Helsinki University of Technology

Industrial Information Technology Laboratory Publications

Espoo 2005 TKK-INIT-2

BUSINESS PROCESS MODELING AND SIMULATION

Alexander Sidnev, Juha Tuominen, Boris Krassi

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY
TECHNISCHE UNIVERSITÄT HELSINKI
UNIVERSITE DE TECHNOLOGIE D’HELSINKI

Helsinki University of Technology
Industrial Information Technology Laboratory Publications
Espoo 2005 TKK-INIT-2

BUSINESS PROCESS MODELING AND SIMULATION

Alexander Sidnev, Juha Tuominen, Boris Krassi

Helsinki University of Technology
Department of Computer Science and Engineering
Industrial Information Technology Laboratory

Distribution:
Helsinki University of Technology
Industrial Information Technology Laboratory
P.O. Box 5400 TKK
FIN-02015 Espoo
Finland
Fax: +358-9-451-5351
www.init.hut.fi

ISBN: 951-22-7928-2 (printed version)
ISBN: 951-22-7929-0 (electronic version)
ISSN: 1459-6458

Otamedia Oy
Espoo 2005

HELSINKI UNIVERSITY OF TECHNOLOGY

http://www.tkk.fi/

ABSTRACT PAGE

Department/laboratory and URL/Internet address

Computer Science and Engineering

Industrial Information Technology Laboratory

www.init.hut.fi

Publisher

Otamedia Oy

Authors

Alexander Sidnev, Juha Tuominen, Boris Krassi

Title

Business Process Modeling and Simulation

Abstract

The textbook provides the essentials of the Business Process (BP) Modeling and Simulation (M&S) from the verbal BP description to the formulation of the mathematical scheme of the model and the simulation program.

Both the analytical modeling and the simulation approaches to BP M&S are considered. Special attention is given to the theoretical and practical aspects of the BP M&S. The text covers the following topics: fundamentals of the BP M&S, conceptual modeling using IDEF3 standard, cost metrics and the activity based costing, analytical modeling (queuing networks, linear and dynamic programming), simulation with GPSS, timed Petri Nets, and Crystal Ball toolkits. Case studies include BP simulations with BPwin and GPSS.

The intended readers are senior graduate students and junior postgraduate students of computer science and industrial management.

Keywords

BP modeling and simulation, linear programming, dynamic programming, timed Petri nets, queuing networks, activity based costing, GPSS

Place	Year	Number of pages	Language of publication	Language of abstract
Espoo	2005	116	English	English

ISBN (printed)	ISSN (printed)
951-22-7928-2	1459-6458
ISBN (electronic)	
951-22-7929-0	

URL (Internet address)

http://lib.tkk.fi/

LIST OF ABBREVIATIONS

ABC	Activity-based costing
BP	Business process
BPwin	BP simulation software of Computers Associates International, Inc
CEC	Current event chain
FEC	Future event chain
GPSS	General purpose simulation system
IDEF	Integrated family of integration definition methods
IDEF0	Integration definition for function modeling
IDEF3	Integration definition (IDEF) method for process description capture
M&S	Modeling and simulation
OROS	Trademark of ABC Technologies
UOB	Unit of behavior

CONTENTS

AKNOWLEDGEMENT

The textbook is based on the lecture notes of the “Business Process Modeling and Simulation” course, which was delivered by Dr. Sidnev at Helsinki University of Technology in May 2004 and September-October 2005. The guest lectures of Dr. Sidnev have been supported by the Academy of Finland.

PREFACE

Paraphrasing Robert Shannon [15], the business process (BP) modeling and simulation (M&S) is probably as much art as science. The subject of this textbook concerns the science of BP M&S, in other words, the part of the BP M&S that can be formalized.

The textbook is based on the lecture notes of "Business Process Modeling and Simulation" course, which was delivered in May 2004 and September-October 2005 for postgraduate students at Helsinki University of Technology.

The basic idea of the book is to consider the capabilities of M&S applied to the BP analysis and synthesis. Equal attention is paid to the following complementary approaches – the BP analytical modeling vs. the BP simulation – as it is expedient to use both of them together. While the analytical models allow formalizing the complicated problems of the BP synthesis, the BP simulation insures an acceptable accuracy of the BP analysis.

The mathematical basis of the material given in the textbook includes systems analysis, probability theory, and theory of queuing networks. Some knowledge of these mathematical disciplines is desirable, but not necessary since the basic introduction is given whenever needed.

The theory is illustrated with a comprehensive example used throughout the text.

The text contains 10 chapters.

Chapters 1 and 2 contain the definitions of the basic ideas of system analysis, which are relevant to the BP M&S.

Chapter 3 is devoted to the basics of the IDEF3 standard employed as a construction tool of the BP conceptual modeling.

Chapter 4 explains the fundamentals of ABC as the basis of the BP cost metrics.

Chapter 5 offers the methods of the BP analytical modeling based on dynamic and linear programming, and theory of queuing networks.

Chapters 6 through 9 are dedicated to the topic of the BP simulation.

Chapters 6 and 9 contain the basics of the BP simulation using GPSS and a case study of the BP GPSS model development.

Chapter 7 describes the process of estimating the accuracy of the BP simulation exploiting the methods of mathematical statistics.

Chapter 8 describes the approach to the BP simulation with the timed Petri nets.

Chapter 10 contains a sample BP M&S training task.

CHAPTER 1. MODELING AND SIMULATION AS THE BUSINESS PROCESS REENGINEERING TOOL

"Business process reengineering (BPR) is the fundamental rethinking and radical redesign of business processes to achieve dramatic improvements in critical, contemporary measures of performance, such as cost, quality, service and speed" [1].

The following four key words are to be outlined in this definition:

- **Fundamental** means starting from scratch in the business organization rethinking;
- **Radical** implies demolishing the company structure rather than marginally improving it;
- **Process** indicates focusing on the process rather then the structure. This concept requires a detailed investigation of the process-oriented mapping of activities;
- **Measures of performance** mean the estimating of the system's quantitative and qualitative characteristics.

To achieve the desired significant improvements on the way of the radical business processes redesign, it is important to insure that there are no mistakes and those radical changes would indeed lead to the expected results.

BPR is a very costly affair. Therefore, assessing the profit of the BPR efforts is crucial to decide whether to undertake the BPR. The BP M&S is often the only way to answer this question.

1.1. Business process reengineering as a subject of system analysis

Systems Analysis is a scientific domain of the semi-structured complex interdisciplinary problem research. Systems Analysis is based on the following four fundamental principles:

- **System approach** to the subject research;
- **Hierarchy**, i.e. the multilevel research of the subject;
- **Synergy principle**, which is based on the idea that a system is greater than a mere sum of its elements;
- **Formalism** means that the subject is represented as a formal model, which makes it possible to achieve constructive results.

The subject of *Decision Taking* is one of the essential tasks of Systems Analysis, while *Operations Research* is a discipline providing the proof of the taken decision.

1.2. Applying operations research to the business process reengineering

Definitions

Operation is a collection of mutually agreed actions directed at achieving a defined goal.

Operation solution is a collection of parameters, which define the way of how the operation is executed.

Unguided operation factors are real probabilistic conditions that affect the outcome of the operation.

Operation Efficiency or *Objective Function* is a quantitative estimate of the operation outcome.

Problems of the Operations Research

Operations Research Model creation. Mathematical Operation Research Model is defined as follows:

$$f(x,z),\ (x \in X;\ z \in Z), \tag{1.1}$$

where:

$f(x,z)$ is an objective function;

x is an operation solution;

X is a set of admissible solutions;

z are unguided operation factors;

Z is a set of unguided factors.

Analysis of the Operation means estimating the selected solution, i.e. finding the value of the objective function $f(x,z)$ for the solution x.

Synthesis of the Operation means finding the optimal solution, which maximizes (minimizes) the objective function:

$$\begin{aligned} &x \ \rightarrow\ \max(\min) f(x,z) \\ &x \in X, \\ &z \in Z. \end{aligned}$$

Business process reengineering as the BP modeling problem

With the reference to the BPR, the Operations Research Model creation can be formulated as determining the dependence of the objective function on the business process parameters.

Selecting the objective function is left beyond the scope of the book. The objective function may be, for example, a temporal or cost BP parameter or an integral criterion in the form of a weighted sum of some local criteria.

If it is possible to construct BP analytical model, i.e. to find an explicit function $f(x,z)$ of the BP organization parameters, the problem of the BP optimization turns out to be a problem of *mathematical programming*.

If it is impossible to create a BP analytical model, the problem of determining $f(x,z)$ can be solved by means of *simulation* in conjunction with the *statistical testing*.

To sum up, the BP mathematical model creation is a necessary stage of its research. Some examples of defining the BP analysis problems in terms of the operations research are presented in Sec. 5.3.1 and 5.3.2.

CHAPTER 2. INTRODUCTION TO MODELING AND SIMULATION

A thorough introduction to the system simulation with a number of examples can be found in [2].

Simulation is an imitation of a real-world process or system over a period of time. Simulation involves creating a *simulation model* of the real system, generating an artificial history of the system by substituting the system with the *simulation* model, and observing the artificial history to draw the inferences regarding the operation characteristics of the real system.

Sometimes it is possible to develop such a model, which is simple enough to be handled by the analytical methods. If this is the case, the model is called *analytical*.

Both *analytical* and *simulation* models are represented in the mathematical form (formulae, tables, plots, algorithms, logic expressions, etc.).

2.1. Analytical modeling vs. simulation

The analytical model enables to obtain the most general results on the behavior of the real system, which are usually represented in the form of strict mathematical expressions.

A disadvantage of the analytical modeling lies in its limited capabilities of adequately mapping real complex systems. The analytical model is bounded by its mathematical scheme.

The simulation model has a larger "mapping capacity" than the analytical model because it is easier to observe the behavior of a system than to get into its underlying principles of management and development.

In many cases simulation is the only way to start the development of a complex system.

A disadvantage of the simulation approach is the form of the modeling results. Each simulation run provides a point-to-point mapping for the sets of the input and the output variables. This disadvantage can be tackled by conducting multiple simulations.

2.2. Types of models

There are many ways of classifying system models. One possible way with five most relevant clustering criteria is shown in Fig. 2.1.

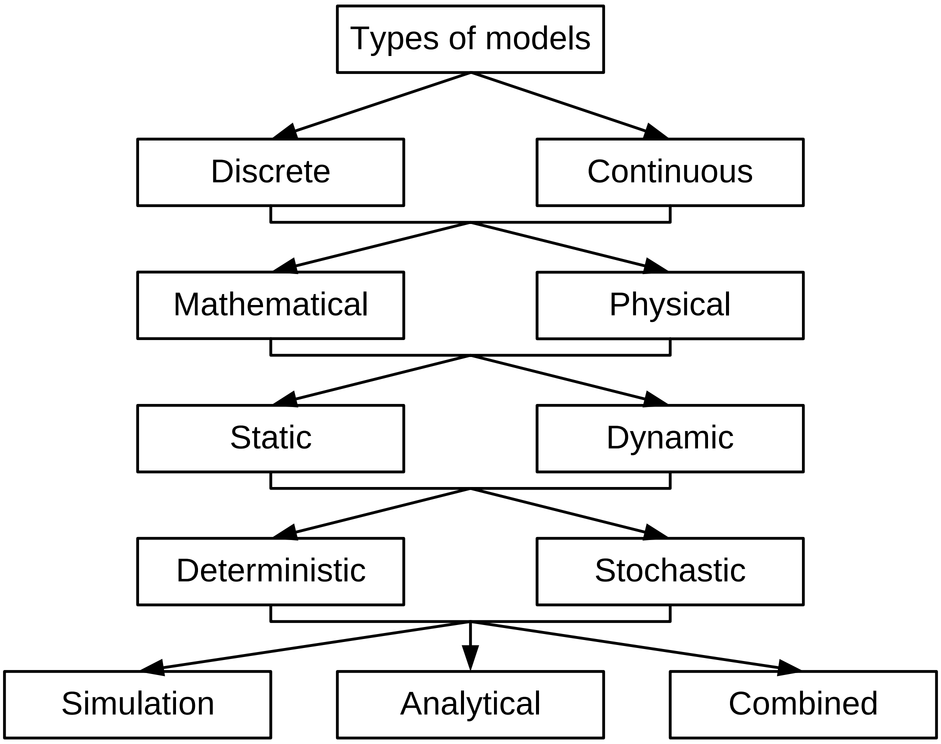


Figure 2.1. Classification of models

- Discrete vs. continuous models (it often depends on the point of view rather than the nature of the modeled system);
- Mathematical vs. physical (material, not virtual);
- Static vs. dynamic;
- Deterministic vs. stochastic;
- Simulation vs. analytical (or combined).

2.3. Structure of the simulation process

The main steps of the simulation study are listed below in the order of their execution and interdependencies (see Fig. 2.2):

- Formulating the problem;
- Setting the objectives;
- Conceptualizing the model;
- Defining the initial data and the quantitative parameters of the model;
- Translating the model;
- Verifying and validating the model;
- Executing the simulation runs and analyzing the results;
- Documenting and reporting.

Let us consider these steps in more detail.

Formulating the problem and setting the objectives

M&S is aimed at reflecting features of the system in question. There is no sense to start modeling without having the purpose of modeling in mind. So, at first the problem has to be formulated (questions, hypotheses).

It is essential to set the objectives of M&S, which are relevant and adequate to the problem at hand. The objectives may include costs, temporal parameters, reliability etc. Hardly is it possible to model the system unless the objectives are clearly defined.

Conceptualizing the model

The first stage of formalizing the model is to select the basic assumptions, which characterize the system results in the conceptual model creation. The conceptual model defines the system elements and the way of their interaction. It is the basis of the mathematical scheme to be implemented in the form of a simulation or an analytical model.

Translating the model

The model must be represented in a computer recognizable format. There are two competing ways of solving this problem: (1) building the model in some universal programming language and (2) using a dedicated simulation language. Each way has its pros and cons. On the one hand, the universal programming language approach is often easier, but it requires quite a good self-made simulation system for driving the simulation program. On the other hand, the dedicated simulation language gives the modelers all the abilities of a powerful simulation system, but it constrains them by the limits of the modeling concept.

Verifying and validating the model

Verification refers to the process of insuring that the model is free from logical errors, i.e. that the model does what it is intended to do.

Validation insures that the model is a reasonable and valid representation of the real system or problem.

Unlike validation, the task of verification is quite formal.

Executing the simulation runs and analyzing the results

This step has two major aspects: (1) designing the simulation experiment to generate the output data, which would be sufficient for the subsequent system investigation, and (2) determining the number of observations required for achieving the desired precision of the simulation results. The second aspect will be considered in Ch. 7.

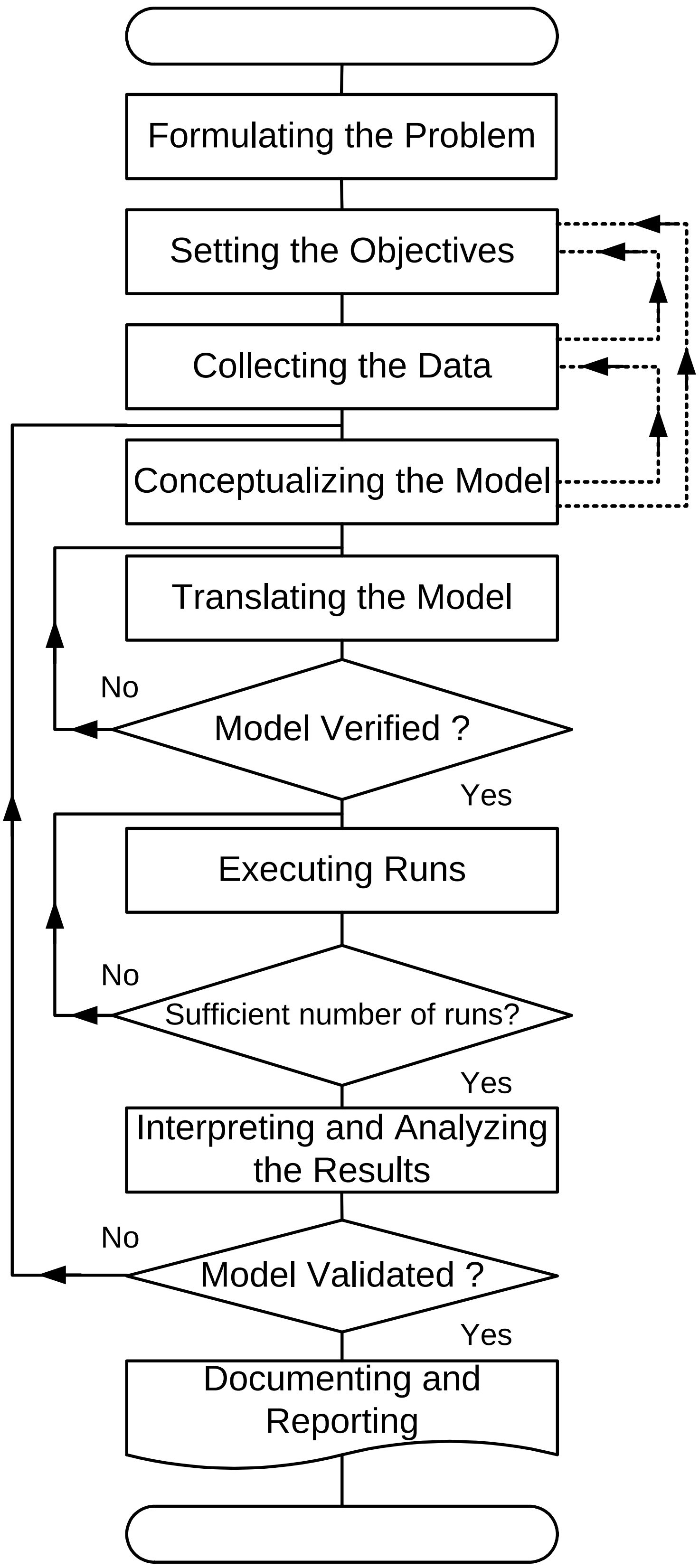


Figure 2.2. Structure of the simulation process

CHAPTER 3. USING IDEF FORMAT FOR THE BP MAPPING

3.1. Function modeling (IDEF0)

IDEF stands for the Integration Definition. It also refers to a family of mutually-supportive methods for enterprise integration.

IDEF0 is the Integration Definition for Function Modeling.

IDEF0 model is a graphical description of the system, which is developed for a specific purpose and considered from a certain viewpoint.

Function is an activity identified by a verb or verb phrase that describes what must be accomplished.

Box is a rectangle, which contains a name and a number. It represents a function.

Input Arrow is a class of arrows that expresses the IDEF0 input, i.e. the data or the objects that are transformed by a function into the output.

Output Arrow is the class of arrows that expresses the IDEF0 output, i.e. the data or the objects produced by a function.

Mechanism Arrow is a class of arrows that expresses the IDEF0 mechanism, i.e. the means of performing a function.

Call Arrow is a type of the mechanism arrow that enables to share details between models (linking models together) or within a model.

Control Arrow is a class of arrows that expresses the IDEF0 control, i.e. the conditions required to produce the correct output.

Function modeling (IDEF0) uses activities and arrows to describe and to document the business processes, see Fig. 3.1. Creating a business process model starts from the context diagram with the only activity, which represents the model, see Fig. 3.2.

The context diagram depicts the highest-level activity in the model. It represents the boundary of the process with respect to the purpose, scope, and viewpoint. Subsequently, one can add the decomposition of the activities and the arrows to specify and to refine the business process model.

The IDEF0 standard uses several types of links between the activities. The arrows can change their type. For example, an Output arrow may become a Control arrow and vice versa. IDEF0 employs a multilevel decomposition of the activities in order to come to a set of elementary activities, which can be explicitly depicted. That is the way to create a coherent function description.

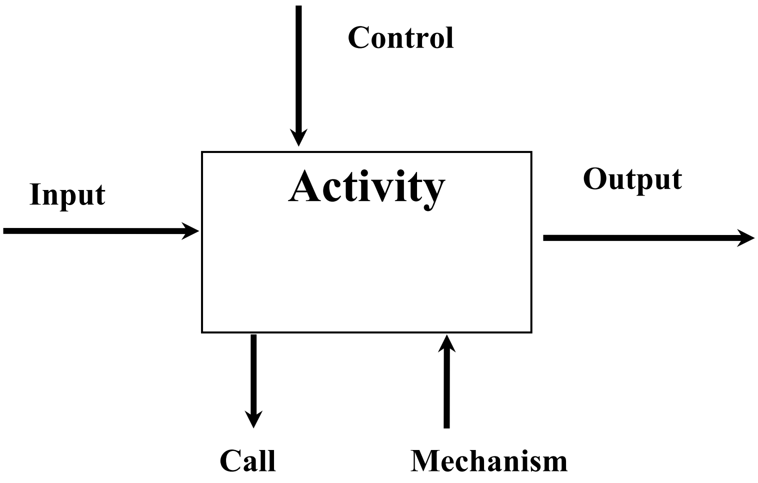


Figure 3.1. Function modeling (IDEF0)

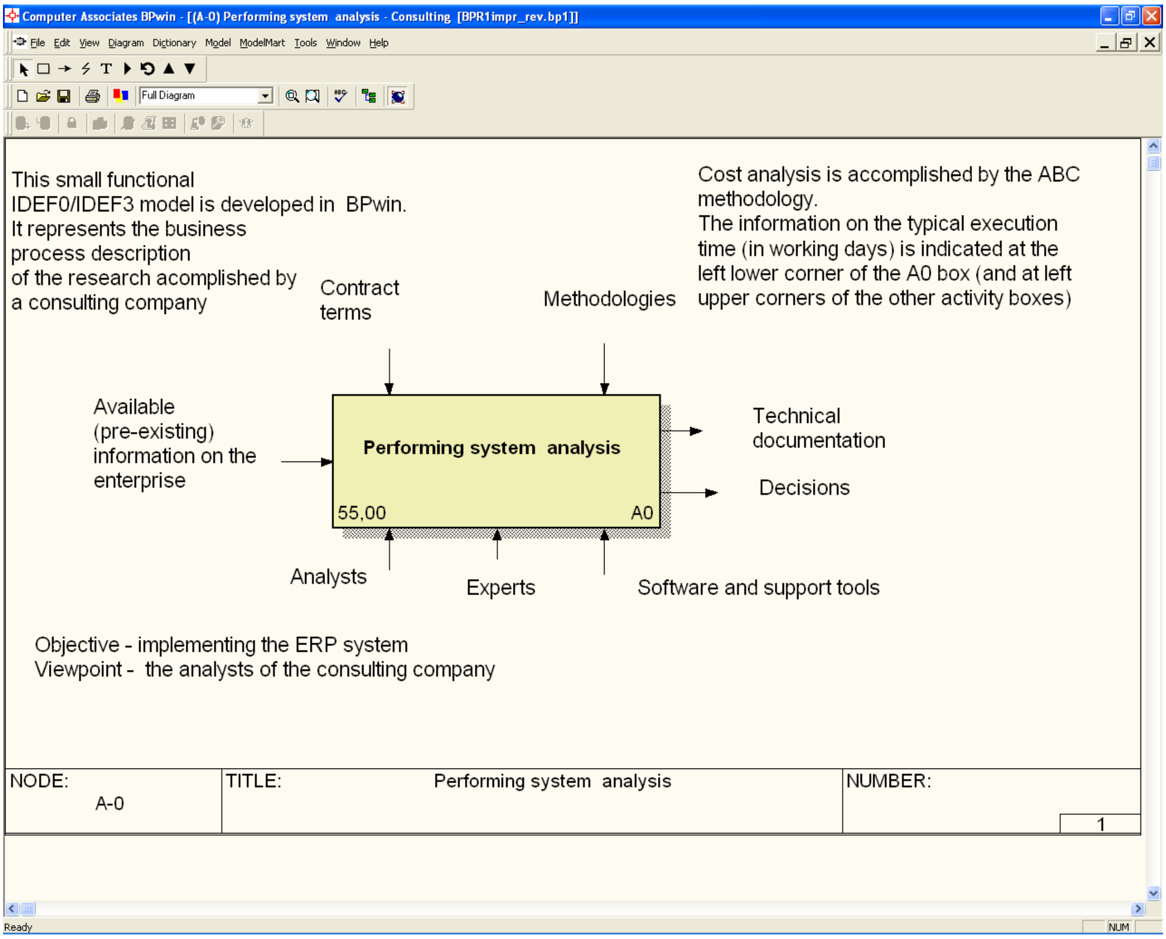


Figure 3.2. Context diagram of a business process (screenshot: BPwin ®)

3.2. Process modeling (IDEF3)

Process Flow modeling, also referred to as IDEF3 modeling, is a modeling methodology used for graphical description and documentation of processes by capturing the information on the process flow, the relationships between processes, and those important objects that are part of the process.

Unit of Behavior (UOB) is a term used in IDEF3 to describe types of events or “happenings”.

UOB Box (Unit of Work, UOW) is a syntactic element of the IDEF3 Schematic Language, which is employed to represent a real-world process. It is similar to the Activity term in the IDEF0 format.

Decomposition of a UOB is a method of tackling the process complexity.

Processes are represented on the schematic diagram as labeled boxes, see Fig. 3.3. Each box represents packets of information about an event, a decision, an act, or a process, i.e. the types of “happenings”.

The logic functions named “junctions”, which are included in the IDEF3 syntax, are aimed at defining the logic connection between the source and the target boxes.

The information about a UOB comprises:

1. Name (often verb-based) indicating what the UOB represents;

2. Names of the objects, which are included in the process, and their properties;

3. Relations between the objects. The arrows (called links) connecting the boxes indicate the precedence relationships (more generally, constraints) between the processes. Thus, the UOB instance at the source of a link completes prior to the start of the UOB instance at the end of the same link.

The BP mapping by means of the IDEF3 standard becomes a true conceptual BP model when it is filled with data (mostly quantitative) about all the details of each UOB.

