quality-of-programs **9**

R

range of solved control problems **36**
Rational Rose **3**,**46**
regressor **55**
regulator **55**
requirements **9**
resampling **11**
revised ABET97 documentation **54**
Riccati equation **12**
roles of the authors **5**
RR **3**,**46**

S

selecting data ranges **10**
selecting sampling period **10**
signal **6**,**55**
simulated data **55**
simulating data **10**
SISO **3**,**6**
special equipment **45**
special expertise **24**
stability of particular steps **12**
stages **50**
standards **53**
state of the documentation and tools. **9**
steps' inputs and outputs **12**
structure estimation **11**,**55**
structure identification **55**
Study **4**,**50**
substeps **10**
synonym(s) **55**

T

technical knowledge **38**
test tool **49**
tools **4**,**9**

U

UML **3**,**5**
user **8**
user's interface **15**
ÚTIA **3**