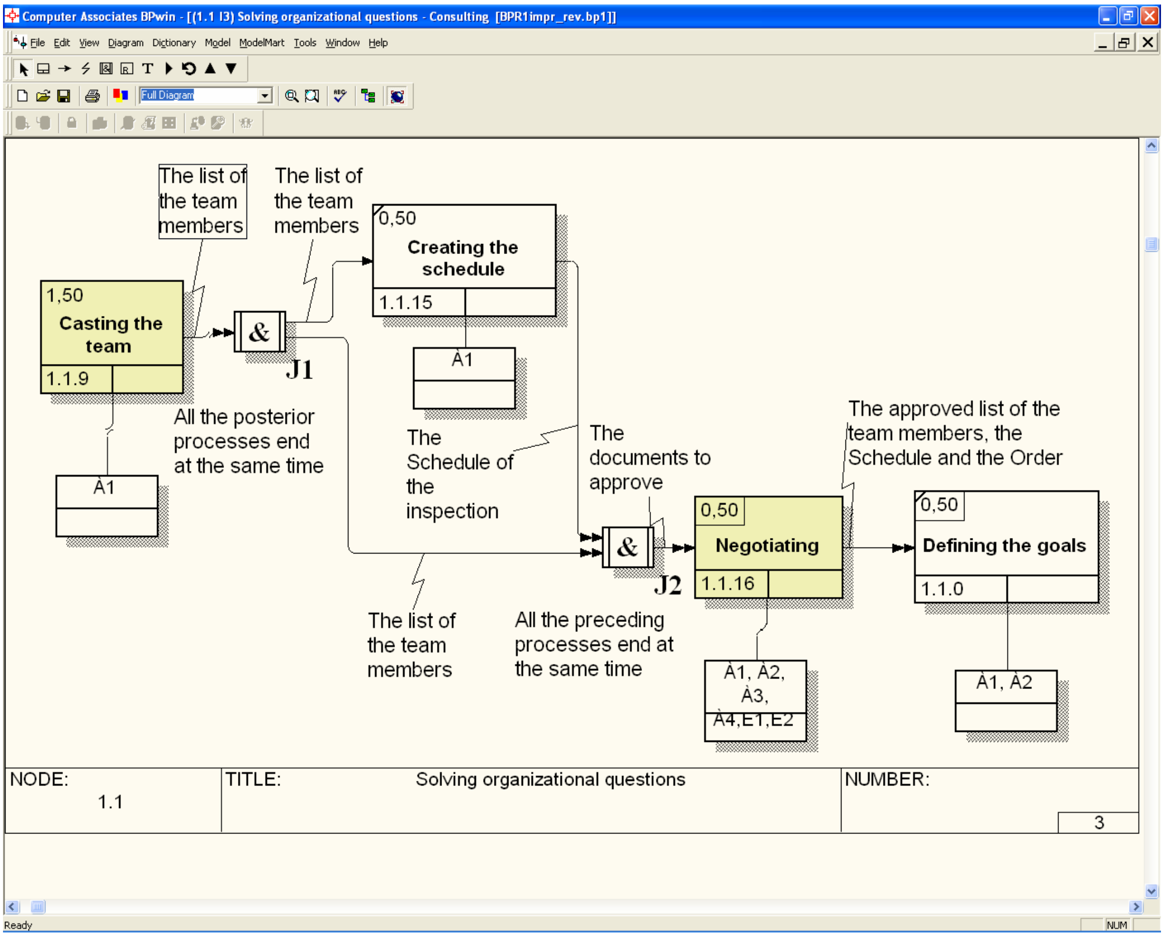


Figure 3.3. Representation of processes (screenshot: BPwin ®)

CHAPTER 4. ACTIVITY-BASED COSTING AS BP COST METRICS

4.1. ABC as an approach to the true costs analysis

Activity-based Costing (ABC) is a management methodology developed in the 1980's as a practical solution for some of the problems associated with the traditional cost management systems [3] [1]. The problem of cost metrics is relevant to the BP modeling for the following two reasons. First, the cost metrics are important for estimating the BP efficiency. Second, since a simulation model is capable of performing only those actions that are determined in the form of strict algorithm, the algorithm should include the calculations of the cost metrics.

Companies may have significant overhead costs. Every company has to spend financial resources to maintain a number of its activities, which are not directly profitable, for instance, marketing, scientific research, consulting assistance and so on. An important problem of the traditional cost systems is their inaccuracy in assessing the overhead costs.

The activity-based cost systems extend traditional ones by linking the resource expenses to the *variety* and the *complexity* of products rather than only to the physical volumes of production. The ABC principles are explained in detail in [3], the book of the ABC founders.

To highlight the difference between ABC and the traditional methods, let us examine the structure of the traditional cost system (Fig. 4.1), where the factory overhead costs are allocated to the *production cost centers*. Allocating (not evaluating) the overhead costs is the key feature of the traditional cost systems. The traditional cost systems easily fail when allocating the overhead expenses of the cost centers (in proportion with the direct labor hours or the machine hours) to the production cost centers.

Sometimes the best of the traditional cost systems can be quite accurate if they directly assign the overhead costs to the production cost centers (based on the actual usage). However, even these systems fail at the next stage, when the costs, which are accumulated at the production cost centers, are assigned to the products processed at each center.

[1] The material of this chapter is partially based on [3]

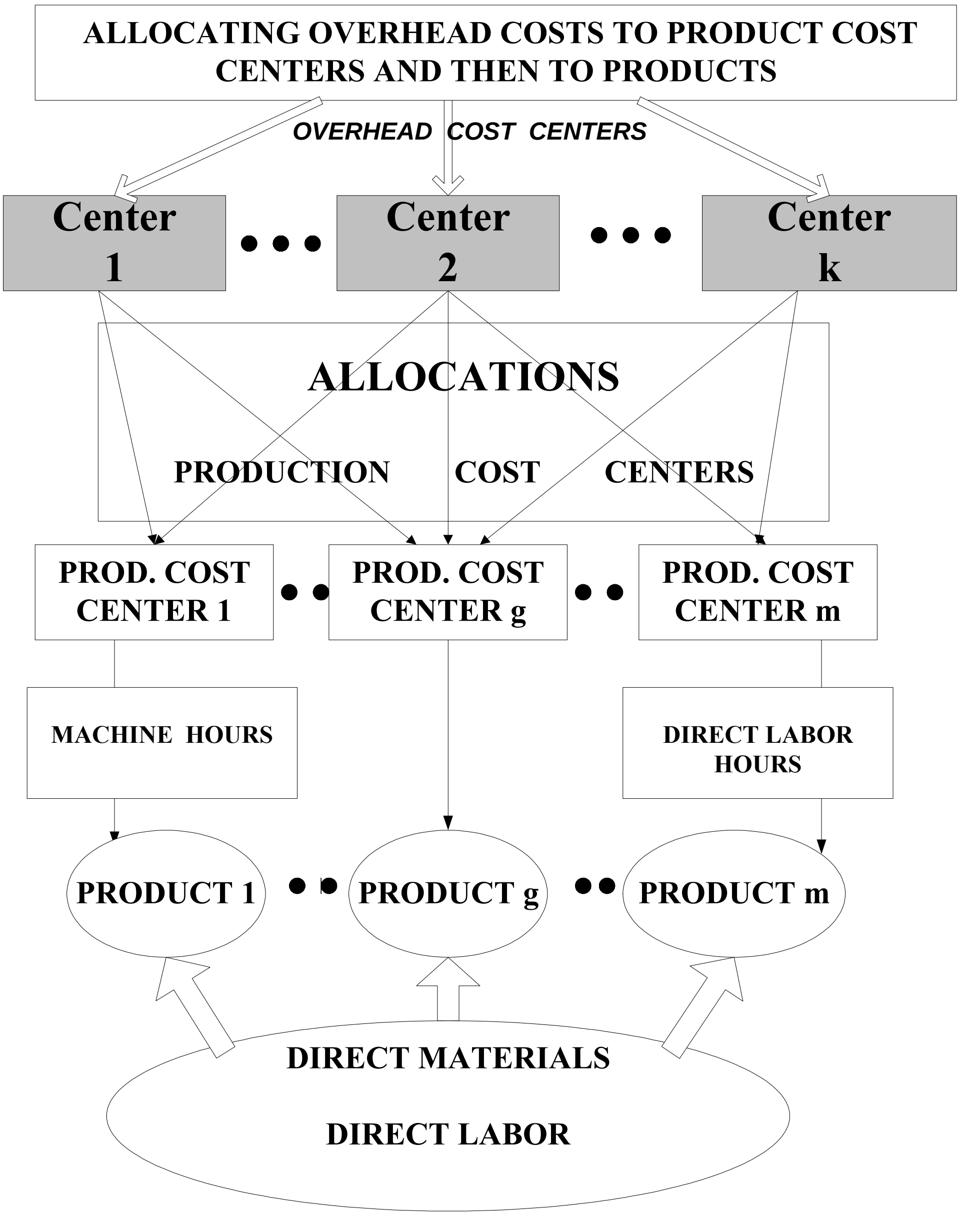


Figure 4.1. Structure of the traditional cost system (based on [3], p.83)

Fig. 4.2 shows the structure of an Activity-Based Cost system. At the first glance, the ABC system appears to be similar. Nevertheless, the underlying structure and the concept are quite different. The focus of the ABC is shifted from *how* to allocate the costs to *why* the organization spends the financial resources in the first place.

The development of the ABC systems comprises four steps [3], which are discussed in the next section.

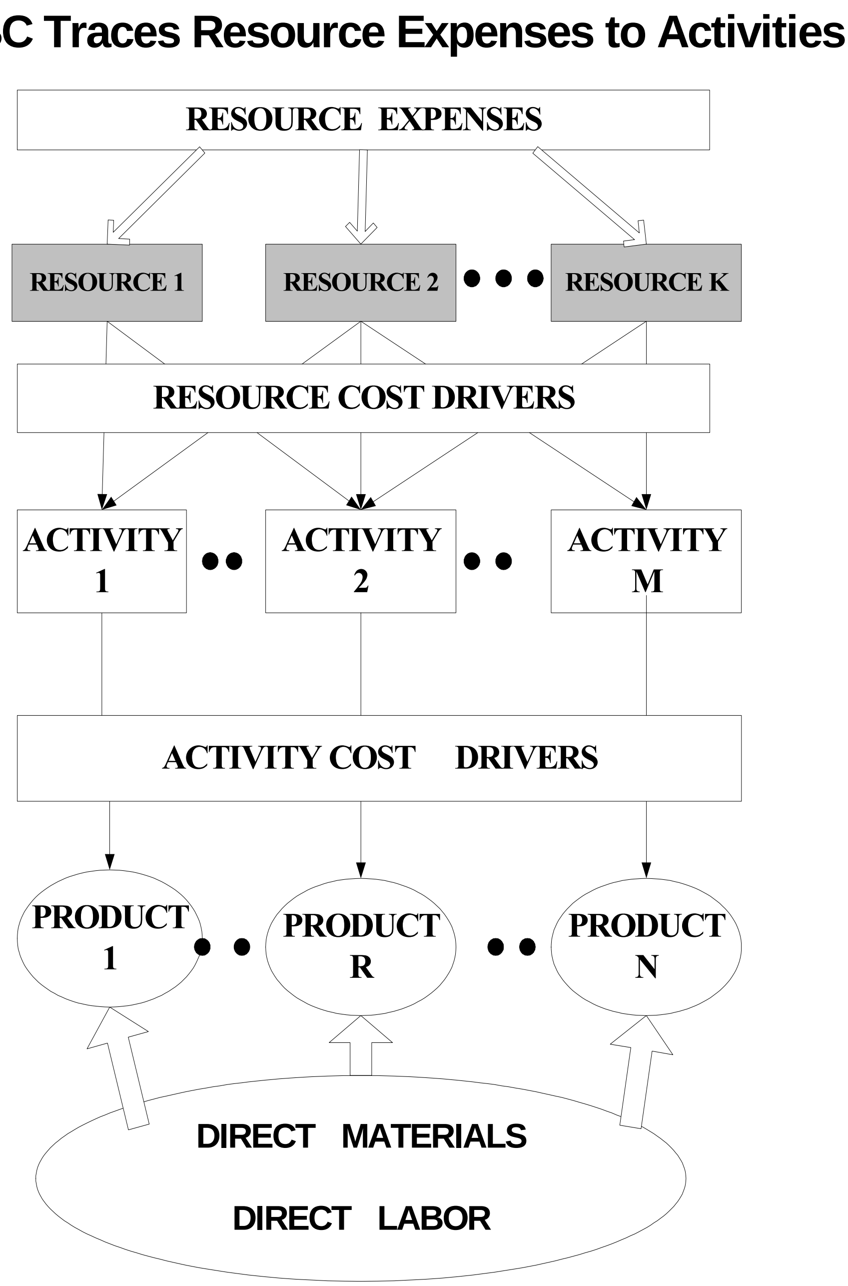


Figure 4.2. Structure of the ABC system (based on [3], p. 84)

4.2. Steps of ABC

<u>Step 1. Develop the activity dictionary</u>

First of all, the organization has to identify the activities associated with its indirect and support expenses.

The activity dictionary is a list of the activities employed within the organization. An entry of the list contains the name of an activity, its definition, output, and appropriate classification.

The activity dictionaries can be relatively brief, for example containing some 10 – 30 activities, especially if the prime focus of the ABC system is on estimating the product and the customer costs. The activities that consume less than 5% of the resource capacity are usually not included in the dictionary. However, there exist ABC systems with hundreds of activities.

Step 2. Determine how much the organization spends for each of its activities

The ABC system maps the resource expenses to the activities by means of the *resource cost drivers*. The resource cost drivers link the spending and the expenses, as captured in the general accounting system, to the performed activities. Business process cost structure should comply with its IDEF-model. The actual mechanics of selecting the resource cost drivers and estimating the quantity of each resource cost driver have to be reasonably well described in the model. BPwin Cost Editor[2] can be employed at the step.

Step 3. Identify the organization's products, services, and customers (cost objects)

Step 3 is simple but important. The question of whether one or another activity or process is worth doing is not trivial. Answering the question requires the activity costs being linked to the products, services, and customers.

Step 4. Select the activity cost drivers, which link the activity costs to the organization's products, services, and customers

A *cost object* is a product, a service, a customer, a location, a unit, a project or a work objective, for which an individual cost measurement is needed.

The linkage between the *activities* and the *cost objects*, such as products, services, and customers, is accomplished by means of the *activity cost drivers*. An activity cost driver is a quantitative measure of the output of an activity.

Selecting the activity cost drivers involves a trade-off between the accuracy and the cost of measurement. The linkage can be given in the form of an incidence matrix with the rows (columns) corresponding to cost objects (activities). Therefore, the number of the matrix elements grows combinatorially as the ABC model dimension increases.

Some examples of the activity cost drivers selection are given in Tab. 4.1.

The described sequence of the ABC steps is not absolutely strict. It appears to be an iterative procedure with the returns to the previous steps should those be needed.

---

[2] *BPwin* is a popular software tool for creating the IDEF models. The latest version of BPwin is named *AllFusion Process Modeller*. Both software products are the registered trademarks of Computers Associates International Inc.

Table 4.1. Examples of the Activity Cost Drivers ([3], p. 95)

Activity	***Activity Cost Driver (ACD)***
Run Machines	Machine Hours
Set Up Machines	Setups or Setup Hours
Schedule Production Job	Production Runs
Modify Product Characteristics	Engineering Change Notices

Having provided a short introduction to the ABC fundamentals we will now compare different cost models and consider some ABC toolkits.

4.3. Comparison of the cost models

Suppose that a factory produces three types of products. Let us calculate the true cost price of each of the products.

The total expenses of the company are known from the accounting. Tab. 4.2 and 4.3 demonstrate the difference between the results of the cost price calculation with the traditional and the ABC cost models.

Comparing the results of costing we can conclude that the traditional approach leads to misleading calculations. According to traditional costing, the price assigned for each product provides its 20% profitability. Therefore, we should be interested in producing all of them.

In reality, when the overheads are *estimated* rather than *assigned*, it turns out that the products have quite different profitability, which is not equal to 20%. Hence, one of the products does not worth being produced.

A good practical guide to Activity Based Costing can be found in [4].

Table 4.2. Traditional costing (based on [3])

	DIRECT EXPENSES (DE): DE = DM + DL			**OVERHEAD (O)** ***allocated***	**Cost Price (CP) CP = DE + O**	**Price = 1.2C** **20% profitability**	**Profit (%)**
	DIRECT MATERIALS (DM) 10$/kg	**DIRECT LABOR (DL) 10$/hour**	**DE = DM + DL**				
Prod. 1	**10$×5=50$**	**10$×10=100$**	**150$**	**195$**	**345$**	**414$**	**20**
Prod. 2	**10$×10=100$**	**10$×15=150$**	**250$**	**325$**	**575$**	**690$**	**20**
Prod. 3	**10×20=200$**	**10$×20=200$**	**400$**	**520$**	**920$**	**1104$**	**20**
TOTAL			**800$**	**1040$**	**1840$**	**2208$**	

Table 4.3. Activity-based costing

	OVERHEAD IS THE SUM OF ALL THE ACTIVITIES EXPENSES (AE)				E = DM + DL	Cost Price (CP) CP = AE+DE= O+DE	Price	Profit (%)
	ACTIVITY 1	ACTIVITY 2	ACTIVITY 3	Total				
	Ordering Purchasing Delivering Material testing	Production operations	Warehousing Shipping Invoicing					
Prod. 1	**90$**	**130$**	**60$**	**280$**	**150$**	**430$**	**414$**	**-4**
Prod. 2	**100$**	**150$**	**50$**	**300$**	**250$**	**550$**	**690$**	**25**
Prod. 3	**220$**	**210$**	**30$**	**460$**	**400$**	**860$**	**1104$**	**28**
TOTAL EXPENSES				**1040$**	**800$**	**1840$**	**2208$**	

4.4. ABC toolkits

All Fusion Process Modeler

All Fusion Process Modeler (the former *BPwin*[3]) worth mentioning although the ABC is not its main function. However, BPwin Cost Editor provides the IDEF0 model with the actual costing. If BPwin is supplied with RPTwin, the toolkit for creating reports, it makes it possible to perform a more thorough cost analysis for the IDEF0 and IDEF3 models. All IDEF0/IDEF3 models, which are presented in this book, have been created with an evaluation copy of All Fusion Process Modeler.

Oros of ABC Technologies

ABC's core product is *OROS*[4], an integrated set of the activity-based software applications that allow companies to track and to manage their critical activity information. The OROS modules are designed to work with one another to provide an integrated solution for implementing business improvements.

Oros Quick is an evaluation version of *OROS ABC Plus module*. The Oros Quick desktop is depicted in Fig. 4.3.

---

[3] *BPwin* is the software package of Computers Associates for the BP mapping into the IDEF models

[4] *OROS* is the registered trade mark of ABC Technologies

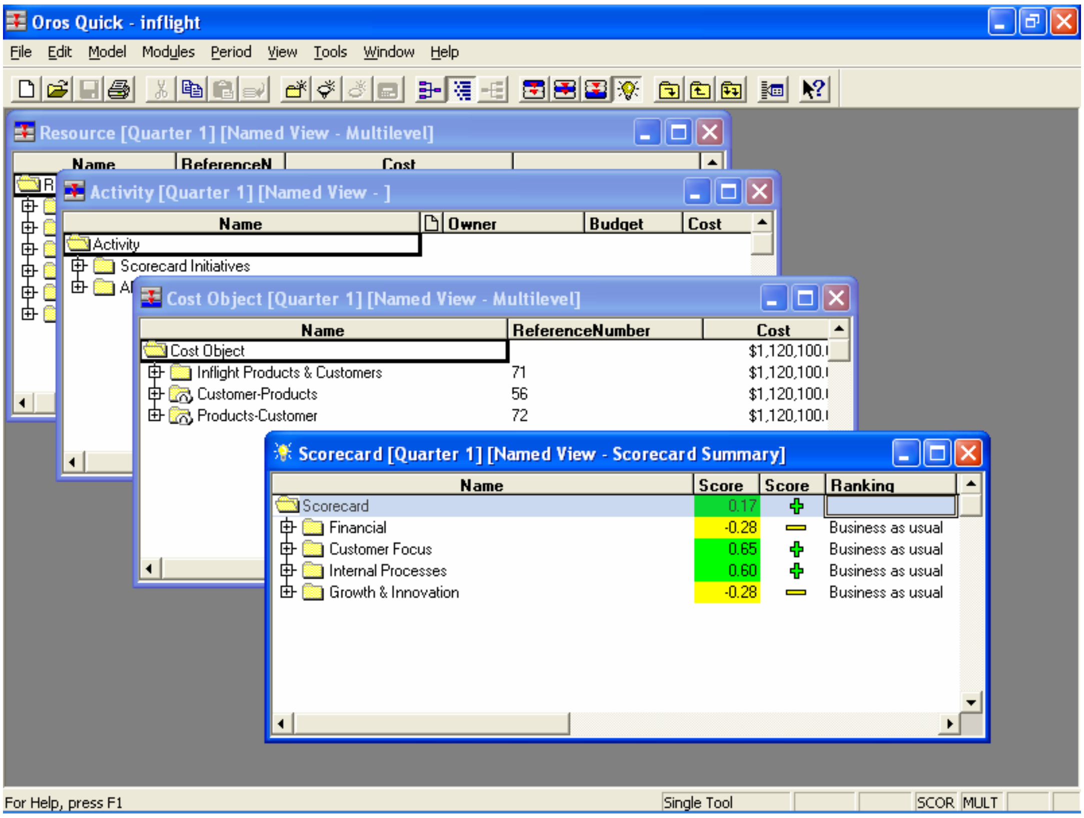


Figure 4.3. Oros Quick desktop (screenshot: OROS ®)

CHAPTER 5. ANALYTICAL BP MODELING

5.1. Steps of the analytical modeling

The principle advantage of the analytical modeling is that it leads to the general results, whereas the simulation results are specific for a simulation run. Let us consider the steps of creating analytical models, Fig. 5.1.

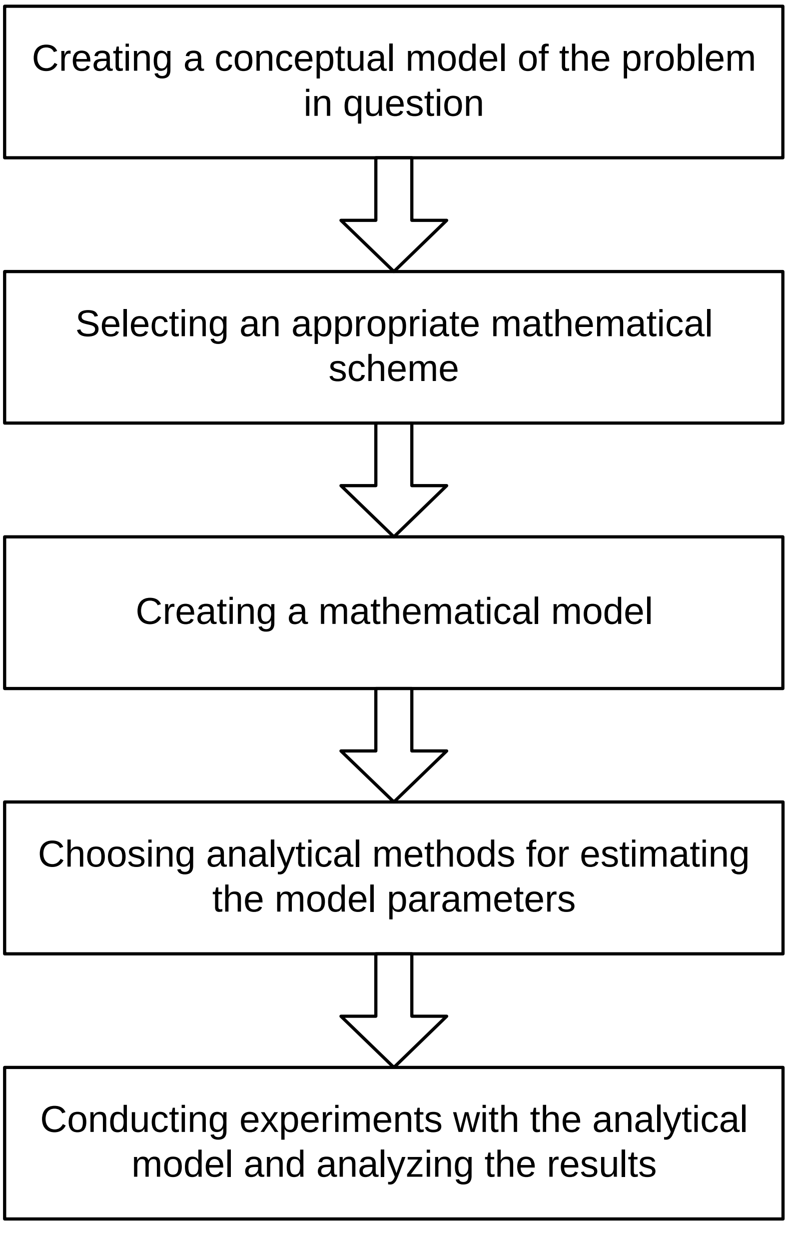


Figure 5.1. Creating a mathematical model

Since the *activities* are the basic BP elements, the conceptual BP model is "responsible" for representing their order and logic of execution. Once created, the conceptual model tends to become a formalized one. As far as the IDEF3 standard is concerned, we can state that due to its formalism and high expressive power it constitutes a solid ground for creating the BP analytical models.

5.2. Classification of the BP models

The BP models are classified into deterministic and stochastic. The models that contain no random variables are deterministic. A stochastic model has one or more random variables either as an input or as a model parameter.

In the deterministic framework, the order and the conditions of the activity execution are assumed to be invariable. That implies an invariable customer routing and an invariable service procedure, specifically, constant service time.

In the stochastic framework, a BP model presumes no pattern for either the order of activity executing or the service procedure.

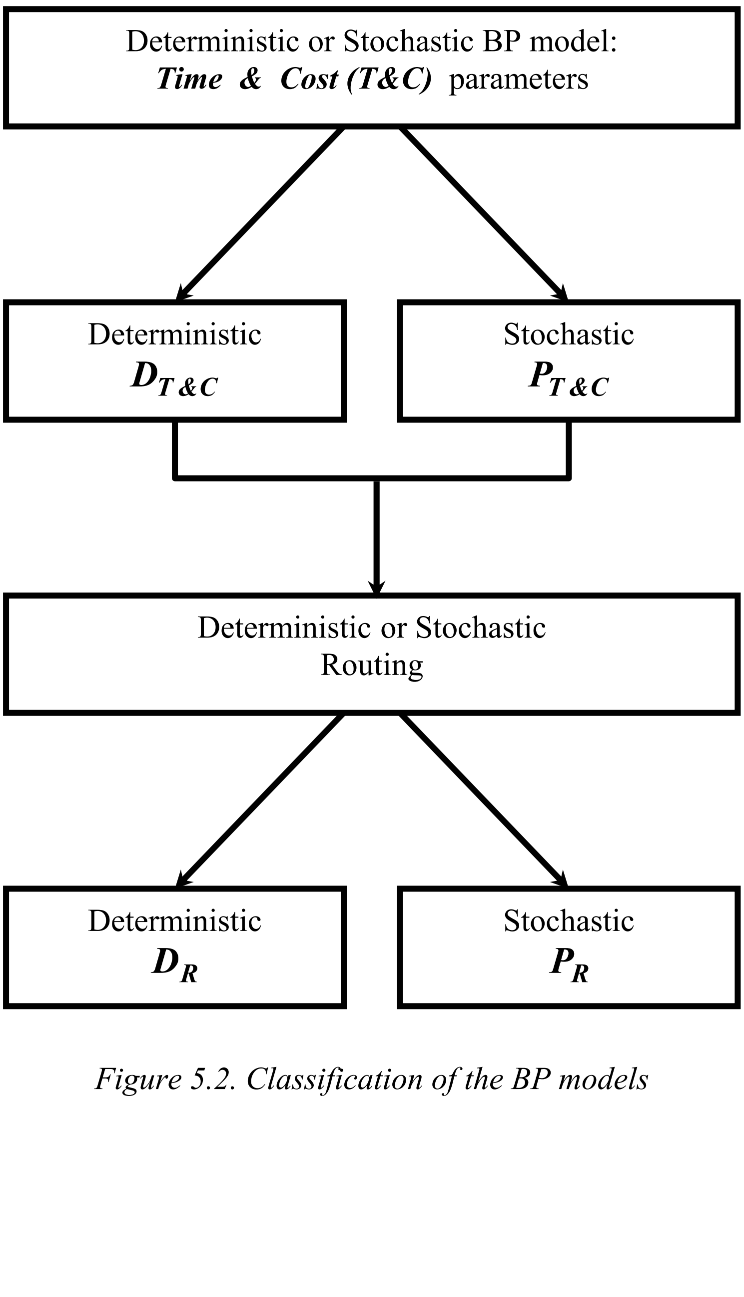


Figure 5.2. Classification of the BP models

According to this classification, we can distinguish the following four groups of the BP models:

$D_R D_{T\&C}$	$D_R P_{T\&C}$
$P_R D_{T\&C}$	$P_R P_{T\&C}$

These groups define the adequate mathematical schemes. D_R and P_R denote the deterministic and the stochastic <u>*BP routing*</u> respectively; $P_{T\&C}$ and $D_{T\&C}$ denote the deterministic and the stochastic <u>*temporary and cost BP parameters*</u> respectively.

5.3. Deterministic BP models $D_R D_{T\&C}$

Appropriate mathematical schemes for this class of the BP model include the methods of dynamic and mathematical programming.

The initial statement of the activity execution order is usually stated in the form of the following table (Tab. 5.1.):

Table 5.1. Activity execution order

Activity denotation	***Predecessors***	***Activity Duration***
a1	*No*	*2*
a2	*No*	*1*
a3	*No*	*1*
a4	*No*	*2*
a5	*a1*	*2*
a6	*a2*	*3*
a7	*a1*	*4*
a8	*a3, a6*	*1*
a9	*a4, a5, a8*	*1*

The IDEF3 model (Fig. 5.3) and the corresponding graph model (Fig. 5.4.) are directly derived from Tab. 5.1.

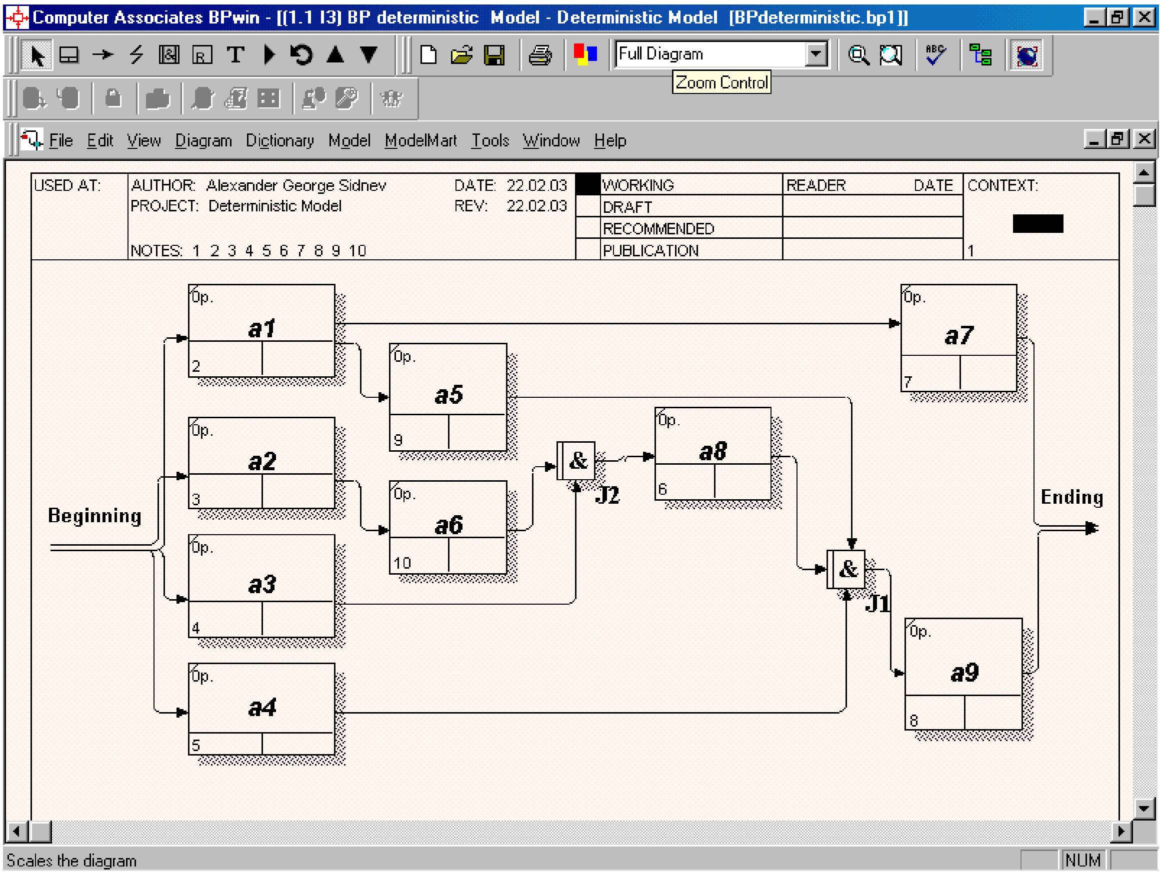


Figure 5.3. IDEF3 model of the activities (screenshot: BPwin ®)

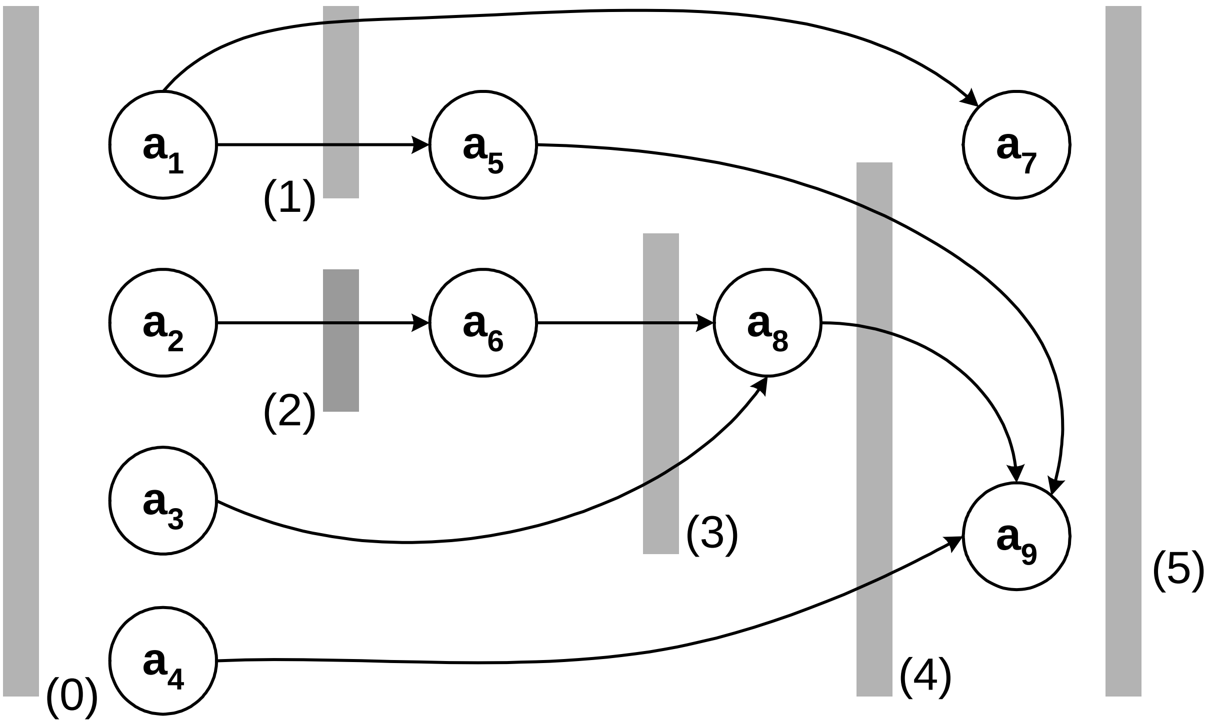


Figure 5.4. Graph model of the activities. The numbers of the states are shown in brackets

Converting the BP model into the network graph is less trivial and requires some explanation. The nodes of the network graph represent states and the arcs (state transitions) denote activities. The node i on the network graph is a *state* of the BP model, which corresponds to the completion of all activities denoted by the graph arcs that are entering the state i. The arc ij corresponds to the *activity* between the adjacent states i and j. Thus, the BP graph model, which is represented in Fig. 5.4., has *six* states marked with numbers in brackets. The state (0) corresponds the BP beginning and the state (5) is the BP ending. At the state (1) the activity a_1 completes and the activities a_5 and a_7 are allowed to start. At the state (3) the activities a_6 and a_3 complete and the activity a_8 is allowed to start. However, since a_3 takes shorter time to complete than $a_2 + a_6$, the start of a_3 "floats" within certain bounds while "waiting" for the completion of $a_2 + a_6$ (*float start time*).

As a result, both the IDEF3 model on Fig. 5.3 and the initial graph model on Fig. 5.4 can be transformed to the network graph with 6 nodes (states) numbered from 0 through 5, see Fig. 5.5. The mapping of the initial graph of activities on the network graph is shown in Tab 5.2.

Table 5.2. Mapping of the initial graph of activities on the network graph

Activity denotation		***Predecessors denotation***		***Activity duration***
Initial graph	***Network graph (arcs)***	***Initial graph***	***Network graph***	
a_1	*01*	*No*	*No*	*2*
a_2	*02*	*No*	*No*	*1*
a_3	*03*	*No*	*No*	*1*
a_4	*04*	*No*	*No*	*2*
a_5	*14*	*a1*	*01*	*2*
a_6	*23*	*a2*	*02*	*3*
a_7	*15*	*a1*	*01*	*4*
a_8	*34*	a_3, a_6	*03, 23*	*1*
a_9	*45*	a_4, a_5, a_8	*04, 14, 34*	*1*

If we represent the network graph (Fig. 5.5) as a Gantt chart, it will be clear that there have to be at least four executors (units that execute individual activities) to run the BP. Here we do not consider any conflicts arising from one executor (a unit that executes an activity) allocated to several activities, i.e. there always exist sufficient (= infinite or equal to the total number of the activities) number of executors.

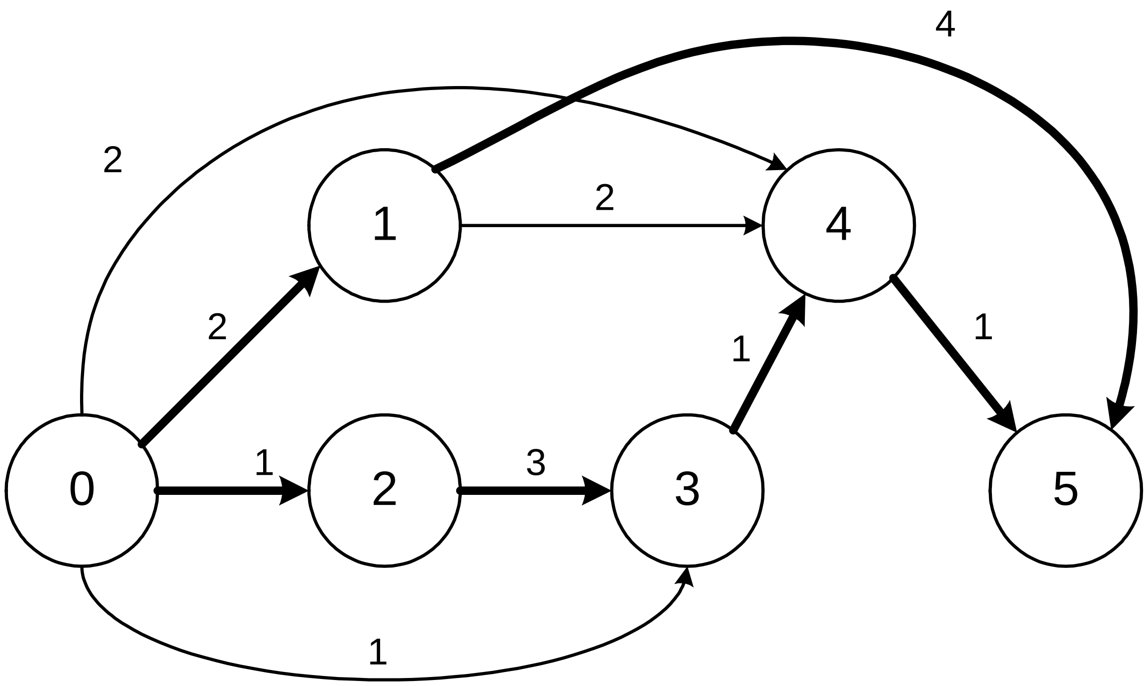


Figure 5.5. Network graph of the activities. Activities 03, 04, 14 have float start times

5.3.1. Bellman's dynamic programming for the BP scheduling

τ_{ij} is the duration of the activity ij;

t_i^* is the earliest moment of time when the event i occurs;

t_i^{**} is the latest moment of time when the event i occurs;

r_{ij} is the float start time of the activity ij.

The meaning of these terms is as follows:

t_i^* is the earliest possible moment when the event i occurs subject to the sequence order of the activities;

t_i^{**} is the latest moment when the event i occurs subject to the BP completion time equality;

r_{ij} is the *float start time* for the activity ij defined by

$$r_{ij} = t_j^{**} - (t_i^* + \tau_{ij}). \tag{5.1}$$

The problem is to derive the schedule from the graph by computing the above introduced schedule parameters.

In terms of the network planning, we need to find out the critical (longest) path on the network graph. The activities that belong to the critical path determine the BP execution (completion time).

The graph on Fig. 5.5 has five paths from the node 0 to the node 5:

$0 \to 1 \to 4 \to 5$

$0 \to 4 \to 5$

$0 \to 1 \to 5$

$0 \to 2 \to 3 \to 4 \to 5$

$0 \to 3 \to 4 \to 5$

Hence, the critical path method leads to the following expression for the total BP execution time:

$$T_\Sigma = \max \{(\tau_{01} + \tau_{14} + \tau_{45}), (\tau_{04} + \tau_{45}), (\tau_{01} + \tau_{15}), \\ (\tau_{02} + \tau_{23} + \tau_{34} + \tau_{45}), (\tau_{03} + \tau_{34} + \tau_{45})\}$$

Generally, the network graph structure is usually too complicated for deriving such an expression. An appropriate body of mathematics for solving the task is Bellman's dynamic programming [12].

Bellman's method converts the task of computing the BP completion time into a sequence of standard steps defined as follows:

$$t_i^* = \max_{j \in G^+(i)} \left\{ t_j^* + \tau_{ji} \right\}, \tag{5.2}$$

where $G^+(i)$ is the set of the nodes adjacent to the node i from *left*. For example, for the node 4: $G^+(4) = \{0, 1, 3\}$.

Equation (5.2) is used repeatedly for computing t_i^* for each node starting from the node 0. The last step concludes computing of the BP completion time.

Also, the schedule parameter t_i^{**} can be computed with Bellman's method:

$$t_j^{**} = \min_{i \in G^-(j)} \left\{ t_i^{**} - \tau_{ji} \right\}, \tag{5.3}$$

where $G^-(j)$ is the set of the nodes adjacent to the node j from *right*. For example, for the node 1: $G^-(1) = \{4, 5\}$.

Equation (5.3) is used repeatedly for computing t_j^{**} for each node starting from the end node and moving backwards to the initial one.

For the case in question, the results of the BP scheduling analysis are graphically represented in Fig.5.6. The shortest BP execution time is: $T_\Sigma = t_5^* = t_5^{**} = 6$. T_Σ is equal to the sum of the activities duration lying on each of the two critical paths: $0 \to 1 \to 5$ or $0 \to 2 \to 3 \to 4 \to 5$ (bold arcs in Fig. 5.5). The activities out of the critical path have the following float start time: $r_{03} = 3$, $r_{04} = 2$, $r_{14} = 1$.

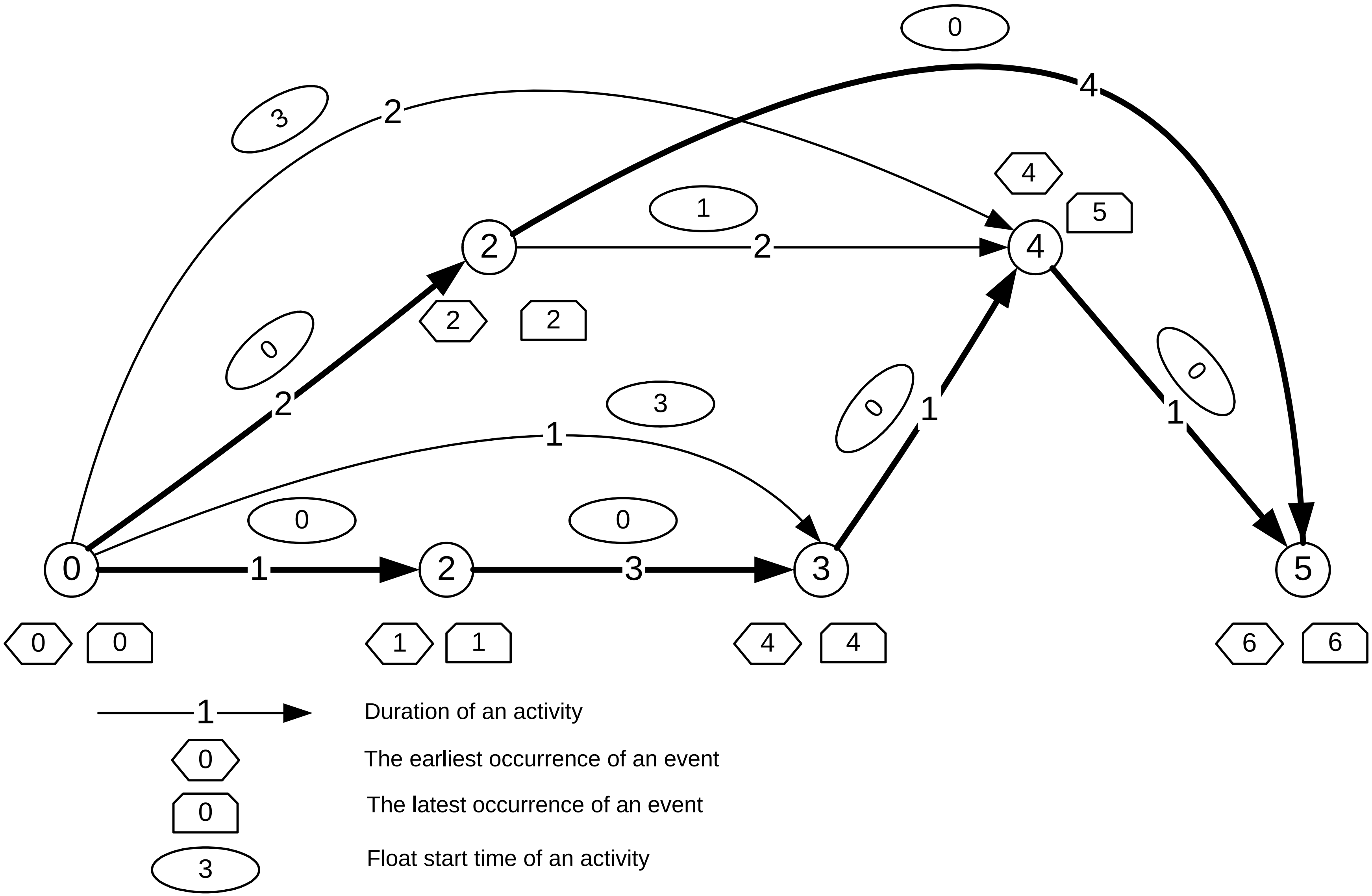


Figure 5.6. BP scheduling analysis (bold arrows depict the critical path)

5.3.2. BP scheduling as a problem of mathematical programming

Starting point

Suppose the order of the activity execution is given. The number of the activity executors is unlimited. So, the activities can start as soon as it is required.

The BP scheduling problem means revealing the actual start moments of each activity.

Let us begin with the problem definition. We are expecting the results, which are the same as ones obtained in the pervious section with Bellman's dynamic programming.

Notation

τ_{ij} is the start time of the activity ij;

T_{ij} is the end time of the activity ij;

t_{ij} is the duration of the activity ij.

Obviously, $T_{ij} = \tau_{ij} + t_{ij}$.

Problem definition

$$
\begin{aligned}
&\min \{T_\Sigma\}, \quad subject \quad to \\
&\left.\begin{aligned} \tau_{ij} &\ge T_{li} \\ T_{li} &= \tau_{li} + t_{ij} \end{aligned}\right\}, \quad i = \overline{1, M-1}; \quad l \in G^+(i), \\
&\left.\begin{aligned} T_{lM} &\le T_\Sigma \\ T_{lM} &= \tau_{lM} + t_{lM} \end{aligned}\right\}, \quad l \in G^+(M) \\
&\{\tau_{0k} = 0\}, \quad k \in G^-(0),
\end{aligned}
\tag{5.4}
$$

where $G^-(j), G^+(j)$ are the sets of the right-hand and the left-hand nodes adjacent to the node j on the BP network graph (see Fig. 5.5). T_Σ is the BP total execution time.

M is the final event node number. Therefore, the total number of the graph nodes equals $M+1$.

The logic of the activity execution order is given in the form of a set of constraints. For instance, the inequality $\tau_{ij} \ge T_{li}$ means that the activity ij does not start unless its predecessor activity li has finished.

In our example (Fig. 5.5) the problem can be defined as follows:

$$\min\ T_\Sigma\ ,\ subject\ \ to$$
$$\begin{cases}
\tau_{14} \geq T_{01} \\
\tau_{15} \geq T_{01} \\
T_{01} = \tau_{01} + t_{01} = \tau_{01} + 2 \\
\tau_{23} \geq T_{02} \\
T_{02} = \tau_{02} + t_{02} = \tau_{02} + 1 \\
\tau_{34} \geq T_{23} \\
T_{23} = \tau_{23} + t_{23} = \tau_{23} + 3 \\
\tau_{34} \geq T_{03} \\
T_{03} = \tau_{03} + t_{03} = \tau_{03} + 1 \\
\tau_{45} \geq T_{14} \\
T_{14} = \tau_{14} + t_{14} = \tau_{14} + 2 \\
\tau_{45} \geq T_{04} \\
T_{04} = \tau_{04} + t_{04} = \tau_{04} + 2 \\
\tau_{45} \geq T_{34} \\
T_{34} = \tau_{34} + t_{34} = \tau_{34} + 1 \\
T_{45} \leq T_\Sigma \\
T_{45} = \tau_{45} + t_{45} = \tau_{45} + 1 \\
T_{15} \leq T_\Sigma \\
T_{15} = \tau_{15} + t_{15} = \tau_{15} + 4 \\
\tau_{01} = \tau_{02} = \tau_{03} = \tau_{04} = 0
\end{cases} \tag{5.5}$$

Compacting (5.5) yields:

$$\min\ T_\Sigma\ ,subject\ \ to$$
$$\begin{cases}
\tau_{14} \geq 2 \\
\tau_{15} \geq 2 \\
\tau_{23} \geq 1 \\
\tau_{34} \geq \tau_{23} + 3 \\
\tau_{34} \geq 1 \\
\tau_{45} \geq \tau_{14} + 2 \\
\tau_{45} \geq \tau_{34} + 1 \\
\tau_{45} \leq T_\Sigma - 1 \\
\tau_{15} \leq T_\Sigma - 4
\end{cases} \tag{5.6}$$

The problem defined as (5.4) is a typical problem of linear programming [13, 14]. The problem has six unknown variables. It is in the form ready for solving with an appropriate tool. Here we use MATLAB `fmincon` function (Optimization Toolbox) function, Fig. 5.7.

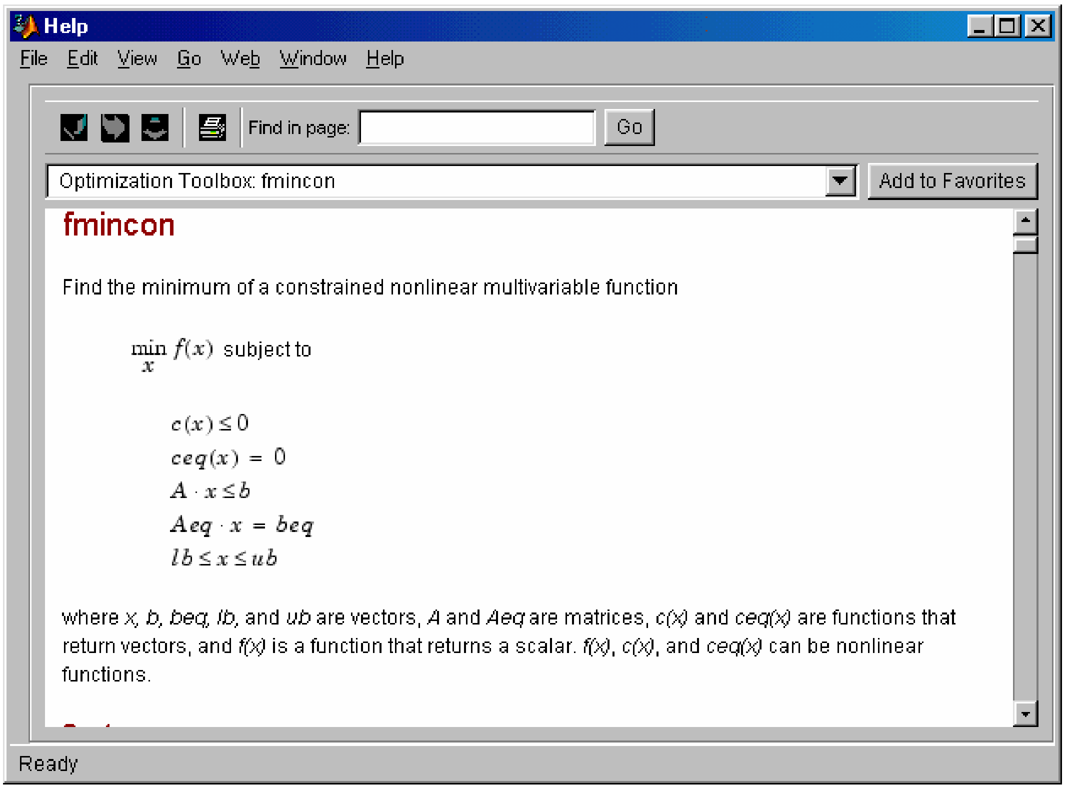


Figure 5.7. `fmincon` *function of MATLAB*[5] *(screenshot: MATLAB ®)*

According to the `fmincon` syntax, the actual MATLAB command is:

```
fmincon(inline('x(6)'),x0,A,b)
```

It is equivalent to:

$$\begin{array}{l} \min\{x6\},\ \ subject\ \ to \\ AX \le b \end{array}, \qquad (5.7)$$

where the components of the vector X define the parameters of the problem (5.6):

$$X = \{x1, x2, x3, x4, x5, x6\} = \{\tau_{14}, \tau_{23}, \tau_{34}, \tau_{45}, \tau_{15}, T_{\Sigma}\}.$$

The matrix A and the vector b are represented as the rows and columns in Tab 5.3.

Table 5.3. Parameters of the problem

x1	***x2***	***x3***	***x4***	***x5***	***x6***	$\le$	***b***
-1	0	0	0	0	0		-2
0	0	0	0	-1	0		-2
0	-1	0	0	0	0		-1
0	1	-1	0	0	0		-3
0	0	-1	0	0	0		-1
1	0	0	-1	0	0		-2
0	0	1	-1	0	0		-2
0	0	0	1	0	-1		-1
0	0	0	0	1	-1		-4

[5] *MATLAB* is the registered trademark of The Mathworks Inc., [www.mathworks.com]

Below is the dialog in MATLAB Command Window demonstrating how the problem is solved:

```
A = [ -1    0     0     0     0      0; ...
      0     0     0     0    -1      0; ...
      0    -1     0     0     0      0; ...
      0     1    -1     0     0      0; ...
      0     0    -1     0     0      0; ...
      1     0     0    -1     0      0; ...
      0     0     1    -1     0      0; ...
      0     0     0     1     0     -1; ...
      0     0     0     0     1     -1];

b=[-2;-2;-1;-3;-1;-2;-2;-1;-4];
x0=[0;0;0;0;0;0];
fmincon(inline('x(6)'),x0,A,b)

Active Constraints:
     3
     4
     7
     8

ans =
     2
     1
     4
     5
     2
     6
```

Here `ans` is the solution for the vector X defining the activities start time and the BP total execution time:

$$X = \{x1, x2, x3, x4, x5, x6\} = \{\tau_{14}, \tau_{23}, \tau_{34}, \tau_{45}, \tau_{15}, T_\Sigma\} = \{2, 1, 4, 5, 2, 6\}$$

This solution $T_\Sigma = 6$ coincides with one obtained using Bellman's method (see Sec. 5.3.1).

Suppose that each activity consumes one unit of resource from a limited resource pool. Then, having the solution X it is easy to calculate the resources occupied at every moment of time from the start to the very end of the BP. The procedure is as follows:

1. Compute the start time τ_{ij} and the end time T_{ij} for each activity ij from the solution X;
2. Sort both time sets $\{\tau_{ij}\}$ and $\{T_{ij}\}$ in ascending order;
3. Process these sets for the events of the release or the seizure of the BP executors.

This is a way to get an insight into the utilization dynamics of executors as shown in Tab. 5.4. Note that the event "Activity Start" corresponds to seizing an executor, while the event "Activity Finish" corresponds to releasing an executor.

Table 5.4. Utilization dynamics of executors

Event number	***Event time***	***Event type***	***Change in the amount of the executors in use***	***The amount of executors in use***
		Activity Start (S) / Activity Finish (F)		
1	10	S	+1	1
2	15	S	+1	2
3	16	S	+1	3
4	22	S	+1	4
5	27	F	-1	3
6	31	F	-1	2
7	33	S	+1	3
8	34	F	-1	2
9	38	F	-1	1
10	45	S	+1	2
11	50	S	+1	3
12	56	S	+1	4
13	57	F	-1	3
14	58	F	-1	2
15	70	S	+1	3
16	73	S	-1	2
17	81	F	-1	1
18	98	F	-1	0

5.3.3. BP resource optimization by means of mathematical programming

Starting point

Assume that the resources (activity executors) are identical and the activity execution order is given. A resource is seized by an activity for the total time of the BP execution. Therefore, a resource cannot be handed from one activity to another and the number of the resources cannot be less then the number of the activities. The actual problem is to minimize the number of the resources (executors) subject to the BP finishing in time.

Notation

m_{ij} is the number of the resources assigned to the activity ij;

τ_{ij} is the activity ij start time;

T_{ij} is the activity ij end time;

Q_{ij} is the activity ij labor expenditures (e.g. man-hours);

t_{ij} is the activity ij duration (e.g. hours).

According to the meaning of the activity parameters: $T_{ij} = \tau_{ij} + t_{ij} = \tau_{ij} + \dfrac{Q_{ij}}{m_{ij}}$.

The problem of the resources optimization can be stated as follows:

$$
\begin{array}{l}
\min \left\{ \sum_{(ij)} m_{ij} \right\}, \ \textit{subject to} \\
\left. \begin{array}{l} \tau_{ij} \ge T_{li} \\ T_{li} = \tau_{li} + \dfrac{Q_{li}}{m_{li}} \end{array} \right\}, \quad i = \overline{1, M-1}; \quad l \in G^{+}(i), \\
\left. \begin{array}{l} T_{lM} \le T_{\Sigma} \\ T_{lM} = \tau_{lM} + \dfrac{Q_{lM}}{m_{lM}} \end{array} \right\}, \quad l \in G^{+}(M) \\
\{\forall ij \ \ m_{ij} \in N \\
\{\tau_{0k} = 0\}, \quad k \in G^{-}(0),
\end{array}
\tag{5.8}
$$

where $G^{-}(j)$, $G^{+}(j)$ are the sets of the right-hand and the left-hand nodes adjacent to the node j on the BP network graph. M is the final event node number; N is the set of the integer numbers. For the graph model in Fig. 5.5, the problem definition is the following:

$$
\min \ (m_{01} + m_{02} + m_{03} + m_{04} + m_{14} + m_{15} + m_{23} + m_{34} + m_{45})
$$

$$
\begin{cases}
\tau_{14} \ge T_{01} \\
\tau_{15} \ge T_{01}, \quad T_{01} = \tau_{01} + \dfrac{Q_{01}}{m_{01}} \\
\tau_{23} \ge T_{02}, \quad T_{02} = \tau_{02} + \dfrac{Q_{02}}{m_{02}} \\
\tau_{34} \ge T_{23}, \quad T_{23} = \tau_{23} + \dfrac{Q_{23}}{m_{23}} \\
\tau_{34} \ge T_{03}, \quad T_{03} = \tau_{03} + \dfrac{Q_{03}}{m_{03}} \\
\tau_{45} \ge T_{14}, \quad T_{14} = \tau_{14} + \dfrac{Q_{14}}{m_{14}} \\
\tau_{45} \ge T_{04}, \quad T_{04} = \tau_{04} + \dfrac{Q_{04}}{m_{04}} \\
\tau_{45} \ge T_{34}, \quad T_{34} = \tau_{34} + \dfrac{Q_{34}}{m_{34}} \\
T_{45=} \le T_{\Sigma}, \quad T_{45} = \tau_{45} + \dfrac{Q_{45}}{m_{45}} \\
T_{15} \le T_{\Sigma}, \quad T_{15} = \tau_{15} + \dfrac{Q_{15}}{m_{15}} \\
\tau_{01} = \tau_{02} = \tau_{03} = \tau_{04} = 0 \\
m_{01}, m_{02}, m_{03}, m_{04}, m_{14}, m_{15}, m_{23}, m_{34}, m_{45} \ \in N
\end{cases}
\tag{5.9}
$$

The problem definition (5.9) includes 14 unknown variables. Nine variables are of the m_{ij} type, i.e. the arguments of the objective function. Five variables τ_{14}, τ_{15}, τ_{23}, τ_{34}, τ_{45} define the BP scheduling.

The problem (5.9) can be solved with MATLAB Optimization Toolbox. As it belongs to the class of nonlinear programming [14], `fseminf` function has to be employed, Fig. 5.8.

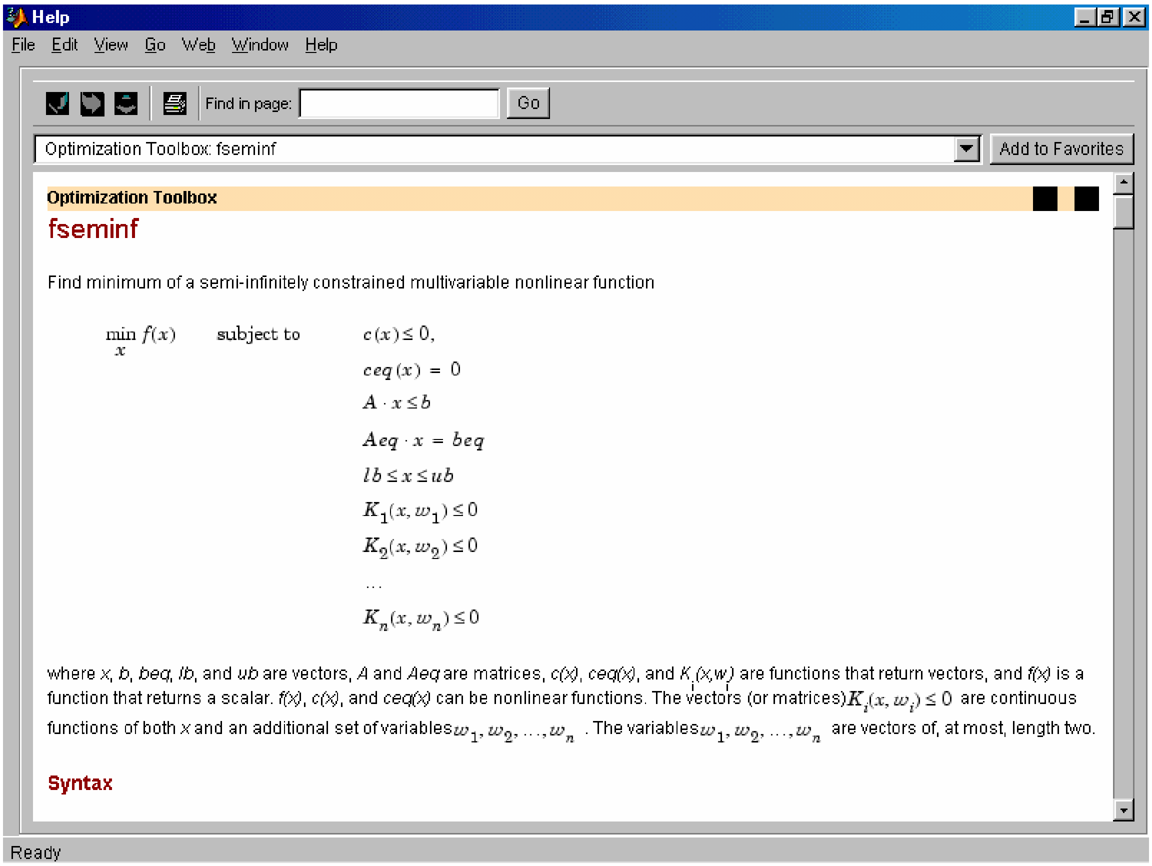


Figure 5.8. `fseminf` function of MATLAB (screenshot: MATLAB ®)

The final solution of problem (5.9) is obtained by iterations of the constrained nonlinear programming. The intermediate solutions are to be analyzed for the non-integer values of m_{ij}. These m_{ij} are to be set to the nearest integer values. Such a solution is considered as sub-optimal and it is a starting point for further improvements.

5.4. Stochastic BP models: $D_R P_{T\&C}$ and $P_R P_{T\&C}$

5.4.1. $D_R P_{T\&C}$ model

Consider the following two problem definitions of the BP scheduling for the $D_R P_{T\&C}$ model:

1. The variance of the activities duration is so small that it does not change the critical paths;
2. The variance of the durations is large enough to change the critical paths.

<u>Unchanged critical path</u>

Both dynamic and mathematical programming methods are appropriate for this case. Note that the deterministic duration time of an activity has to be substituted with the corresponding mean value, i.e. the expectation $E(t_{ij})$ of the activity ij duration t_{ij}.

According to the probability theory [11], the expectation of the BP completion time $E(T_\Sigma)$ equals the sum of the expectations of the activities[6], which belong to the critical path:

$$E(T_\Sigma) \;=\; \sum_{ij \in critical\ path} t_{ij}$$

Moreover, if all t_{ij} are independent variables, the variance $V(T_\Sigma)$ is given by:

$$V(T_\Sigma) \;=\; \sum_{ij \in\ critical\ path} V(t_{ij})\,.$$

Due to the central limit theorem [11], the sum of a large number of independent random variables subject to the same distribution will tend to the normal distribution. Hence, having expectation and variance values of an activity duration it is easy to find out the probability whether the BP completion time belongs to a given interval.

$P\{ \left| T_\Sigma - E(T_\Sigma) \right| \le \varepsilon \}$ is the probability that the absolute value of the difference between the expectation $E(T_\Sigma)$ and its statistical estimate T_Σ is no more than ε, where 2ε is the width of the confidence interval with the center at the point T_Σ.

To compute $P\{ \left| T_\Sigma - E(T_\Sigma) \right| \le \varepsilon \}$ one has to use the cumulative normal distribution tables (also known as the Laplace integral tables) [11]:

---

[6] This result holds for <u>any</u> random variables, whereas the variance (dispersion) of the sum of random variables equals the sum of the individual variances only if the variables are independent

$$P\{\left| T_\Sigma - E(T_\Sigma) \right| \le \varepsilon\} = 2\Phi(\frac{\varepsilon}{\sqrt{V(T_\Sigma)}}),$$

where

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_0^x e^{-\frac{t^2}{2}} dt$$

is the tabulated Laplace Integral.

In case there are several critical paths, one can use a stricter approach. Thus, in Fig 5.5 the graph model of the BP has two critical paths. Therefore, the probability of the BP guaranteed completion within certain period of time can be computed as follows:

$$P\{T \le T_\Sigma\} = [1 - P\{T_{crit1} > T_\Sigma\}][1 - P\{T_{crit2} > T_\Sigma\}],$$

where T_{crit1} and T_{crit2} are the lengths of the corresponding critical paths.

Note that the probabilities $P\{T_{crit1} > T_\Sigma\}$ and $P\{T_{crit2} > T_\Sigma\}$ are computed with the assumption that T_{crit1} and T_{crit2} are normally distributed.

Let us supplement our case study with the variance of the activities duration, Tab. 5.5.

Table 5.5. Variance of the activities duration

Activity	***Duration***			***Dispersion measure***		
	mean	***min***	***max***	***variance***	***st. dev.***	***halfwidth***
01	*2.00*	*0.77*	*3.23*	*0.50*	*0.71*	*1.23*
02	*1.00*	*0.05*	*1.95*	*0.30*	*0.55*	*0.95*
03	*1.00*	*0.05*	*1.95*	*0.30*	*0.55*	*0.95*
04	*2.00*	*0.77*	*3.23*	*0.50*	*0.71*	*1.23*
14	*2.00*	*0.77*	*3.23*	*0.50*	*0.71*	*1.23*
23	*3.00*	*1.55*	*4.45*	*0.70*	*0.84*	*1.45*
15	*4.00*	*2.27*	*5.73*	*1.00*	*1.00*	*1.73*
34	*1.00*	*0.05*	*1.95*	*0.30*	*0.55*	*0.95*
45	*1.00*	*0.05*	*1.95*	*0.30*	*0.55*	*0.95*

Computing the variances, we have:

$$V\{T_{crit1}\} = 0.5 + 1.0 = 1.5;$$

$$V\{T_{crit2}\} = 0.3 + 0.7 + 0.3 + 0.3 = 1.6.$$

The expectation of the BP execution time is:

$$T_\Sigma = T_{crit1} = T_{crit2} = 6.$$

Let us calculate the probability that the BP execution time is less than 8. For the normally distributed T_{crit1} and T_{crit2} we obtain:

$$P\{T_{crit1} > 8\} = 0{,}5 - \Phi(\frac{8-6}{\sqrt{1.5}}) \approx 0{,}5 - \Phi(1.633) = 0{,}5 - 0.449 = 0.051,$$

$$P\{T_{crit2} > 8\} = 0.5 - \Phi(\frac{8-6}{\sqrt{1.6}}) \approx 0{,}5 - \Phi(1.581) = 0.5 - 0.443 = 0.057.$$

Hence, $P\{T_\Sigma \le 8\} = (1 - 0.051)(1 - 0.057) = 0.895$.

5.4.1.1. BP scheduling analysis with Crystal Ball

Let us consider another approach to the problem of computing the probability that the BP is completed within certain period of time. Here we will use Crystal Ball[7], a software tool that employs the Monte-Carlo method also known as the method of statistical testing.

Let the duration of an activity be uniformly distributed as defined in Tab. 5.5. A dialogue window for setting uniformly distributed duration of the activities is shown in Fig. 5.9.

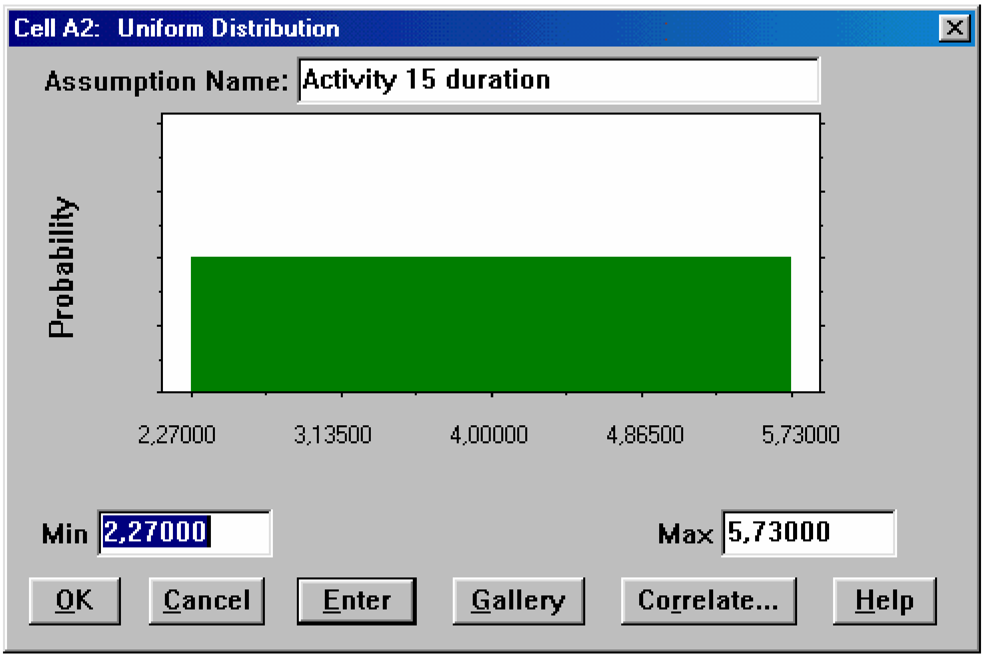


Figure 5.9. Setting the uniformly distributed duration of activities (screenshot: CrystalBall ®)

The results of two independent experiments are the distributions T_{crit1} and T_{crit2} shown in Fig. 5.10 and 5.11. Each experiment consists of 1000 runs. We can see that $P\{T_{crit1} > 8\} = 0.055$ and $P\{T_{crit2} > 8\} = 0.057$. Therefore, $P\{T_\Sigma \le 8\} = 0.945 \cdot 0.943 \approx 0.891$.

[7] *Crystal Ball* is the registered trade mark of Decisioneering Inc., [www.decisioneering.com]

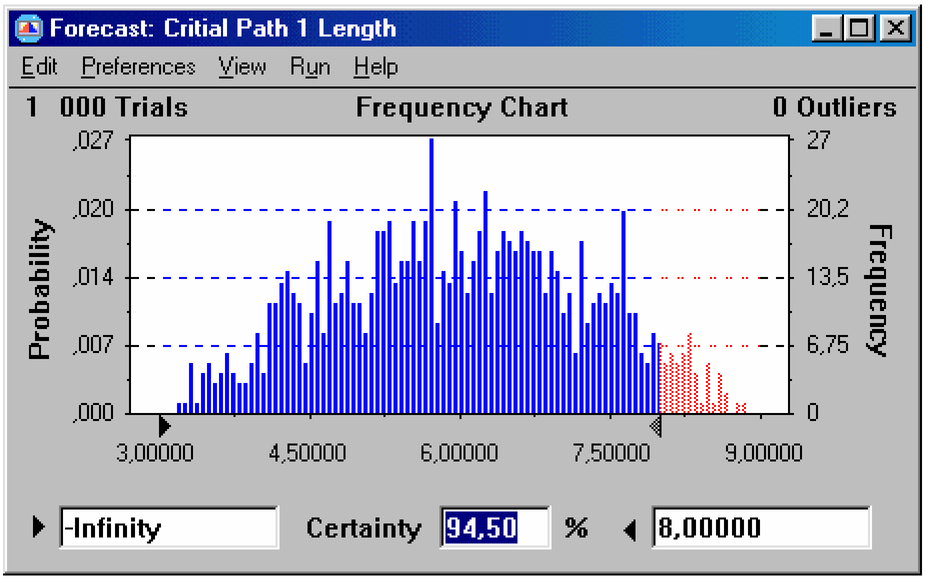


Figure 5.10. Distribution of the duration for the first critical path. The uniform distribution case (screenshot: CrystalBall ®)

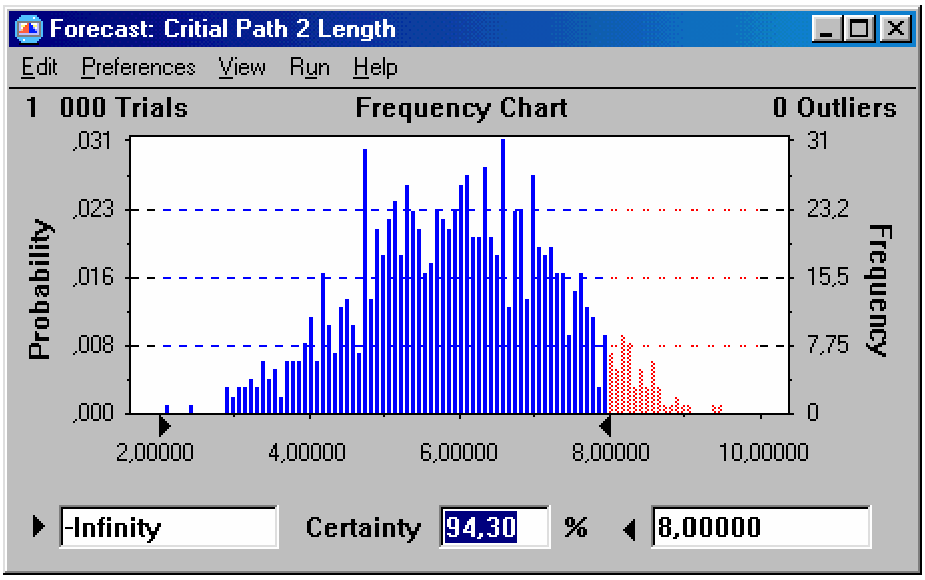


Figure 5.11. Distribution of the duration for the second critical path. The uniform distribution case (screenshot: CrystalBall ®)

Let us change the duration distributions from uniform to normal with the same mean and variance. The resulting histograms for each of the critical paths are represented in Fig. 5.13 and 5.14. According to the histograms we obtain: $P\{T_{crit1} > 8\} = 0.048$ and $P\{T_{crit2} > 8\} = 0.065$. Finally, $P\{T_{\Sigma} \le 8\} = 0.952 \cdot 0.935 \approx 0.890$. These results comply with ones obtained using the central limit theorem.

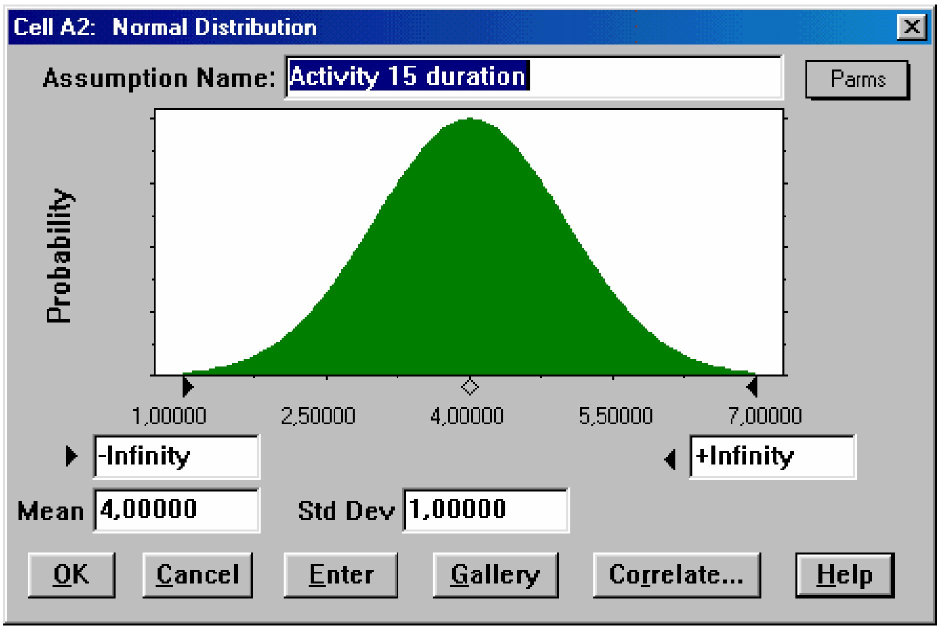


Figure 5.12. Setting the normally distributed duration of activities (screenshot: CrystalBall®)

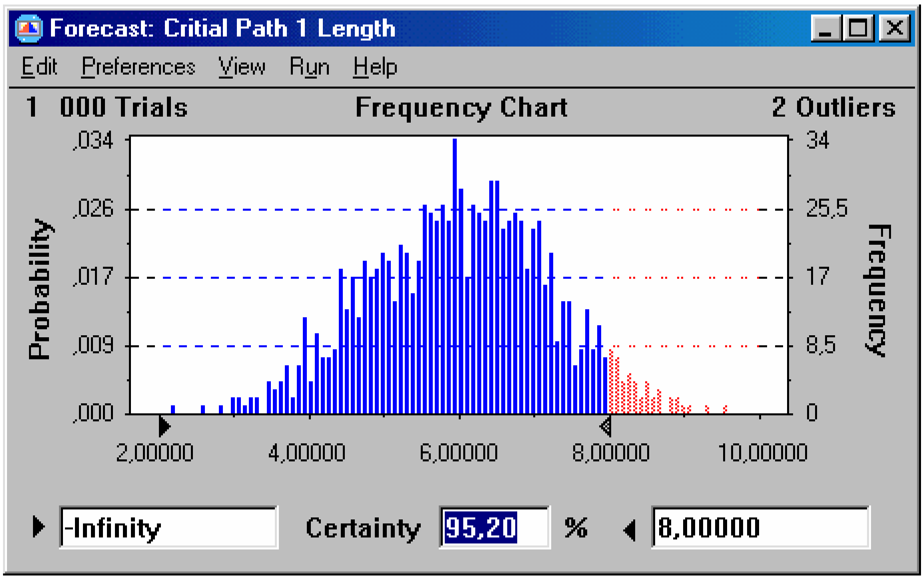


Figure 5.13. Distribution of the duration for the first critical path. The normal distribution case (screenshot: CrystalBall ®)

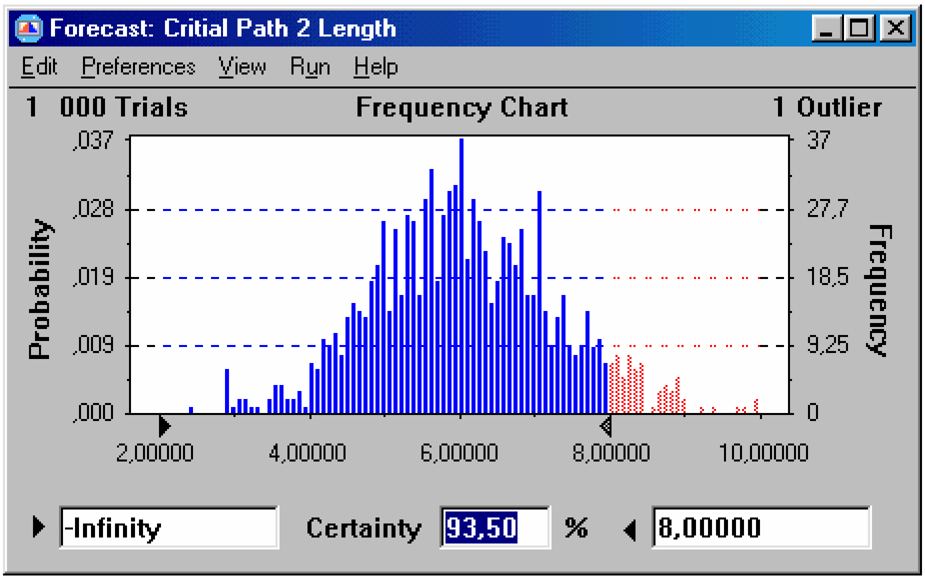


Figure 5.14. Distribution of the duration for the second critical path. The normal distribution case (screenshot: CrystalBall ®)

Let us take a closer look at stating the tasks in the built-in Excel[8] Crystal Ball environment. For our example the critical path method leads to the following expression for the BP completion time:

$$T_\Sigma = \max\{(\tau_{01}+\tau_{14}+\tau_{45}), (\tau_{04}+\tau_{45}), (\tau_{01}+\tau_{15}), (\tau_{02}+\tau_{23}+\tau_{34}+\tau_{45}), (\tau_{03}+\tau_{34}+\tau_{45})\}$$

The same function is set in an Excel cell for Crystal Ball in the following way:

T_Σ = *MAX((D2+D6+D10);(D5+D10);(D2+D10);(D3+D7+D9+D10);(D4+D9+D10)).*

Here the cells *D2 – D10* denote the stochastic arguments τ_{ij} of the *Excel* MAX function.

The simulation results in the form of the histograms representing the BP completion time are shown in Fig. 5.15 and 5.16 for the uniform and the normal distribution of the activities duration. We can see that the probability of the BP completing in time almost does not depend on the type of the activities duration distribution. Thus, for the uniform distribution we have: $P\{T_\Sigma \le 8\} \approx 0.942$, though for the normal distribution we obtain almost the same result: $P\{T_\Sigma \le 8\} \approx 0.937$.

[8] *Excel* is the trademark of Microsoft Corporation, [www.microsoft.com]

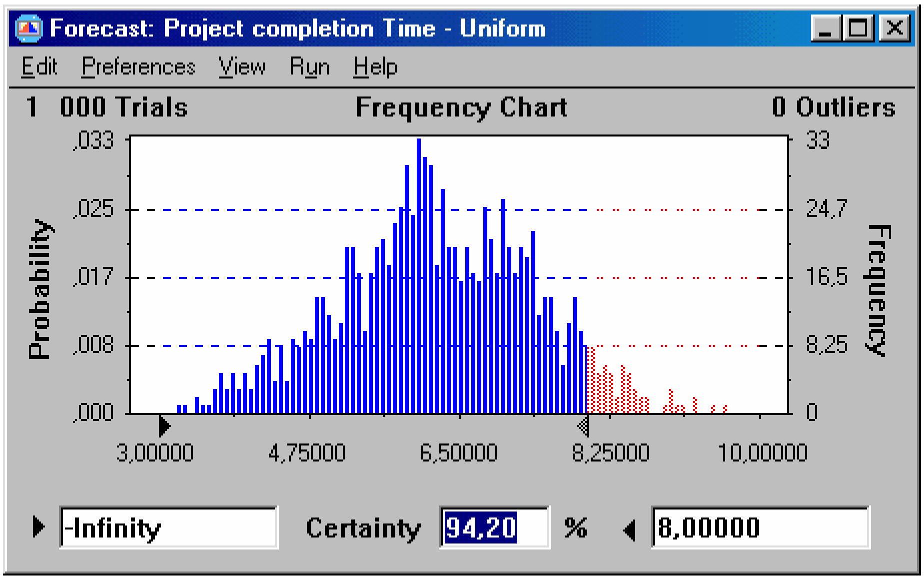


Figure 5.15. Simulation results for the uniform distribution (screenshot: CrystalBall®)

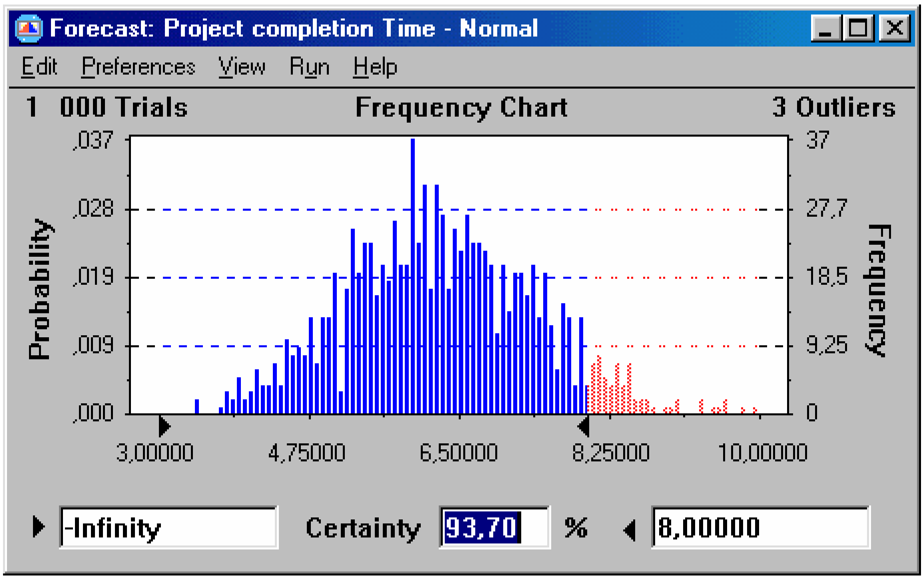


Figure 5.16. Simulation results for the normal distribution (screenshot: CrystalBall ®)

5.4.2. Stochastic $P_R P_{T\&C}$ models with an account for the resource utilization

Let us return to the BP IDEF3 model and complement it with a resource reference using the so called Reference Objects of the IDEF3 syntax. For our case study, when the resources are taken into account, the initial IDEF3 model (Fig. 5.3) becomes one shown in Fig. 5.17. Each activity is assigned to the one of the BP resources D1 – D4.

Suppose that the duration of an activity means the net time of the activity processing. For that kind of the BP model, we consider a resource as a serving system with a queue of “customers”. A “customer” has an “order” for an activity execution.

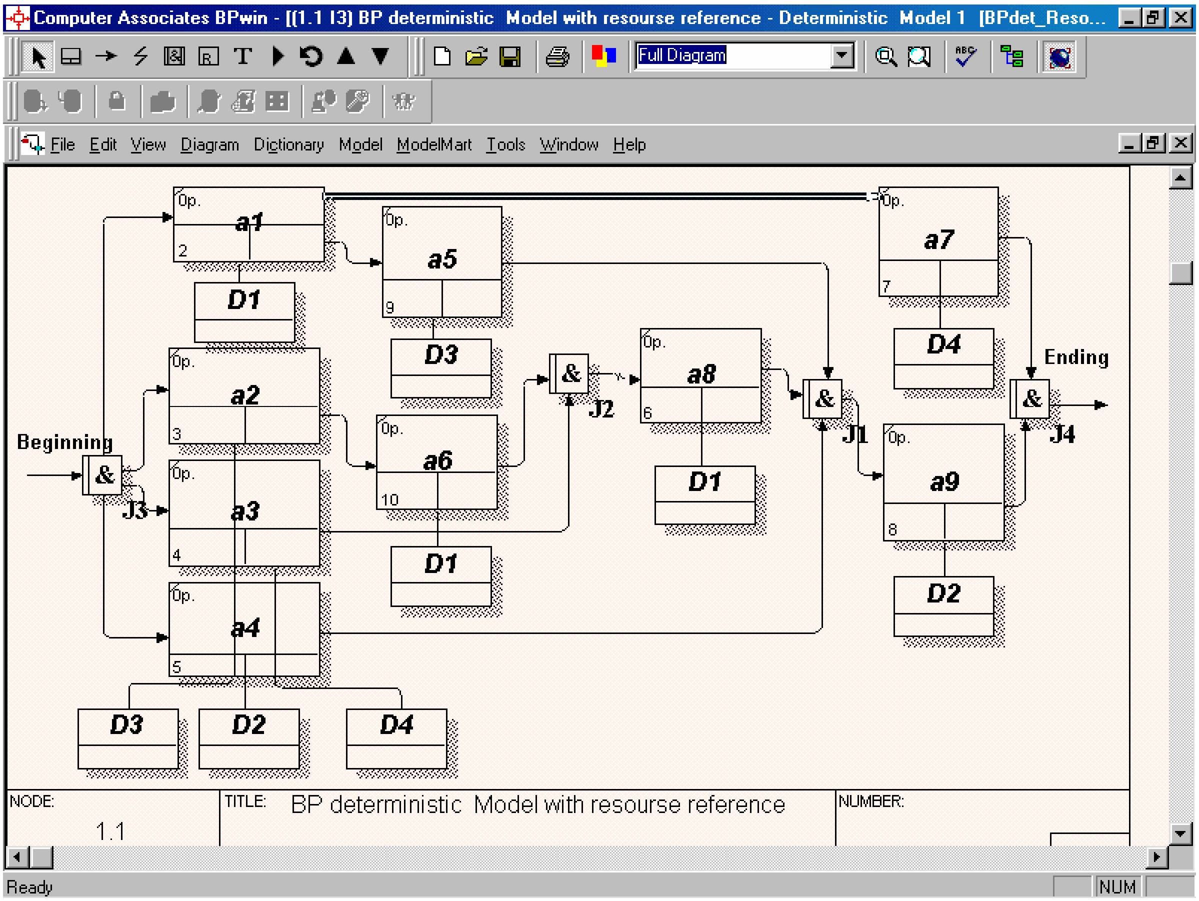


Figure 5.17. IDEF3 model with resources (screenshot: BPwin ®)

Having this in mind, we conclude that the waiting time in queues has to be taken into account when calculating the BP completion time. This is a good reason for applying the queueing theory as a basis of the BP analytical modeling.

5.4.2.1. BP $P_R P_{T\&C}$ model in the form of the queueing network

Fundamentals

The inhomogeneous networks with infinite queues are suitable for BP modeling. A queueing network is a network of queues. In other words, a queueing network is a set of nodes, where a node is a serving system with a queue of customers at its entrance. Once entering a network, the customers of the system are routed from one node to another receiving service at each node and, finally, leaving the network. An open network receives customers from an outside independent source. A leaving customer is interpreted as the order which has been fulfilled or a client which has been served.

For more information on the queueing theory refer to [5, 2].

M/G/1 – inhomogeneous customer flow at the entrance

Incoming customers are inhomogeneous because they arrive at different rates and need different time of serving.

In terms of the queuing theory, a serving system has an inhomogeneous customer incoming flow summing the flows with the rates $\lambda_1, \lambda_2, \ldots \lambda_n$. Each of them is the flow of Poisson's (exponential) arrivals. The customers from the flow i are served according to the *general* (i.e. arbitrary) distribution with the mean $\overline{x_i}$ and the variance V_i, see Fig. 5.18.

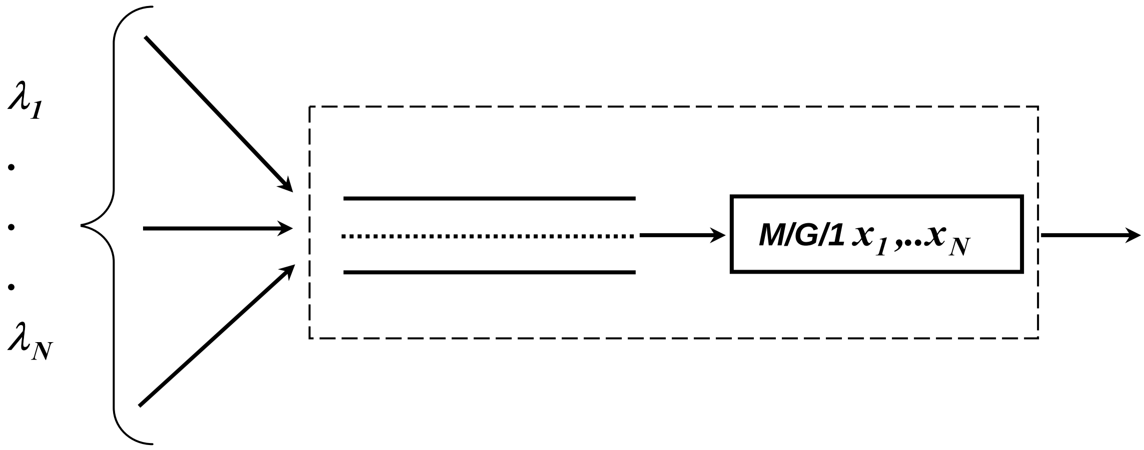


Figure 5.18. Model of a serving system

It is known [6] that the mean waiting time in the queue, w_Q, is the same for each incoming flow of customers and is given by:

$$w_Q = \frac{\overline{T_0}}{1-R}, \tag{5.10}$$

$$\overline{T_0} = \frac{1}{2}\sum_{i=1}^{N} \lambda_i \left[\left(\overline{x_i}\right)^2 + V_i \right], \tag{5.11}$$

$$R = \sum_{i=1}^{N} \rho_i, \quad \rho_i = \overline{x_i}\lambda_i, \tag{5.12}$$

where $\overline{T_0}$ denotes the mean remaining service time for the customer occupying the server. The parameter R has a sense of the utilization of the server loaded with the inhomogeneous customer flow. The condition $R_j < 1$ is required for the serving system to be stable.

According to the BP modeling application, we consider each resource (the reference object of the IDEF3 model, Fig. 5.17) as a single-channel serving system M/G/1. Since a resource object is shared by several activities (D1, D2, D3, and D4 in Fig. 5.17), the serving system has customers of various classes at the entrance. The number of classes is equal to the number of the activities within the BP IDEF3 model.

Inhomogeneous open network of single-server queues

While the rigorous coverage of the queueing networks is beyond the scope of the book, the main concepts are outlined below.

Let us consider a stable system with an infinite calling population[9] and no limit on the system capacity. We can obtain the following results [2]:

1. Provided no customers are created or destroyed in the queue, on the average (i.e. over the long run) the queue outcoming rate is the same as the incoming one;

2. The customers are able to change their class by routing from one queue to another;

3. If the customers of class l arrive to queue i at rate λ_{il} and the fraction $0 \le p_{il,jk} \le 1$ of them are routed to queue j upon their transformation to class k, then on the average the arrival rate of the class k from the queue i to the queue j is $\lambda_{jk} = \lambda_{il} p_{il,jk}$;

4. The overall arrival rate of the class l into the queue j, λ_{jl}, is the sum of the arrival rate from all sources. If the customers arrive from the outside of the network at the rate λ_0, then:

$$\lambda_{jk} = \lambda_0 p_{o,jk} + \sum_{i=1}^{M} \sum_{l=1}^{N} \lambda_{il} p_{il,jk} \quad , \quad j = \overline{1, M} \, , k = \overline{1, N} \, , \tag{5.13}$$

where

M is equal to the number of queues (i.e. nodes) in the network,

0 denotes the “outside” source of the incoming customers,

N is equal to the number of customer classes;

4. The combination il of the node number and the class number is referred to as the task il being assigned to the node i. Thus, $p_{il,jk}$ is interpreted as the probability of the transition from the task il to the task jk;

5. If the queue j has a server working at rate μ_{jl} for the l-class customers, the average utilization of each server is:

$$R_j = \sum_{l=1}^{N} \rho_{jl}, \quad \rho_{jl} = \frac{\lambda_{jl}}{\mu_{jl}} \tag{5.14}$$

and $R_j < 1$ is required for the queue to be stable.

In conclusion, let us provide a template for the inhomogeneous open network of the single server queues. The network in question can be defined by the following parameters:

[9] A set of potential customers

$$\left\{\lambda_o; M; \pi = \{p_{il,jk}\}\ i,j = \overline{0,M},\ \ l,k = \overline{1,N};\ \{\mu_{il}\},\ i = \overline{1,M},\ \ l = \overline{1,N}\right\},$$

(5.15)

where

λ_0 is the customer input flow,

M is the number of the network nodes,

N is the number of the customer classes,

μ_{il} is the serving rate of the node i for the customer class l.

5.4.2.2. Representing the BP IDEF3 model as the queuing network

In this section we will show how to represent the BP IDEF3 model in the form of the queueing network. We will use the pattern of the inhomogeneous queuing network with the customers grouped into a number of classes. The conversion procedure of the BP model into the queuing network is described at length in [7].

Converting the BP IDEF3 model to the queueing network form comprises three steps, which are explained below and illustrated using our case problem (see Fig. 5.13).

<u>Steps in creating the queueing network</u>

1. Defining the queueing network nodes

Each serving system (D1 through D4) is considered as a network node. Hence, M is equal to four.

2. Defining the classes of customers

Each node is assigned to some activity (class), Tab. 5.6, that requires various serving times. From this point of view, the nodes have the inhomogeneous customer flow at the entrance, see Fig. 5.15. Tab. 5.6 shows that 3 nodes have 2 classes of flow and the first node has 3 classes of flow at the entrance.

The total number N of the customer classes is equal to 9 (the number of the activities).

3. Creating the transmission matrix

Suppose that the incoming customer arrival rate is measured in number-of-customers-per-time-unit and the rate equals λ_0.

Table 5.6. Assigning activities to serving systems

Node i	***Probability of assigning the activity l to the node i***								
	l								
	1	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
1	1					1		1	
2				1					1
3		1			1				
4			1				1		

The transmission matrix π, which is given in Tab. 5.7, defines the routing of the customers in the network. The matrix π is derived from the BP IDEF3 model in accordance with Tab. 5.6: $\pi = \{p_{il,jk}\}$, where $p_{il,jk}$, $(i, j = \overline{0,4}\ ;\ l, k = \overline{1,9})$ is the probability of transmitting a customer to the task jk after completing the task il. In other words, $p_{il,jk}$ is the probability that a customer of the class l leaves the node i, becomes a customer of the class k and enters the node j. Here, the index jk means that the activity k is executed by the node j.

Table 5.7. Transmission matrix π

	0	***11***	***16***	***18***	***24***	***29***	***32***	***35***	***43***	***47***
0		1			1		1		1	
11								1		
16				1						
18						1#				
24						1#				
29	1									
32			1							
35						1#				
43				1						
47	1									

The list of all tasks processed in the context of the BP is the following (see Fig. 5.17): *{11, 16, 18, 24, 29, 32, 35, 43, 47}*.

The analysis of the matrix π reveals the following two essential properties:

1. The matrix π is not stochastic. For a stochastic matrix the sum of the row elements must be equal one. The sum being greater than one implies the customer "splitting" effect;
2. The assembling symbol # is not typical for the linear queueing networks. It means the customer "assembling" (merging) effect. If this is the case, placing the symbol # in column 29 of the matrix π means that node 2 is able to start executing activity 9 (i.e. task 29) only after tasks 18, 24, and 35 have been completed.

Estimating the BP execution time and costs

Let us introduce the relative rate of the customers of the class *k* at the entrance of the node *j*: $\alpha_{jk} = \lambda_{jk} / \lambda_0$ Then, the equations of the network flow balance for α_{jk}, $j = \overline{1, M}$, $k = \overline{1, N}$ are defined as follows:

$$\alpha_{jk} = \alpha_0 p_{0,jk} + \sum_{i=1}^{M} \sum_{k=1}^{N} \alpha_{il} p_{il,jk}; i, j = \overline{1, M}; \; i, k = \overline{1, N}; \; jk \notin \# \tag{5.16}$$

$$\alpha_{jk} = \min_{(il)} \{ \alpha_{il} p_{il,jk} \}; \quad i, j = \overline{1, M}; \; l, k = \overline{1, N}; \; jk \in \# \;, \tag{5.17}$$

where $\alpha_0 = 1$ and $\lambda_{jk} = \lambda_0 \alpha_{jk}$.

The set of equations, which is composed according to (5.16) and (5.17) for the case problem in question, is given below:

$$\begin{cases} \lambda_{11} = \lambda_{24} = \lambda_{32} = \lambda_{43} = \lambda_0 \\ \lambda_{16} = \lambda_{18} \\ \lambda_{29} = \min\{\lambda_{18}, \lambda_{24}, \lambda_{35}\}, \qquad 29 \in \# \\ \lambda_{18} = \lambda_{29} \\ \lambda_{29} = \lambda_0 \\ \lambda_{32} = \lambda_{16} \\ \lambda_{43} = \lambda_{18} \\ \lambda_{47} = \lambda_0 \end{cases} \tag{5.18}$$

Solving (5.18), we obtain $\lambda_{ik} = \lambda_0, \quad ik = \{11, 16, 18, 24, 29, 32, 35, 43, 47\}$.

Estimating the BP execution time for the BP model in the form of the linear queueing system

If there are no &-junctions in the BP IDEF3 model, the matrix π does not include the assembling symbols # and it has no rows whose sum of elements exceeds one (no customer splitting). In this case the queuing system is linear open one and the BP execution time can be calculated as follows:

First, the set $\{\alpha_{il}\}$ of the relative rates of the inhomogeneous customers at the entrance of each node of the system is calculated using (5.16);

Second, the total staying time set $\{w_{il}\}$ is computed for each node using (5.10):

$$w_{il} = w_{Qil} + \frac{1}{\mu_{il}} \tag{5.19}$$

Third, the BP execution time is computed according to the theory of the inhomogeneous linear open queueing systems as:

$$E(T_{\Sigma}) = \sum_{i=1}^{M} \sum_{l=1}^{N} \alpha_{il} w_{il} \ , \tag{5.20}$$

where α_{il} is the relative rate of the customers of the class l at the entrance of the node i node, w_{il} is the total residence time in the node i of a customer of the class l (i.e. the activity l execution time in the node i).

Estimating the BP execution time for the BP model in the form of the nonlinear queueing system

If the flow rates at the entrance of every node are known, one can calculate the customer waiting time using (5.10).

The parameters of the activities serving time distributions for the problem in question are given in Tab. 5.8.

Table 5.8. Serving time parameters

			Activity l								
			1	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
Node i	***1***	$\bar{x}$	2					3		1	
		Std. Dev.	0.71					0.84		0.55	
	2	$\bar{x}$				2					1
		Std. Dev.				0.71					0.55
	3	$\bar{x}$		1			2				
		Std. Dev.		0.55			0.71				
	4	$\bar{x}$			1				4		
		Std. Dev.			0.55				1		

The total execution time w_{il} for the activity l with the account for waiting in the queue is defined as:

$$w_{il} = \overline{x_{il}} + w_{Qil}\,, \qquad (5.21)$$

where w_{Qil} is calculated using (5.10) – (5.12).

Let $\lambda_0 = 0.1$, then using (5.18) yields the results given in Tab. 5.9.

Table 5.9. Total activity execution time

			Activity l								
			1	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
Node i	***1***	$\overline{x}$	2					3		1	
		w	3.94					4.94		2.94	
	2	$\overline{x}$				2					1
		w				2.41					1.41
	3	$\overline{x}$		1			2				
		w		1.41			1.41				
	4	$\overline{x}$			1				4		
		w			2.83				5.83		

In the nonlinear case, expression (5.20) no longer holds for estimating the BP execution time. Applying the critical path method to our data leads to the following estimate of the mean BP execution time:

$$E(T_\Sigma) = \max\{(w_1 + w_5 + w_9), (w_4 + w_9), (w_1 + w_7), \\ , (w_2 + w_6 + w_8 + w_9), (w_3 + w_8 + w_9)\} \approx 10.7$$

Another way to obtain an estimate of $E(T_\Sigma)$ is the BP simulation, which will be considered in the next chapters.

The logic conditions & in the structure of the BP may lead to the nonordinarity of the customer flows. If a flow is not ordinary, the customers can arrive in packets (two or more customers at a time). As a result, the waiting time and the total BP execution time significantly increase. This phenomenon cannot be taken into account by the queueing networks approach as it assumes the ordinarity of flows.

These considerations can explain the discrepancy between the analytical and simulation results. The analytical estimate of the BP execution time is $T_\Sigma = 10.7$, whereas the GPSS simulation gives $T_\Sigma = 16.6$ (see Appendix A, program 3). As a consequence, the above-described analytical method, which is based on the nonlinear queueing networks, leads to the lower bound of the BP execution time estimate.

BP costing for the BP queueing model

Suppose that the costs C_{il} of the activity l executed by the node i are known for all tasks $\{il\}$. Then, the BP execution costs are computed as follows (regardless of the queueing BP model being linear or nonlinear):

$$E(T_\Sigma) = \sum_{i=1}^{M} \sum_{l=1}^{N} \alpha_{il} C_{il} \ . \tag{5.22}$$

CHAPTER 6. BP SIMULATION WITH GPSS

6.1. GPSS fundamentals

This chapter is mainly based on the material from [8, ch. 4]. Comprehensive examples and detailed explanations can be found in [9].

The General Purpose Simulation System (GPSS), which was initially released by IBM in 1961, is one of the most popular packages for discrete event simulation. One of its implementations is GPSS World[10]. GPSS World Student Version is available on the Web at www.minutemansoftware.com.

6.1.1. System representation

GPSS takes the process interaction approach to organize the simulated system behavior. Processes (temporary entities), which are called transactions, interact with each other and with the permanent entities called facilities and storages.

The GPSS *transactions* are the active temporary entities. The GPSS *facilities* and *storages* are the passive permanent entities. For example, each client of a serving system is represented by a transaction and each server represents one unit of storage or one facility.

GPSS assigns a record with attribute fields called *parameters* to each transaction. While processing a simulation run, GPSS assigns numerical values to these attributes using it for decision taking. Facilities and storages have attributes as well.

6.1.2. GPSS language

Transaction generation

GENERATE Block creates transactions as follows (some operands are optional):

GENERATE A, B, C, D, E

A is a mean inter generation time;

B is the inter generation time half-range;

C is the start delay time;

D is the creation limit;

E is the priority level.

---

[10] *GPSS World* is the registered trademark of Minuteman Software, [www.minutemansoftware.com]

Example:

GENERATE 0.1

This is the simplest way to use the **GENERATE** Block. This Block causes a transaction to enter the simulation every tenth of a time unit.

Service

Facilities and storages are dealt with the following two pairs of statements:

SEIZE A and **RELEASE A. SEIZE A** allows a transaction to occupy the facility A and **RELEASE A** allows it to release the facility A.

ENTER A and **LEAVE A** perform the same functions with respect to the storage A.

The capacity of the storage A has to be defined before the block statement **ENTER A** appears in the simulation program.

Statistics

Statistics are gathered by means of the **QUEUE, DEPART, MARK,** and **TABULATE** statements.

If a transaction enters the **QUEUE A** statement, one unit is added to the queue A. As soon as a transaction enters **DEPART A** statement, the latter reduces the queue A by one unit. That is the way to keep track of the queue length of every serving system reflected in the simulation program.

The **MARK** and **TABULATE** statements provide a way of gathering data on the transit time of a transaction through the system. The **MARK** statement causes the transaction arrival time at the **MARK** block to be recorded as the contents of the **M1** system numerical attribute. When GPSS subsequently encounters **TABULATE A**, it computes the elapsed time between the transaction's arrival at the **MARK** block and its arrival at the **TABULATE** block and records this time to **TABLE A**. In this way the transit time between any two blocks in the program can be determined by an appropriate placement of the **MARK** and **TABULATE** blocks.

Logical Testing

The **TEST** statement checks for the following conditions between a pair of the system numerical attributes according to Tab. 6.1. Example:

TEST G C1, 70000

In this example the active transaction enters the **TEST** block if the relative system clock value is greater than 70000. Otherwise, the transaction is blocked until the test is true.

Table 6.1. Parameters of the TEST statement

Condition Operator	***Explanation***
G	Greater than
GE	Greater than or equal
E	Equal
NE	Not equal
LE	Less than or equal
L	Less than

The Timing Routine

Generally speaking, GPSS simulates concurrent processes with many events occurring simultaneously. The GPSS mechanism for advancing simulation time has to guarantee that all events occur in the correct chronological order.

Transactions are temporarily bound to other entities by occupying linked lists called chains. To understand the timing routine function, it is essential to consider the following two chains: the Current Events Chain (CEC) and the Future Events Chain (FEC).

The CEC is a linked list of the ready transactions, which have blocks to be run before the simulation time advances. Transactions are linked in the CEC according to their priority levels.

The FEC is a time-ordered chain holding transactions, which must wait for some later moment of the simulation time.

Suppose that a scheduled event has occurred and the control has returned to the timing routine. The timing routine scans the CEC until it finds a linked transaction whose scan indicator is active. GPSS then attempts to continue execution of this transaction. If it is successful, the control passes to the transaction.

If the transaction enters an **ADVANCE** statement, the transaction becomes linked to the FEC. When execution is blocked in any other way, it is linked to the CEC with its scan indicator inactive. The control passes back to the timing routine, which continues its scan of the CEC. If this scan finds no transaction on the CEC that can continue execution, the timing routine advances the simulation time to the desired execution time of the first transaction on the FEC. The routine also removes all transactions scheduled for that time from the FEC and links them to the CEC according to their priority levels. An example of the timing routine interaction with the CEC and FEC is shown in Fig. 6.1.

CEC for the Simulation Time equal to 10

Xact No	*Priority Level*	*Blocked?*	*Scan indicator*
5	10	yes	Passive
28	9	no	Active
3	0	no	Passive
6	0	no	Passive
19	0	yes	Passive

CEC for the Simulation Time equal to 10 –
No transactions to be executed at this Simulation Time

Xact No	*Priority Level*	*Blocked?*	*Scan indicator*
5	10	yes	Passive
28	9	yes	Passive
3	0	yes	Passive
6	0	yes	Passive
19	0	yes	Passive

FEC for the Simulation Time equal to 10

Xact No	*Future execution time*
2	20
40	20
4	30
1	45
28	50

CEC for the Simulation Time equal to 20

Xact No	*Priority Level*	*Blocked?*	*Scan indicator*
40	12	no	Active
5	10	yes	Passive
3	0	no	Passive
2	0	no	Passive
6	0	no	Passive
19	0	yes	Passive

FEC for the Simulation Time equal to 20

Xact No	*Future execution time*
4	30
1	45
28	50

Figure 6.1. Illustration of the timing routine

<u>Typical GPSS segment of the BP modeling program</u>

Considering the BP modeling problem in terms of the queueing theory we can say that a queueing system fulfils the incoming orders of the BP instances execution. Since a serving system is a basic element of a queuing system, then the problem turns out to be the task of modeling a serving system. The latter accomplishes such activities as queuing, seizing a server, service processing, and releasing a server. Each of these activities has its mapping into the appropriate GPSS statement. A typical segment of the BP GPSS program shown below.

```
*******************************************************
*       BP modeling segment of a queuing system node     *
*******************************************************

QUEUE               Node1; a customer enters the queue
SEIZE               Node1; the customer occupies the server
DEPART              Node1; the customer leaves the queue
ADVANCE     10      ; the customer is being served for 10 time units
RELEASE             Node1; the customer releases the server
```

6.2. BP IDEF3 model interpretation for the GPSS modeling

<u>Interpreting the activity start as the arrival of a transaction</u>

GPSS model is the process-oriented description of the system behavior. In other words, it is a formalized presentation of the temporal operations sequence as the content of the business process. In contrast, the BP IDEF3 model is less formal because it is closer to the conceptual system description. Therefore, converting the BP IDEF3 model to the GPSS program may be ambiguous. The nature of the ambiguity arises from the following factors. The IDEF3 and the GPSS models have different expressive power. The formalism of the descriptive IDEF3 model does not assume a strict reflection of the information about the organization of the system in question. The basic element of the IDEF3 model is an activity, i.e. a box with inputs and outputs. The fact of the activity start driven by some event (arrived client, received order etc.) has to be associated with the arrival of a transaction *since the GPSS statements cannot be executed unless there are no arriving transactions.*

<u>Arrow types</u>

Each input causes two ways of the activity initialization essential to the GPSS-model constructing. That is why it makes sense to distinguish the following two arrow types.

<u>The first arrow type.</u> Activating this arrow results in a condition that appears and remains true until it is not cancelled (a sort of logic key).

<u>The second arrow type.</u> Activating this arrow results in creation of a single (or multiple in case of a cycle) activity executing procedure.

Each of the two arrows can be Input or Output.

The Reference Object pointing the resource to accomplish an activity substitutes the Mechanism arrow of the IDEF0 standard. This sort of a Reference Object is associated with such a GPSS concepts as a Facility or a Storage reflected in the GPSS program by means of the pairs of statements **SEIZE** – **RELEASE** or **ENTER** – **LEAVE**.

If there are second type Input arrows to the activity, then all of them have to be initialized for this activity to begin. This fact has to be reflected in the BP GPSS model as all input transactions have to be terminated except for one transacton.

Transaction parameters as a customer specification

GPSS assigns a record with attribute fields (*parameters*) to each transaction. This helps to distinguish transactions from each other. The **ASSIGN** statement is used to place or modify some value in a transaction *parameter*.

Modeling concurrent processes

Concurrent branches of the BP are accomplished in its GPSS model using the block statements **SPLIT** and **ASSEMBLE**.

The **SPLIT** block allows the incoming transactions to born child transactions and to introduce them into the GPSS model. The transaction that executes a **SPLIT** block is named *parent* and the child transactions are named *children* or *descendants*.

The **ASSEMBLE** block unifies certain transactions, which belong to the same family of a single transaction. This block resembles an assembly line.

BP routing in the GPSS program

Another typical situation of converting the IDEF3 description to the GPSS program appears when there are two or more output arrows with a probabilistic transfer to another activity. The **TRANSFER** block, which has several operating modes, including the probabilistic one, is employed in this case.

Stages of creating the BP GPSS model

In brief, the technique of the BP simulation in the GPSS can be represented as a number of steps resulting in creating the BP GPSS program frame. The starting point is the context IDEF3 diagram. Then, the following steps have to be performed:

- Place GENERATE block at the beginning of the program;
- Place TERMINATE block at the bottom of the program;
- Reserve the program space between the head and the bottom for the activities to be defined in the form of a GPSS segment while processing the context activity.

- Define the parameters of activities processing at the top of the program.

Hence, we obtain the program frame open for the further elaboration according to the nature of the BP IDEF3 model at hand:

Parameters Definition
GENERATE
UNDISCLOSED
PROGRAM
BODY
TERMINATE

Using Bpwin in conjuction with GPSS

It is important to use all service facilities of the BPwin toolkit to simplify the creating of the BP GPSS program. For instance, each activity in the Bpwin IDEF3 model has many fields suitable for placing the text of the corresponding GPSS program segment. Each program segment of that sort is associated with a leaf node activity, which is located at the lowest level of the activity tree.

Converting the hierarchy of the BP IDEF3 model into the structure of the GPSS program

Each activity after being decomposed can be represented as a set of its leaf node activities. Obviously, it is possible to make a leaf node activity simple enough for being easily converted into a GPSS segment.

When all leaf node activities are supplied with the corresponding GPSS segments, one only has to place the segments to the right places. This procedure is rather straightforward because we can associate each text segment with its activity location in the IDEF3 model.

The steps of the GPSS BP modeling are demonstrated in Ch. 9 using the example of modeling an enterprise.

CHAPTER 7. OUTPUT ANALYSIS

7.1. Precision of the simulation results

The output analysis concerns the data generated by a simulation. Its purpose is to predict the values of the BP parameter such as execution time, costs etc.

If the performance is measured by parameter Θ, the result of a set of simulation experiments will be an estimate $\hat{\Theta}$ of Θ. The precision of the estimate $\hat{\Theta}$ can be measured by the variance (or standard deviation) of $\hat{\Theta}$.

The purpose of the statistical analysis is to estimate this variance and to determine the number of the observations required to achieve the desired precision of the estimate.

7.1.1. Point estimation

A point estimate of Θ based on the data sample $\{x_1, x_2, \ldots, x_N\}$ can be defined by:

$$\hat{\Theta}_N = \frac{1}{N}\sum_{i=1}^{N} x_i , \qquad (7.1)$$

where $\hat{\Theta}_N$ is the sample mean (arithmetic mean) based on the sample of size N.

A point estimate $\hat{\Theta}_N$ is said to be **consistent** for Θ if $\hat{\Theta}_N \to \Theta \quad as \quad N \to \infty$.

A point estimate $\hat{\Theta}_N$ is said to be **unbiased** for Θ if its expected value is Θ, i.e. $E(\hat{\Theta}_N) = \Theta$.

A point estimate $\hat{\Theta}$ is said to be **efficient** for Θ if $E\{(\hat{\Theta} - \Theta)^2\} = \min$ among all known estimators of Θ.

If the sample $\{x_1, x_2, \ldots, x_N\}$ belongs to the normal distribution, the point estimate (7.1) is consistent, unbiased and efficient.

7.1.2. Interval estimation

The precision of the point estimate $\hat{\Theta}$ can be given in terms of the **confidence interval** and the **confidence probability**:

$$P\{\left|\hat{\Theta} - \Theta\right| < \varepsilon\} = 1 - \alpha . \qquad (7.2)$$

Equation (7.2) means that the probability of the confidence interval $[\hat{\Theta} - \varepsilon,\ \hat{\Theta} + \varepsilon]$ with the stochastic bounds to cover the unknown parameter Θ is equal to the confidence level $(1-\alpha)$. The less the halfwidth ε is, subject to the given confidence probability, the more precise the estimate $\hat{\Theta}$ is.

7.1.3. Estimating the confidence interval

Let $\hat{\Theta}$ be a sample mean computed by (7.1). We also assume that the sample $\{x_1, x_2, \ldots, x_N\}$ is normally distributed[11]. Then the confidence interval estimate is computed through the following steps:

1. Compute the sample variance S^2 as:

$$S^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \hat{\Theta}_N)^2 \tag{7.3}$$

2. Calculate the point $t_{\alpha/2, f}$ of the t distribution (Student's distribution)[12] with f degrees of freedom[13]. Here, $t_{\alpha/2, f}$ is the solution of the equation $P\{t < t_{\alpha/2, f}\} = 1 - \alpha/2$. The shape of Student's distribution is shown in Fig. 7.1. The number of the degrees of freedom f equals $N-1$. For instance, if $\alpha = 0.1$ and $f = N-1 = 29$, then $t_{0.05,29} \approx 1.70$ [11].

3. Define the halfwidth ε of the confidence interval by:

$$\varepsilon = t_{\alpha/2}\frac{S}{\sqrt{N}}, \tag{7.4}$$

where S is the standard deviation estimate computed according to (7.3).

4. Compute the confidence interval: $[\hat{\Theta} - \varepsilon,\ \hat{\Theta} + \varepsilon]$.

---

[11] Otherwise, if the distribution is not normal, due to the central limit theorem [11], the distribution of the estimate $\hat{\Theta}$ tends to the normal one as the size of the sample increases (N>15-20).

[12] Student's curve approachs a normal distribution curve as N increases (N>20)

[13] The estimate $\hat{\Theta}$ follows Student's distribution if the variance is unknown (the sample variance (7.3) is used instead)

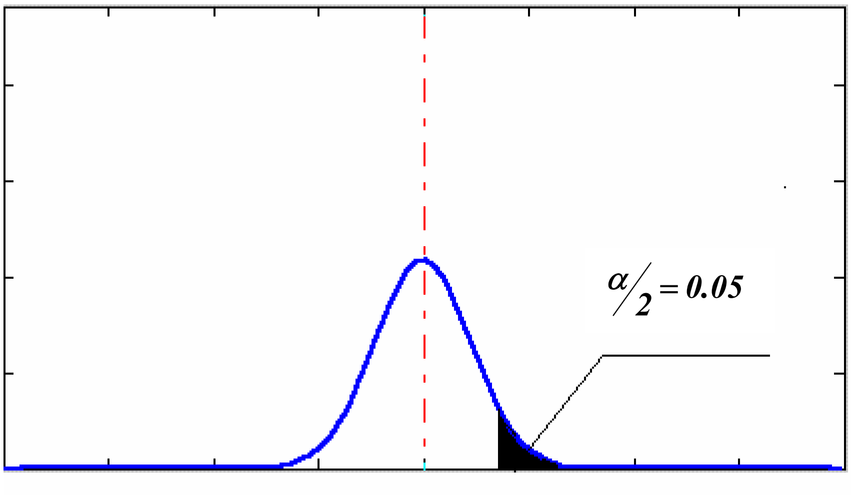


Figure. 7.1. Student's distribution

7.2. Selecting the sample size

Assume the estimate $\hat{\Theta}$ is efficient: $\hat{\Theta}_N \to \Theta$ *as* $N \to \infty$. Then, it is easy to see that the sample size N, the confidence probability (level) and the width of the confidence interval depend on each other [11]:

$$1.\ N \geq F_1(\varepsilon, 1-\alpha) = \frac{S^2\varepsilon^2}{t^2_{\alpha/2}} \tag{7.5}$$

$$2.\ \varepsilon = F_2(N, 1-\alpha) = t_{\alpha/2}\frac{S}{\sqrt{N}} \tag{7.6}$$

$$3.\ t_{\alpha/2} = F_3(N, \varepsilon) = \frac{\varepsilon\sqrt{N}}{S} \tag{7.7}$$

Equation (7.5) helps to choose the sample size N, which provides the required estimate precision, measured with the halfwidth ε. Equations (7.6) and (7.5) allow to define the parameters of the confidence interval for the given sample size N.

CHAPTER 8. BP SIMULATION WITH THE TIMED PETRI NETS

8.1. Petri nets

The Petri nets are used primarily for studying the dynamic concurrent behavior of the network-based systems. A comprehensive treatment of the Petri nets can be found in [10].

The Petri nets have the same expressive power as the finite state machines. However, an important feature of the Petri nets is that two compatible nets can be composed into one composite net by connecting the outputs of the first net with the inputs of the second one. Such a combination in a state machine is more complex.

A Petri Net is a 4-tuple defined as follows: ***C = (P, T, I, O)***, where

$P = \{p_1, p_2, \ldots, p_n\}$ is a finite set of places (also known as conditions);

$T = \{t_1, t_2, \ldots, t_m\}$ is a finite set of transitions;

$I: \ T \to P^r$ is the input function specifying a function from a single transition to a bag[14] of places;

$O: \ T \to P^q$ is the output;

Marking $\vec{\mu}$ (vector) is an assignment of tokens $(\mu_1, \mu_2, \ldots, \mu_n)$ to the places of a net.

An example shown in Fig. 8.3 is defined as: ***C = (P, T, I, O)***, where $P = \{p_1, p_2\}$, $T = \{t_{1,} t_2\}$, $I\{t_1\} = \{p_2\}$, $O\{t_1\} = \{p_1\}$, $I\{t_2\} = \{p_1\}$, $O\{t_2\} = \{p_2\}$. $I\{t_j\}$ and $O\{t_j\}$ are the input and output functions, which are the bags of the input and output positions of the transition t_j. The initial marking of the net is $\vec{\mu} = (0, 1)$.

The Petri nets execute by means of firing transitions. When a transition fires, it accomplishes the following activities:

1. Removes as many tokens from each input place as arcs from the place to the transition;
2. Deposits one token into each of its output places for each arc from the transition to the place.

A transition fires if the number of tokens in its input places is not less than the number of the arcs from the place to the transition. This rule is used by the semaphores introduced in the next section.

---

[14] A bag is a set where multiple occurrence of the elements of the same kind are allowed

The timed Petri nets allow to associate individual transition time with each transition. In the stochastic timed Petri nets the transition time can be described by some probability distribution.

A subset of the stochastic timed Petri nets has been implemented in the *F-Net toolkit*[15] developed by Fort-Inform Ltd [www.fi.ru/os/petri.php3].

8.2. BP modeling with the F-Net toolkit

Let us consider the application of the timed Petri net formalism to the BP modeling.

Each BP activity can be represented as a Petri subnet, which consists of two transitions and one place (Fig. 8.1). The transition t_1 firing represents the start of the activity ij and the transition t_2 denotes the possibility for the activity ij to finish.

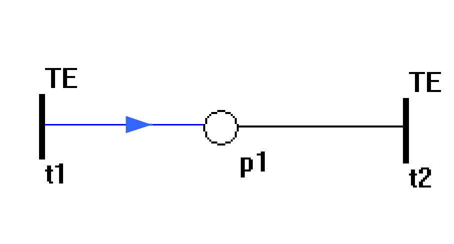


Figure 8.1. Representing a BP activity (notation on Fig. 8.1-8.3, 8.5, 8.7: arrowed lines point from transitions to places; the lines without arrows point from places to transitions)

If the number of the resources available to the activity ij is limited, the subnet is complemented with a semaphore, which is accomplished by means of the place p_2. The initial marking of the place p_2 is equal to the number of the resources available for the activity ij. Fig. 8.2 illustrates the semaphore made of the place p_2.

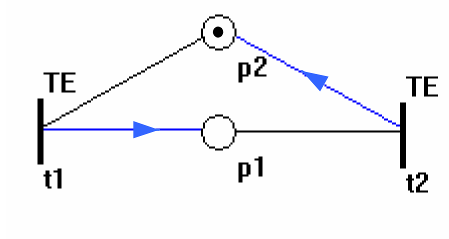


Figure 8.2. Place p2 as a semaphore

[15] F-net stands for "functional net"

If a set of activities is assigned to a pool of resources, the semaphore p_1 limits the number of the tokens in the subnet, which represents the corresponding activities, see Fig. 8.3. The initial marking in the place p_1 is equal to 2, therefore, only two activities can be processed at a time.

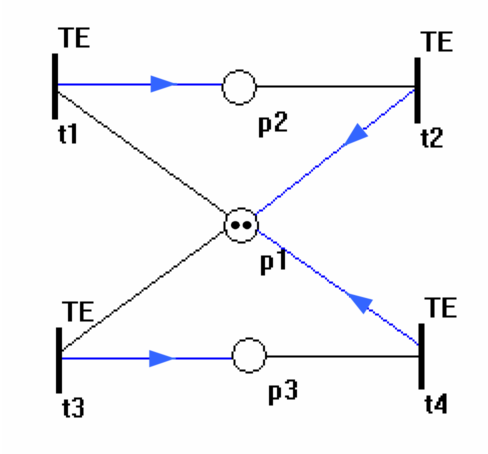


Figure 8.3. Semaphore p1 represents a limited resource

8.3. Examples of the timed Petri nets for the BP modeling

8.3.1. Example 1: unlimited resources

The BP IDEF3 model is shown in Fig. 8.4 and its parameters are specified in Tab. 8.1. Fig. 8.5 represents the timed Petri net for the case in question. There are no semaphores to limit the number of the activities being processed at a time. Executing the net on Fig. 8.5 yields the duration of the BP $T_\Sigma = 6$ (see the next section for more detail regarding the executing of the net). This result complies with one obtained with the analytical methods (sec. 5.3.1 and 5.3.2).

Table 8.1. Parameters of the BP model

Activity denotation	***Predecessors***	***Activity duration***
a1	*No*	*2*
a2	*No*	*1*
a3	*No*	*1*
a4	*No*	*2*
a5	*a1*	*2*
a6	*a2*	*3*
a7	*a1*	*4*
a8	*a3, a6*	*1*
a9	*a4, a5, a8*	*1*

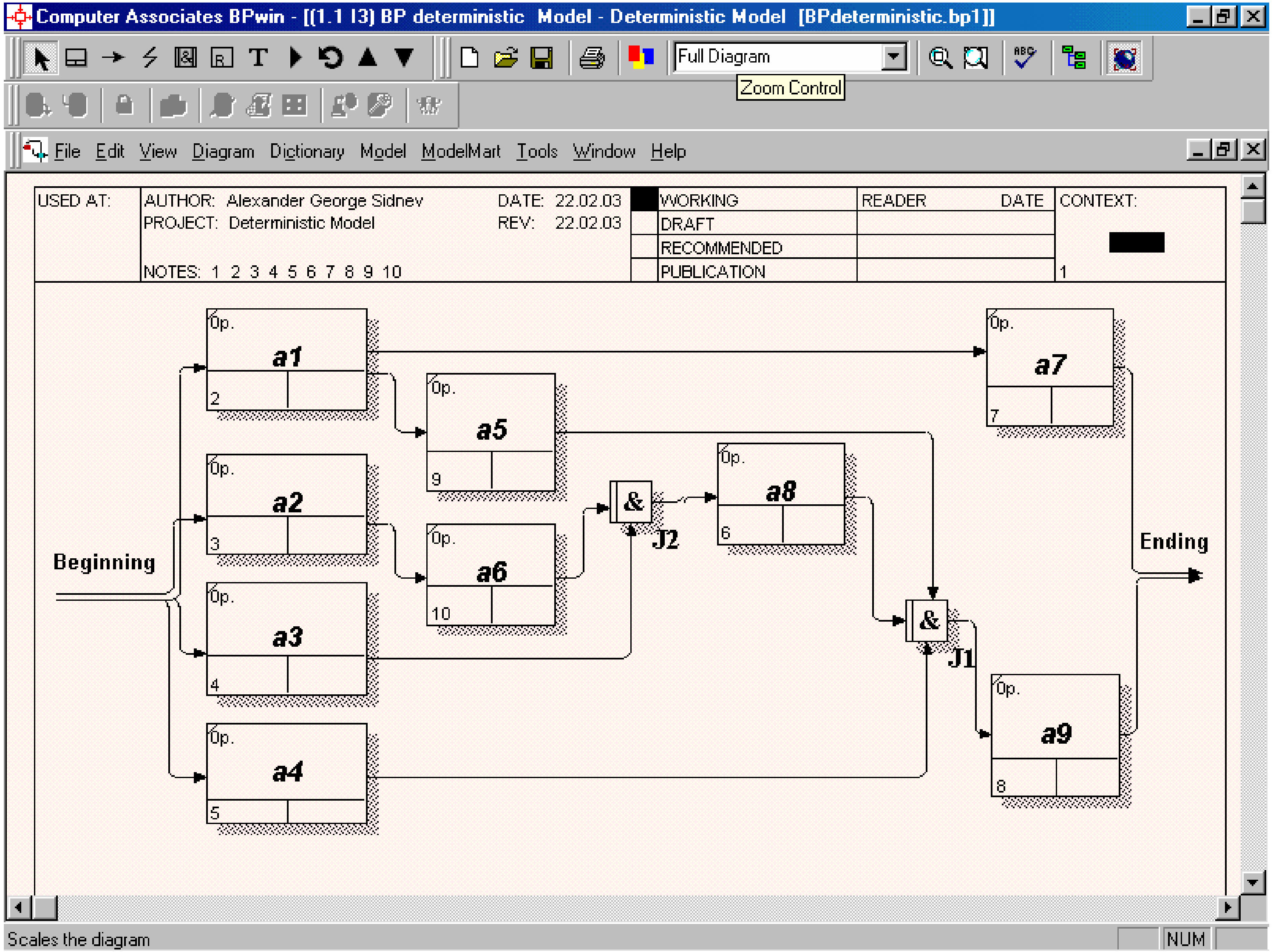


Figure 8.4. IDEF3 BP model with no reference to the resources assignment (screenshot: BPwin ®)

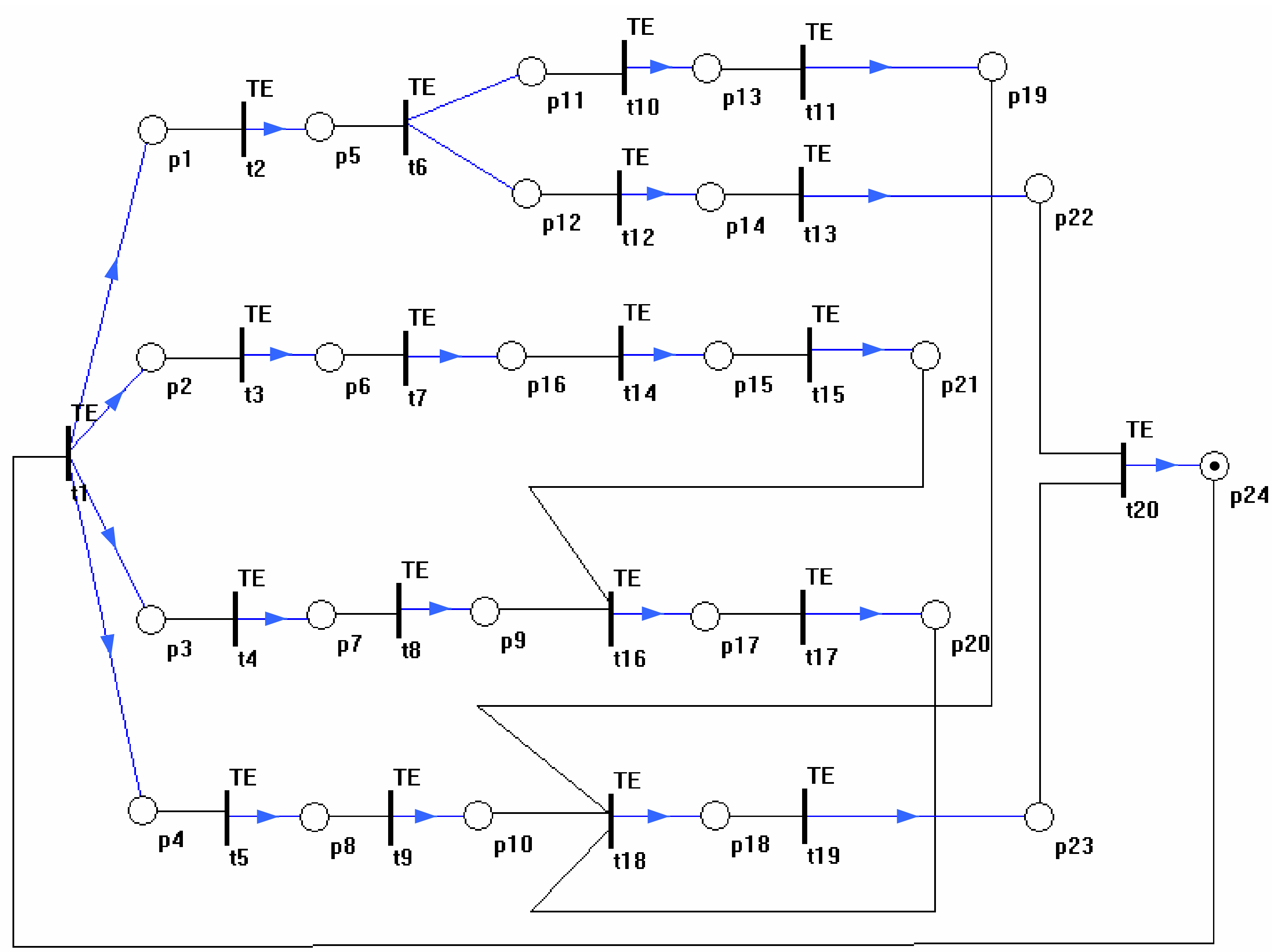


Figure 8.5. Timed Petri net corresponding to the model in Fig. 8.4.

8.3.2. Example 2: taking the resources limitation into account

BP IDEF3 model with the resources assignment is represented in Fig. 8.6. Let us consider the corresponding timed Petri Net depicted in Fig. 8.7.

The initial marking of places $p_{25}, p_{26}, p_{27}, p_{28}$ limits the number of the resources in each of the four pools assigned to the following activity collections (2,5), (4,9), (3,7), (1,6,8) correspondingly. Each pool possesses a single resource.

The toolkit sets the priority level to the transitions according to their numbers, i.e. the less the number the higher the priority.

The activity ij processing time is defined as the sum of the temporal parameters of the transitions t_1 and t_2, which belong to the activity subnet (Fig. 8.1 and 8.2): $t(t_1)+t(t_2)=t_{ij}$.

Let $t(t_1)=0$, then $t(t_2)=t_{ij}$. Thus, the total time of the place p_1 being not empty defines the total time of a resource being occupied by the activity *ij*. The Petri Net in question is a closed one. As soon as the BP has finished (a token arrives at the place p_{24}), the transition t_1 fires and the BP starts executing again. The number of the transition t_{20} firings in the period T of the modeling time, allows to compute the BP execution mean time.

The simulation reports in the form of the statistical data tables are shown in Fig. 8.8 and 8.9. By the modeling time $T_{mod} = 1000$, the transition t_{20} has fired 142 times. Hence, the mean time of the BP execution can be given by: $T = 1000 / 142 \approx 7.0$.

Recall that the BP execution time for the same BP with no resource limitations (each activity has its own executor) equals $T_\Sigma = 6$ (sec. 8.3.1). Here the execution time $T_\Sigma = 7.0$, is smaller than $T_\Sigma = 10.7$, which is calculated analytically for the queueing system (sec. 5.4.2.2). Why does it happen? Considering Tab. 5.9, we come to the conclusion that the rate of the incoming customer flow is such that the customers wait for the servers to become available for service. Thus, the net execution time of the activity 1 in the node 1 equals *2.0*, whereas the total activity execution time is *3.94*. This applies to other activities as well. Hence the larger total execution time for the queueing system $T_\Sigma = 10.7$ in comparison with the Petri net $T_\Sigma = 7.0$. The queues in the Petri net, Fig. 8.7, are shorter because we consider a closed model with a single customer, i.e. all executors (servers) serve just one customer.

Note that the Petri net in Fig. 8.7 is purely deterministic since its transitions have deterministic transition times, whereas the queueing systems are stochastic.

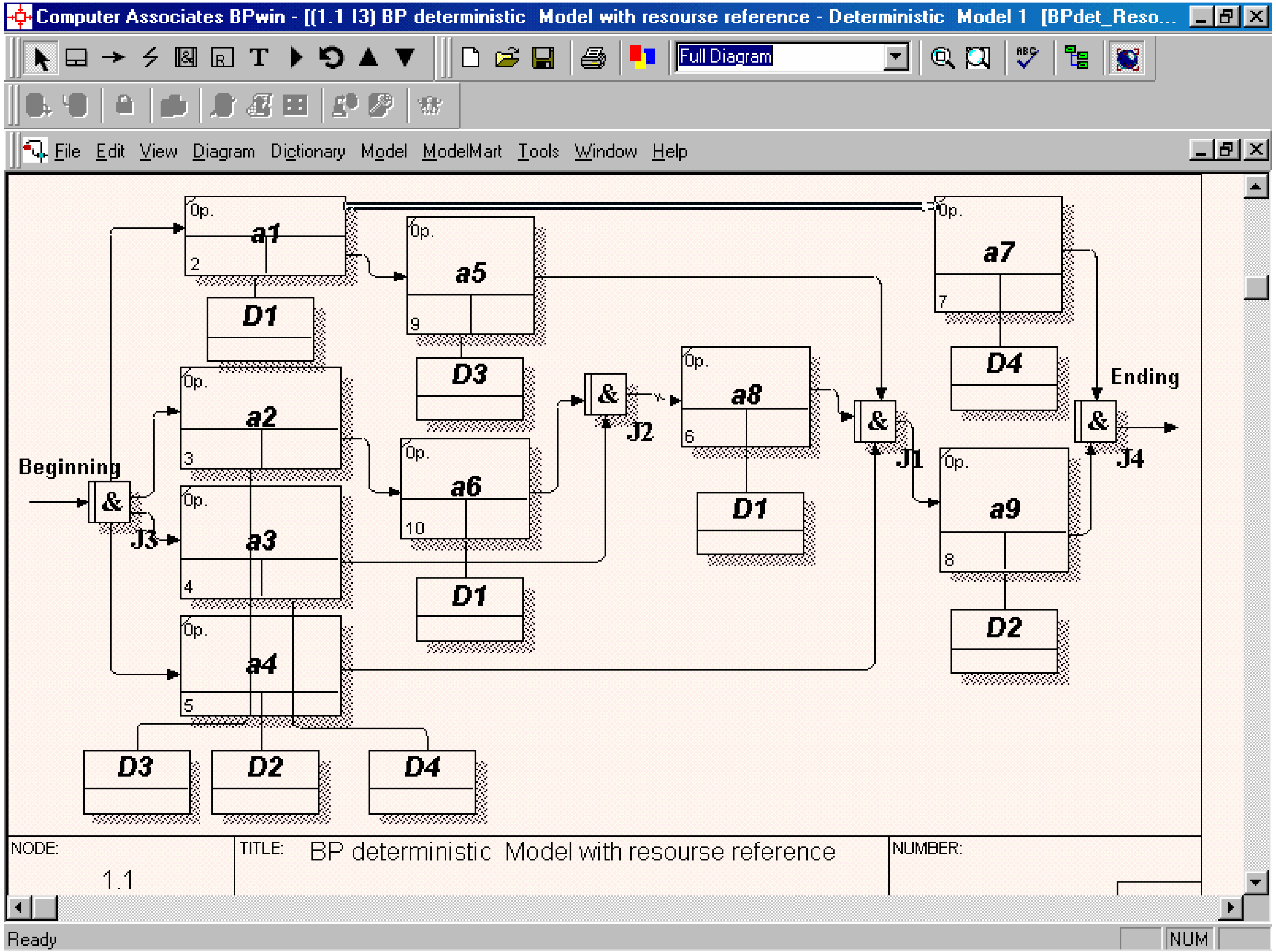


Figure 8.6. IDEF3 BP model accounting for the resources

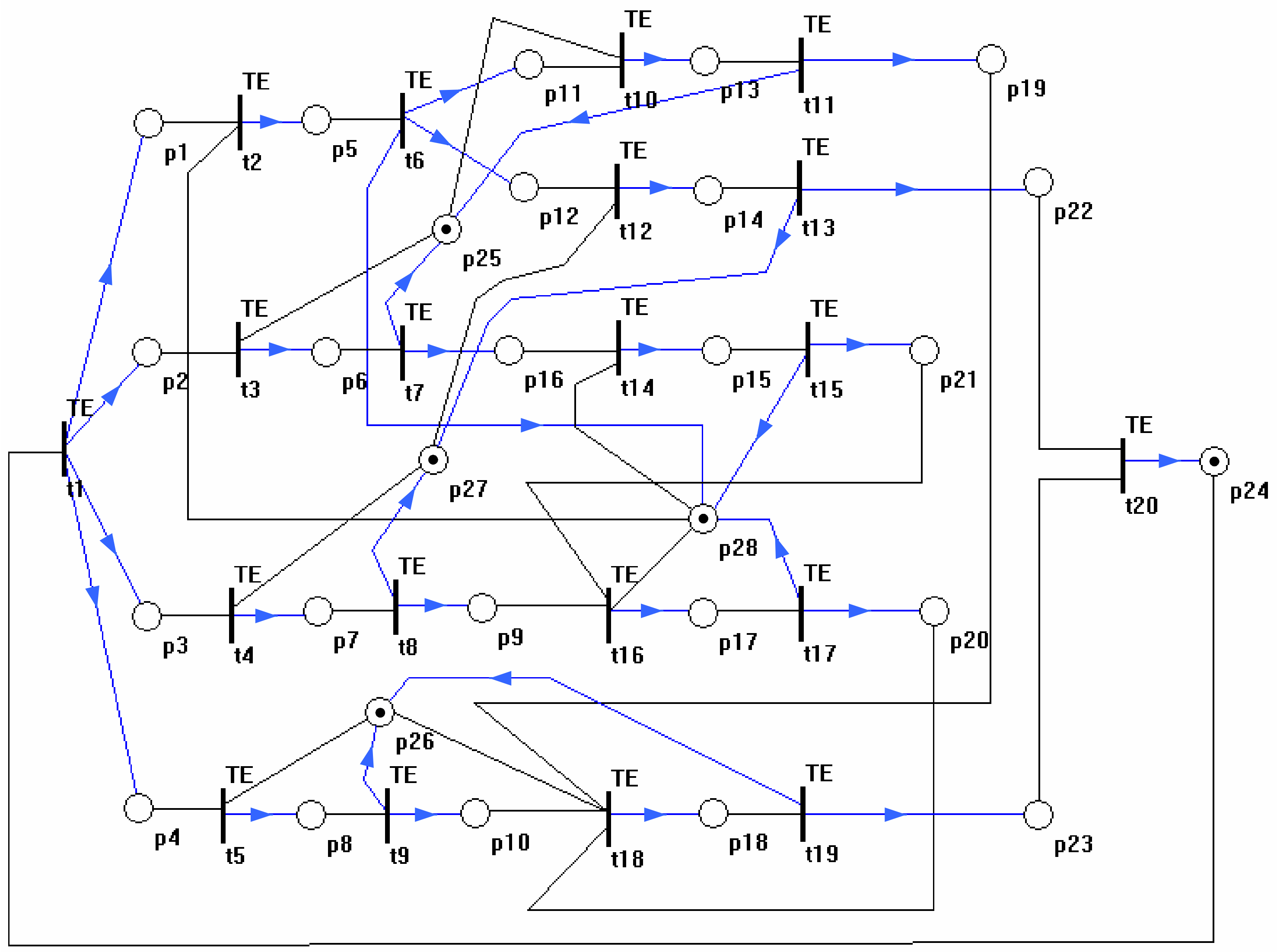
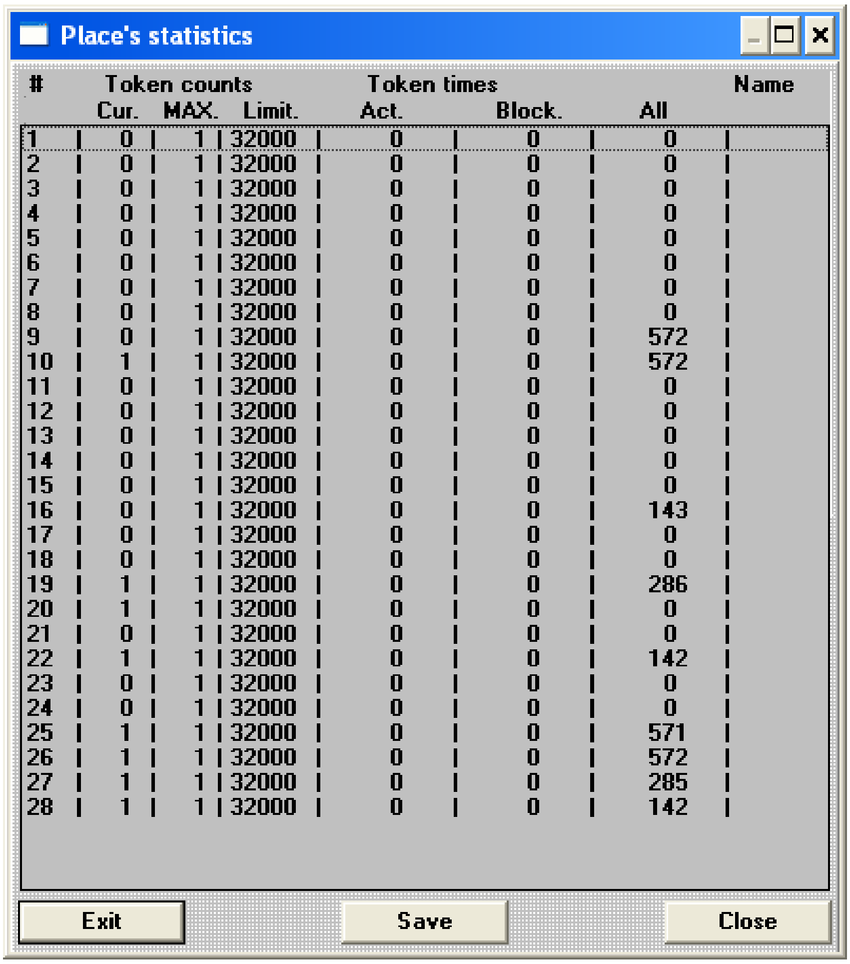
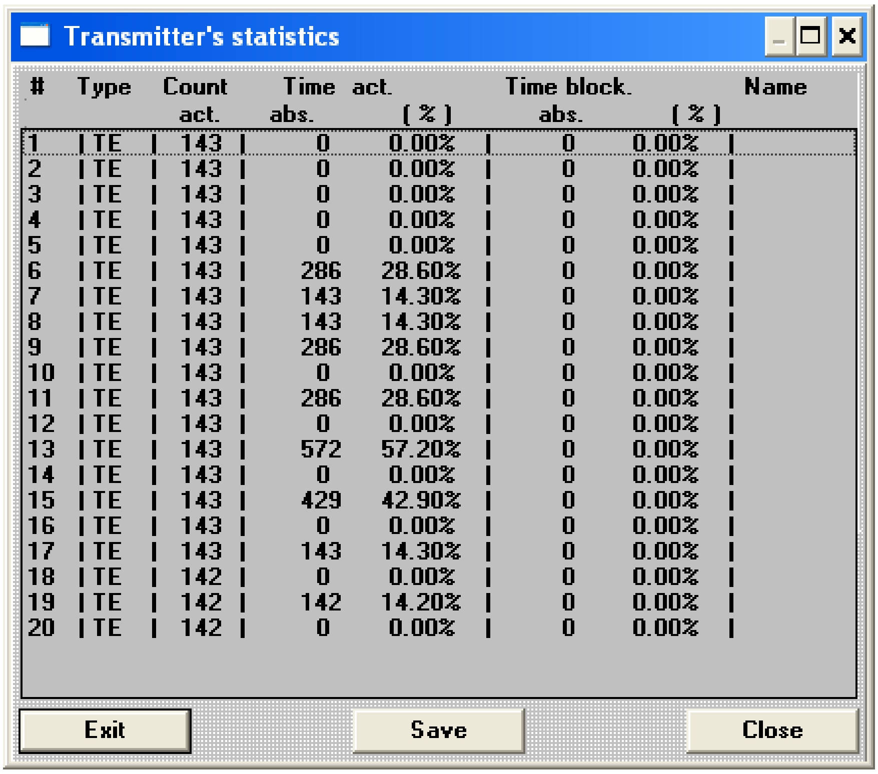


Figure 8.7. Timed Petri net corresponding to the model in Fig. 8.6.



#	Token counts Cur.	MAX.	Limit.	Token times Act.	Block.	All	Name
1	0	1	32000	0	0	0	
2	0	1	32000	0	0	0	
3	0	1	32000	0	0	0	
4	0	1	32000	0	0	0	
5	0	1	32000	0	0	0	
6	0	1	32000	0	0	0	
7	0	1	32000	0	0	0	
8	0	1	32000	0	0	0	
9	0	1	32000	0	0	572	
10	1	1	32000	0	0	572	
11	0	1	32000	0	0	0	
12	0	1	32000	0	0	0	
13	0	1	32000	0	0	0	
14	0	1	32000	0	0	0	
15	0	1	32000	0	0	0	
16	0	1	32000	0	0	143	
17	0	1	32000	0	0	0	
18	0	1	32000	0	0	0	
19	1	1	32000	0	0	286	
20	1	1	32000	0	0	0	
21	0	1	32000	0	0	0	
22	1	1	32000	0	0	142	
23	0	1	32000	0	0	0	
24	0	1	32000	0	0	0	
25	1	1	32000	0	0	571	
26	1	1	32000	0	0	572	
27	1	1	32000	0	0	285	
28	1	1	32000	0	0	142	

Figure 8.8. Statistics for the places



#	Type	Count act.	Time act. abs.	(%)	Time block. abs.	(%)	Name
1	TE	143	0	0.00%	0	0.00%	
2	TE	143	0	0.00%	0	0.00%	
3	TE	143	0	0.00%	0	0.00%	
4	TE	143	0	0.00%	0	0.00%	
5	TE	143	0	0.00%	0	0.00%	
6	TE	143	286	28.60%	0	0.00%	
7	TE	143	143	14.30%	0	0.00%	
8	TE	143	143	14.30%	0	0.00%	
9	TE	143	286	28.60%	0	0.00%	
10	TE	143	0	0.00%	0	0.00%	
11	TE	143	286	28.60%	0	0.00%	
12	TE	143	0	0.00%	0	0.00%	
13	TE	143	572	57.20%	0	0.00%	
14	TE	143	0	0.00%	0	0.00%	
15	TE	143	429	42.90%	0	0.00%	
16	TE	143	0	0.00%	0	0.00%	
17	TE	143	143	14.30%	0	0.00%	
18	TE	142	0	0.00%	0	0.00%	
19	TE	142	142	14.20%	0	0.00%	
20	TE	142	0	0.00%	0	0.00%	

Figure 8.9. Statistics for the transitions

Appendix A contains two GPSS simulation models, which correspond to the above-considered timed Petri nets. The duration of the activities is deterministic, so are the total BP execution times. Programs 1 and 2 model the closed nonlinear queueing networks. The first program corresponds to the Petri net shown in Fig. 8.5, while the second one is an analog of the Petri net in Fig. 8.7. The GPSS simulations lead to the same results as the Petri nets (program 1: $T_{\Sigma} = 6.0$, program 2: $T_{\Sigma} = 7.0$).

CHAPTER 9. CASE STUDY OF THE BP MODELING WITH GPSS

9.1. Problem definition

This example follows the pattern of the BP IDEF3 model built by V. Chebotarev, the chief analyst of Interface Ltd [www.interface.ru]. The original IDEF3 description has been shortened and translated into English. The background of the problem is outlined below.

An enterprise decides to implement an Enterprise Resource Planning (ERP) system. The enterprise contracts a consulting company to perform a comprehensive research of the enterprise business process (an “order”). The consulting company assigns a team of professionals to conduct the research. The team consists of four analysts and three experts. The responsibilities within the team are shared between its members depending on their qualifications. It is assumed that the analysts are capable of formulating tasks, while the experts are merely qualified performers. Here is the list of the team activities:

1. **“Casting the team”** is forming and assigning the roles in the team;
2. **“Creating the schedule”** is creating the enterprise exploration timetable. Each exploration stage has to be provided with beginning and ending dates;
3. “**Negotiating”** is receiving some assistance from the enterprise leaders, specifically regarding the delegation of power to the team members;
4. “**Defining the goals”** is stating the purpose of the ERP system implementation (for example, achieving profitability, promoting new goods, decreasing the risks etc);
5. **“Performing the function modeling”** is creating a set of the IDEF models for the enterprise business process;
6. **“Training the staff to use BPwin”** is training the enterprise staff to operate the BPwin toolkit and to create adequate IDEF models;
7. **“Creating the IDEFIx model”** is creating the BP data structure in the form of the IDEFIx-model. This activity, which is essential for the database construction, can be accomplished using special software, for instance, Erwin[16];
8. **“Creating the ABC-model”** is developing the Activity Based Costing methodology used for the comprehensive BP costs analysis;
9. **“Training the staff to use ERWin”** is training the enterprise staff to operate the ERWin toolkit. Close cooperation with the staff insures creating a valid BP model;

---

[16] *Erwin* is the software package of Computers Associates for creating data models

10. “**Documenting”** is releasing the documentation according to the established standards. This activity is aimed at providing the enterprise with the detailed requirements specification of the future ERP system.

The list of the activities is not complete, but it provides a good insight to the problem. Below we represent a simplified version of a real BP research, which precedes the efforts of the ERP system implementation. Note that the BP IDEF3 model has to be supplemented with the details necessary for converting it into the GPSS program.

9.1.1. BP IDEF3 model

The hierarchy of the BP IDEF3 model is presented in this section.

The first activity to be defined is the overall business process. The context activity is shown in Fig. 9.1. The context activity is the parent of all other activities of the diagram. The context activity is represented by its child activities, some of which are also decomposed in separate diagrams (Fig. 9.2 – 9.5). The leaf node activities, which are not detailed further, are marked with a skew line at the left upper corner of the corresponding box.

The activity tree diagram of the BPwin Model Explorer representing the model structure is shown in Fig. 9.6.

The BPwin toolkit provides each IDEF model activity with a number of text fields suitable for the user to place comments. That also helps to associate the GPSS program segments with the corresponding activities.

Fig. 9.7 represents the piece of the GPSS text associated with the activity named *Performing the Function Modeling*. The text is put into the *Definition/Note* text field of the activity № 23.1.0.

As soon as each leaf node activity is provided with its GPSS segment, it is time to combine the local segments into the GPSS modeling program according to the BP structure represented by the activity tree diagram.

Another way to follow the BP structure is to create the Node tree diagrams of the integrated activities. The node tree diagram of the context activity is shown in Fig. 9.8.

For complex BPs it is reasonable to create the BP flowchart with the local segments denoted by the labels supposed to be used in the GPSS program. Fig. 9.9 represents the flowchart for the BP in question.

Converting the IDEF3 model into the GPSS program requires some quantitative data about the BP activities. It is worth mentioning that the IDEF3 model itself does not pretend to be detailed enough for the quantitative modeling.

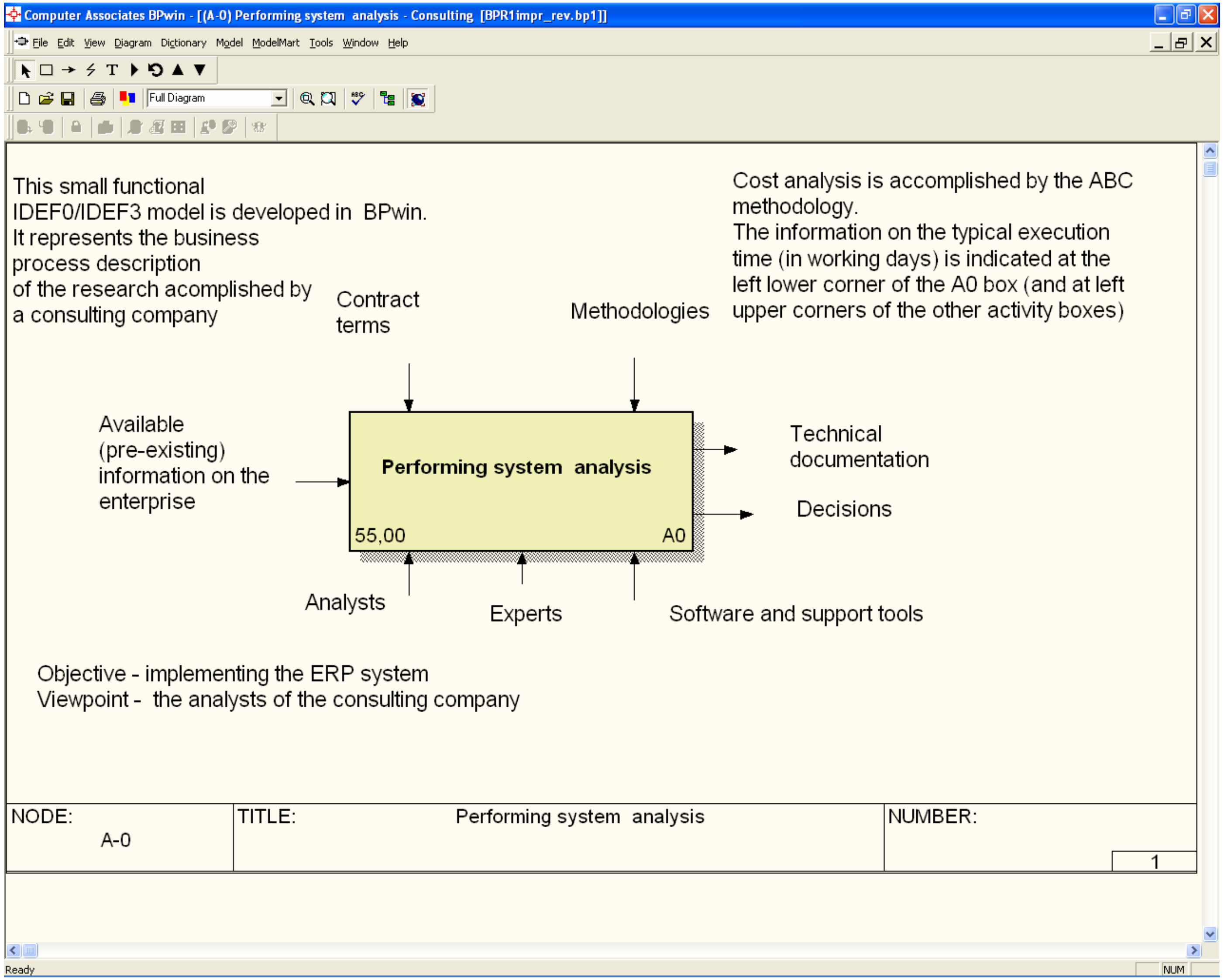


Figure 9.1. BP context activity diagram (screenshot: BPwin ®)

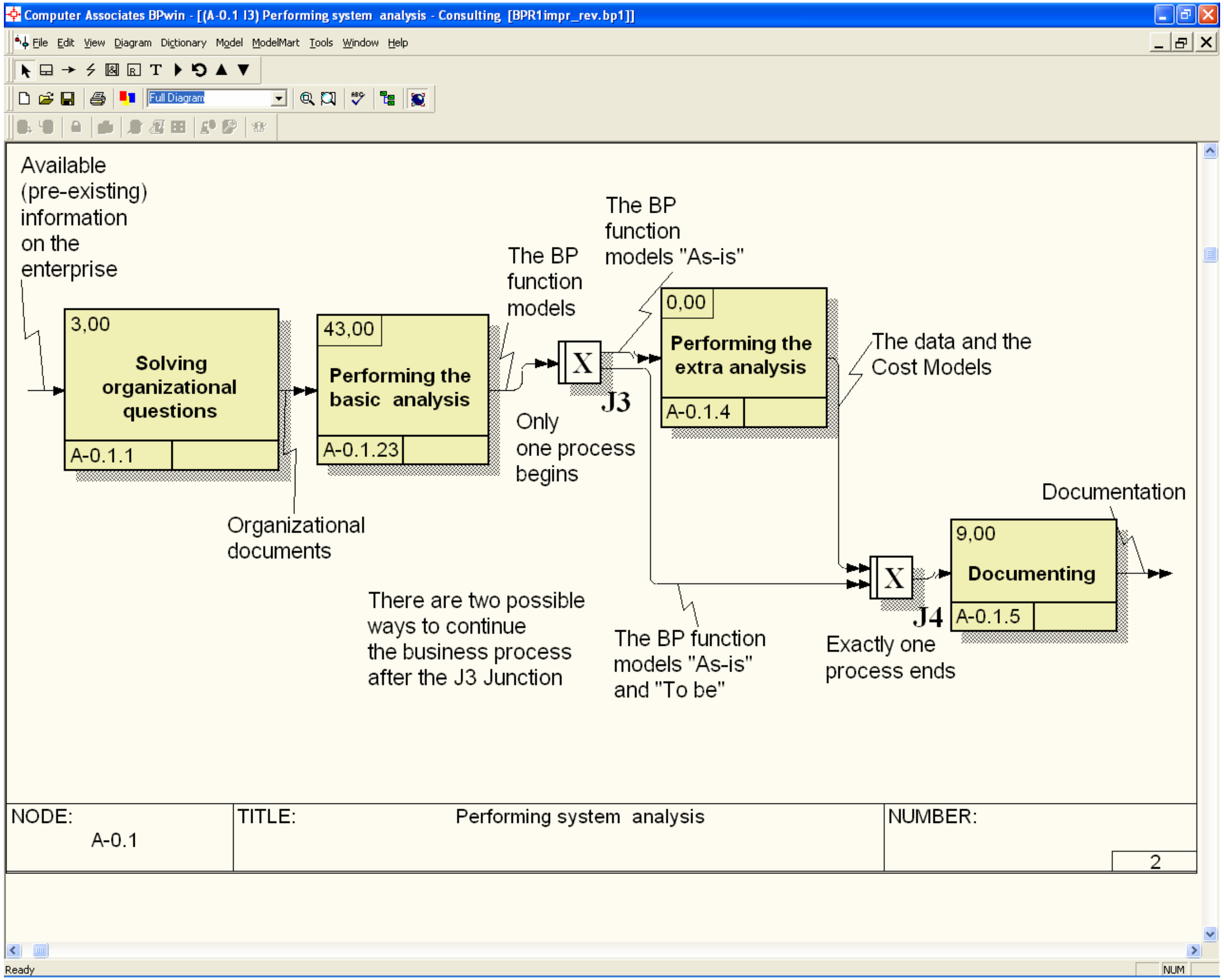


Figure 9.2. Performing system analysis of the enterprise activities: the top level decomposition (screenshot: BPwin ®)

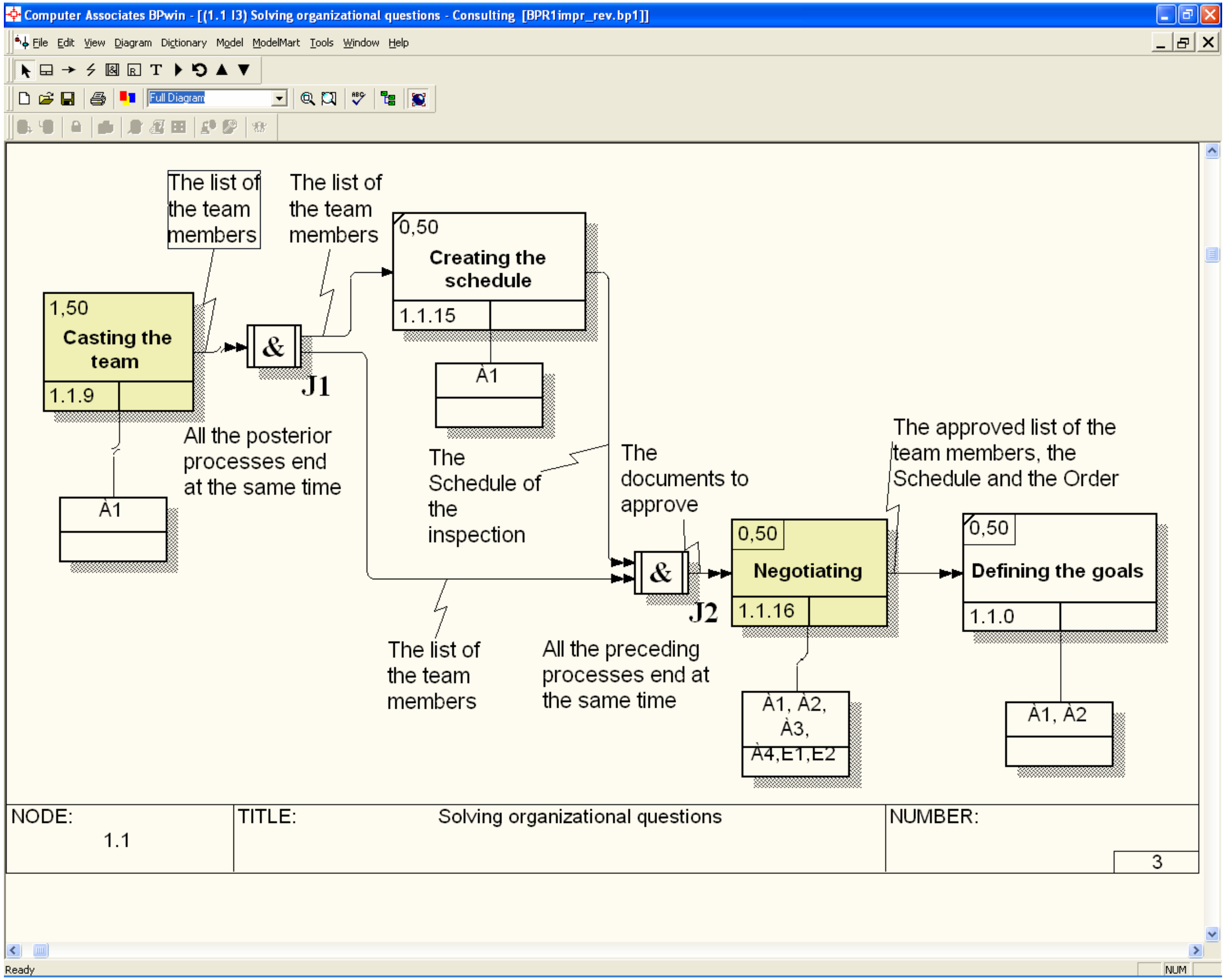


Figure 9.3. Solving organizational questions (screenshot: BPwin ®)

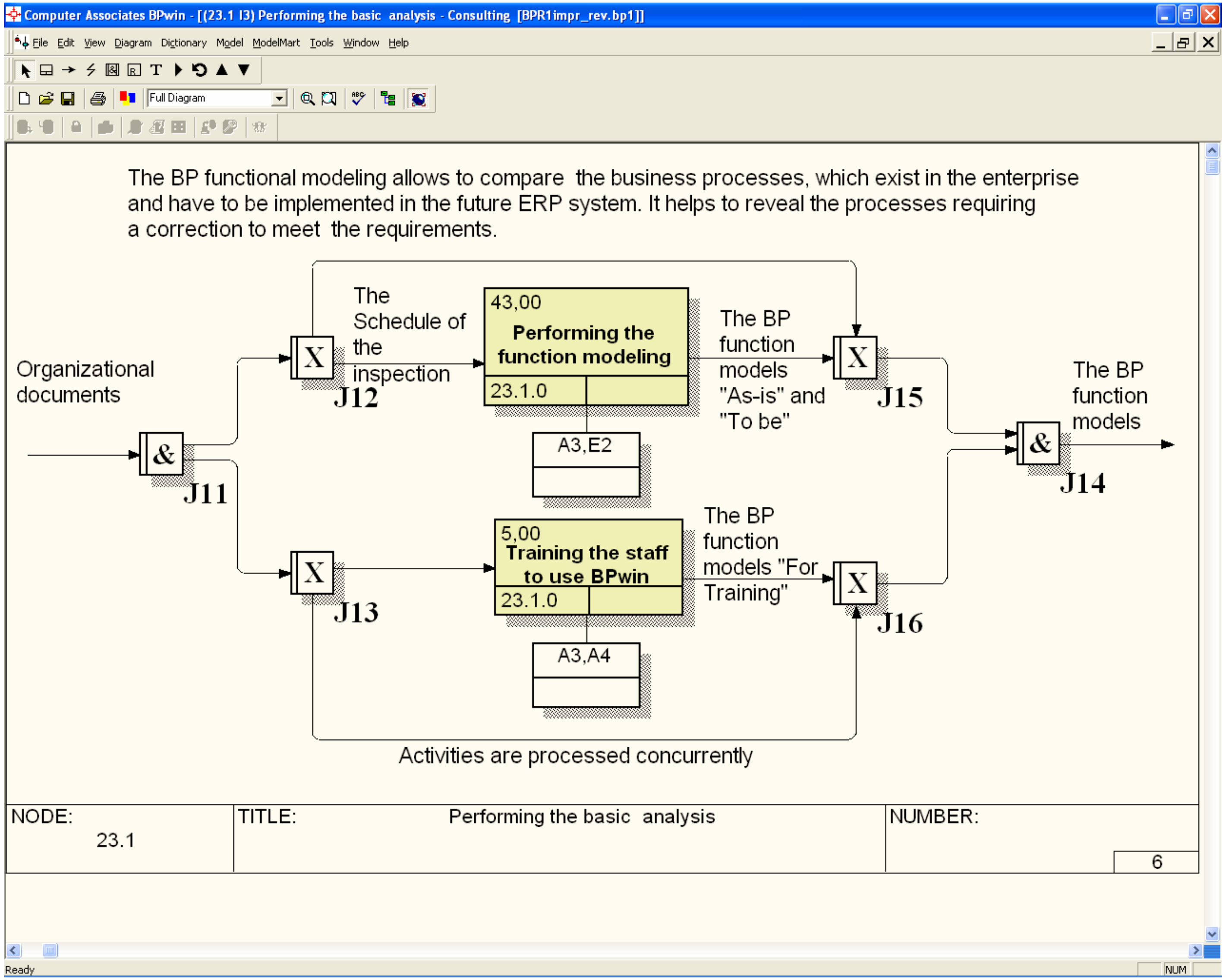


Figure 9.4. Basic analysis (screenshot: BPwin ®)

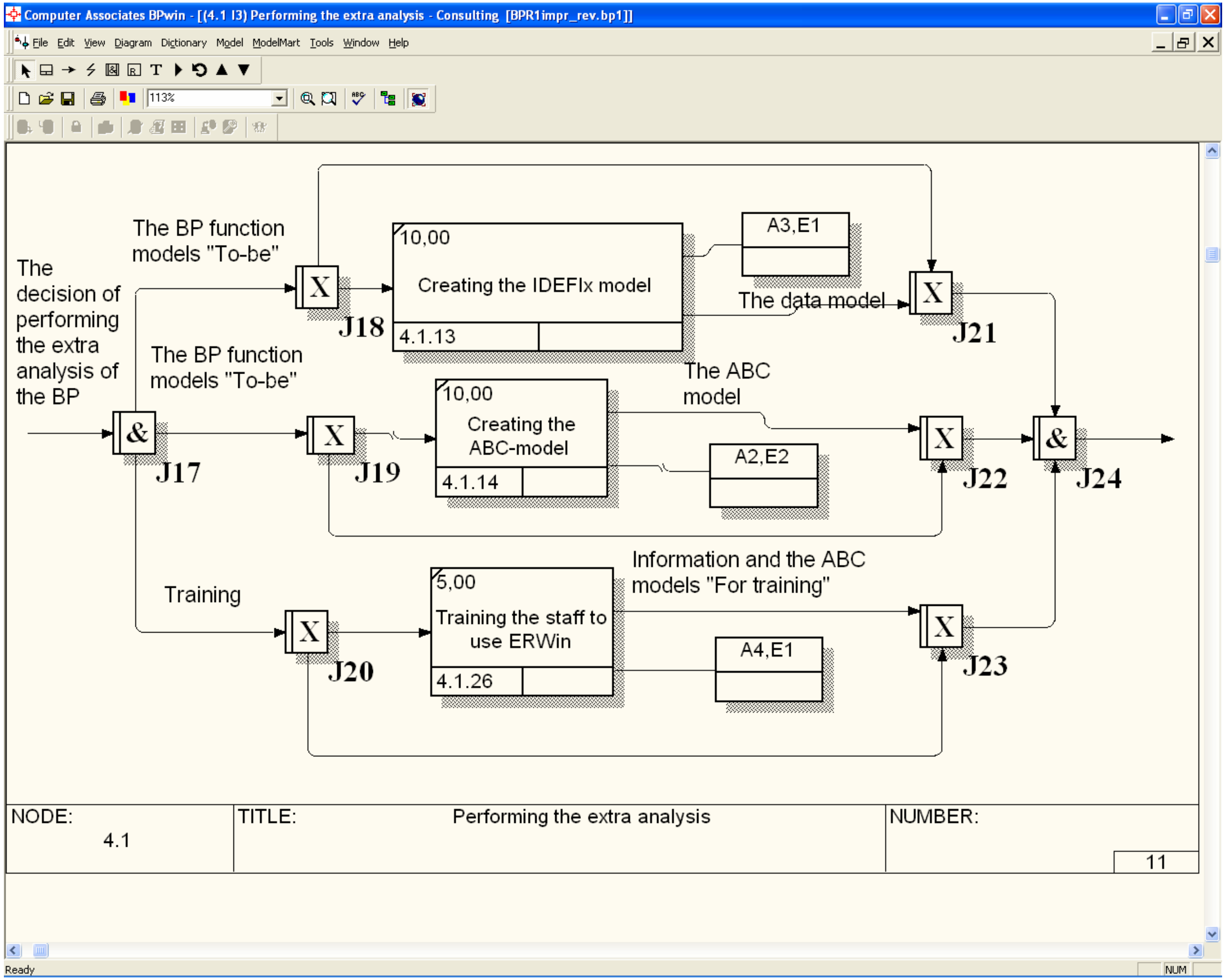


Figure 9.5. Extra analysis (screenshot: BPwin ®)

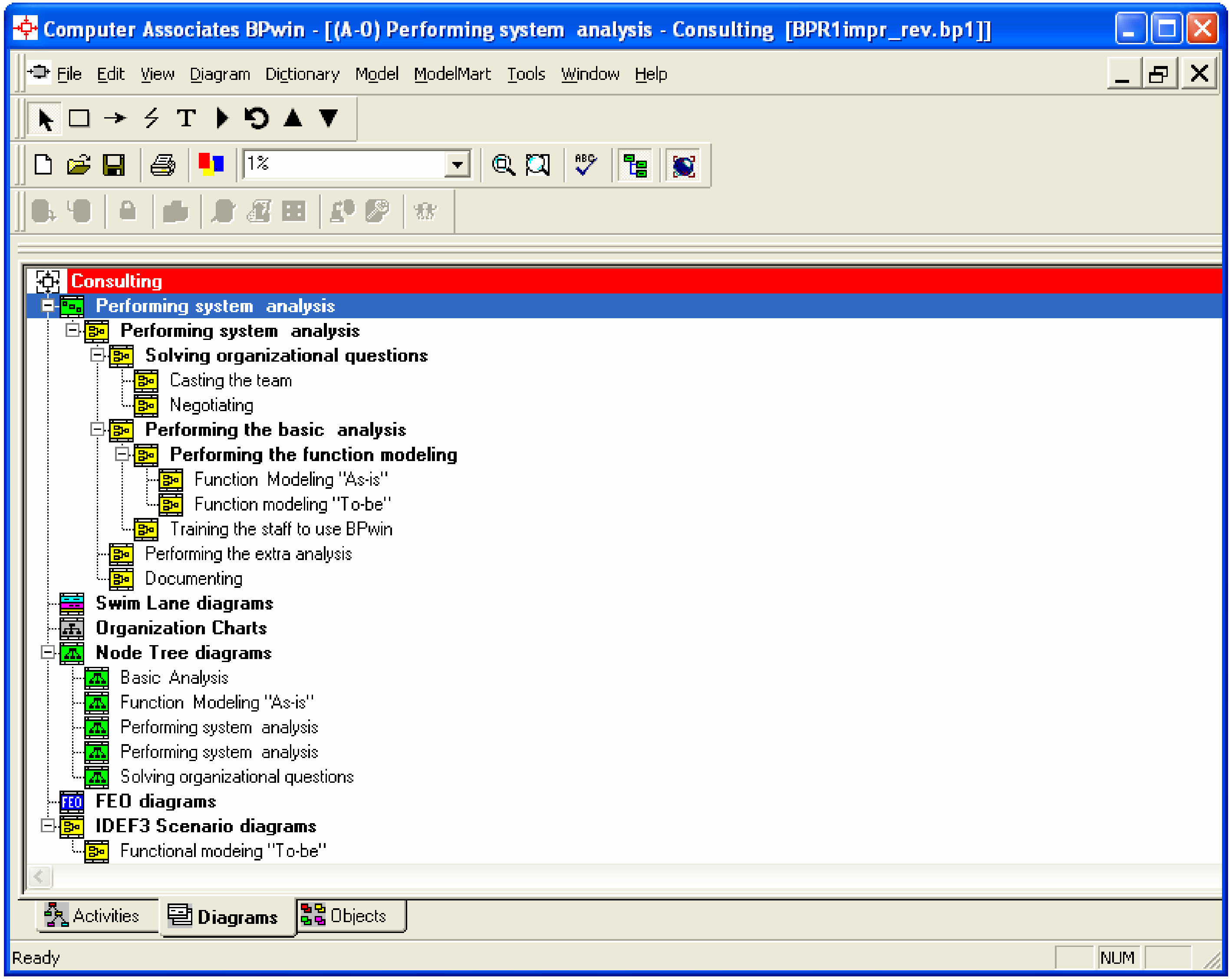


Figure 9.6. BP node tree diagram (screenshot: BPwin ®)

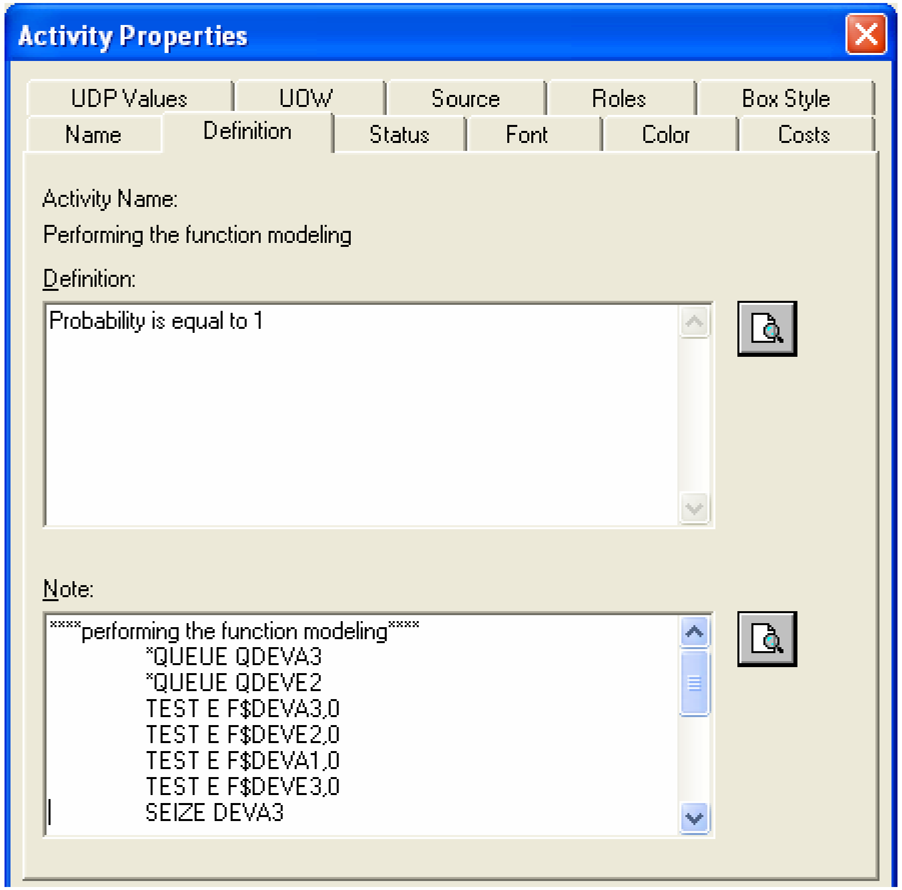


Figure 9.7. Activity properties (screenshot: BPwin ®)

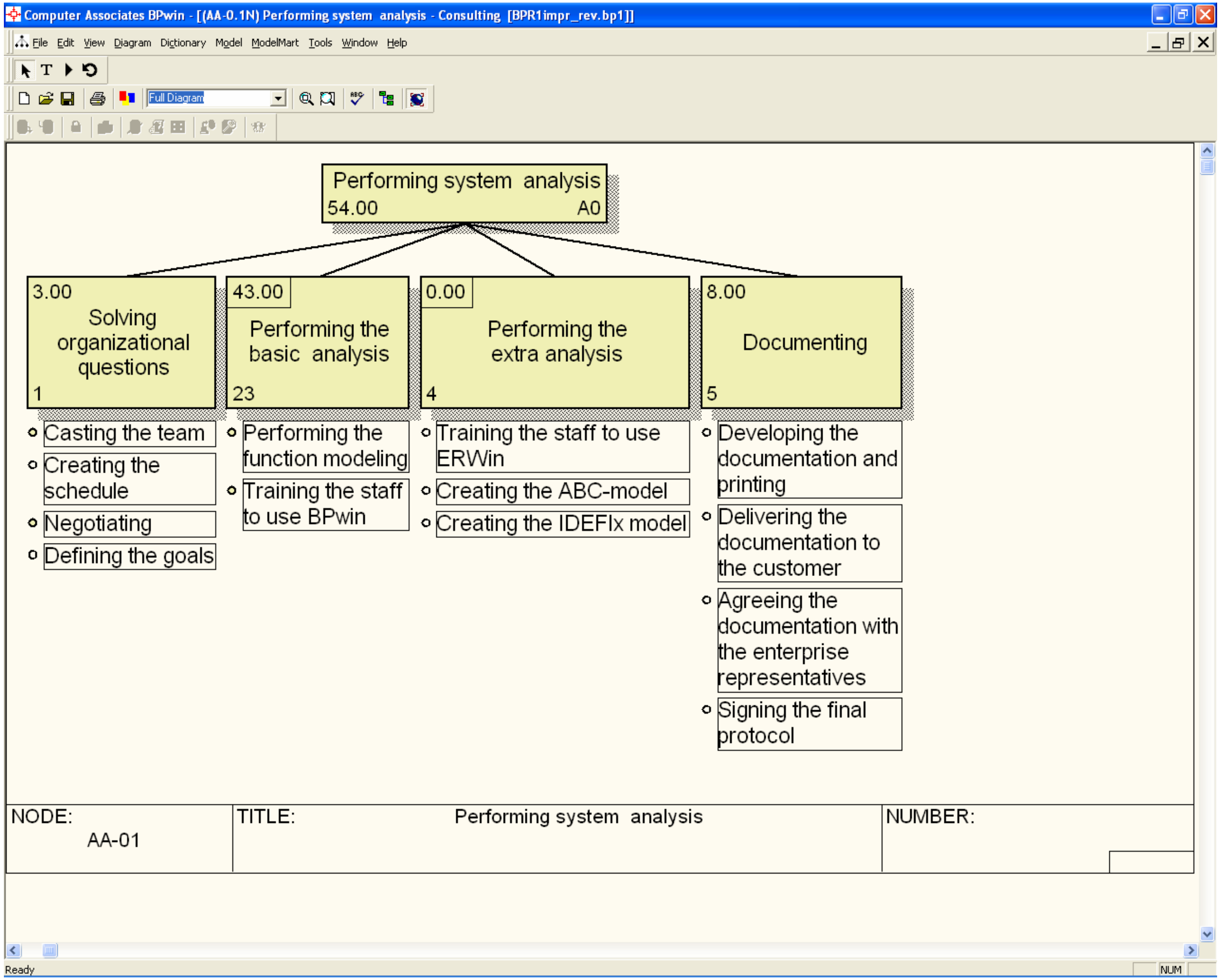


Figure 9.8. Node tree diagram for the context diagram (screenshot: BPwin ®)

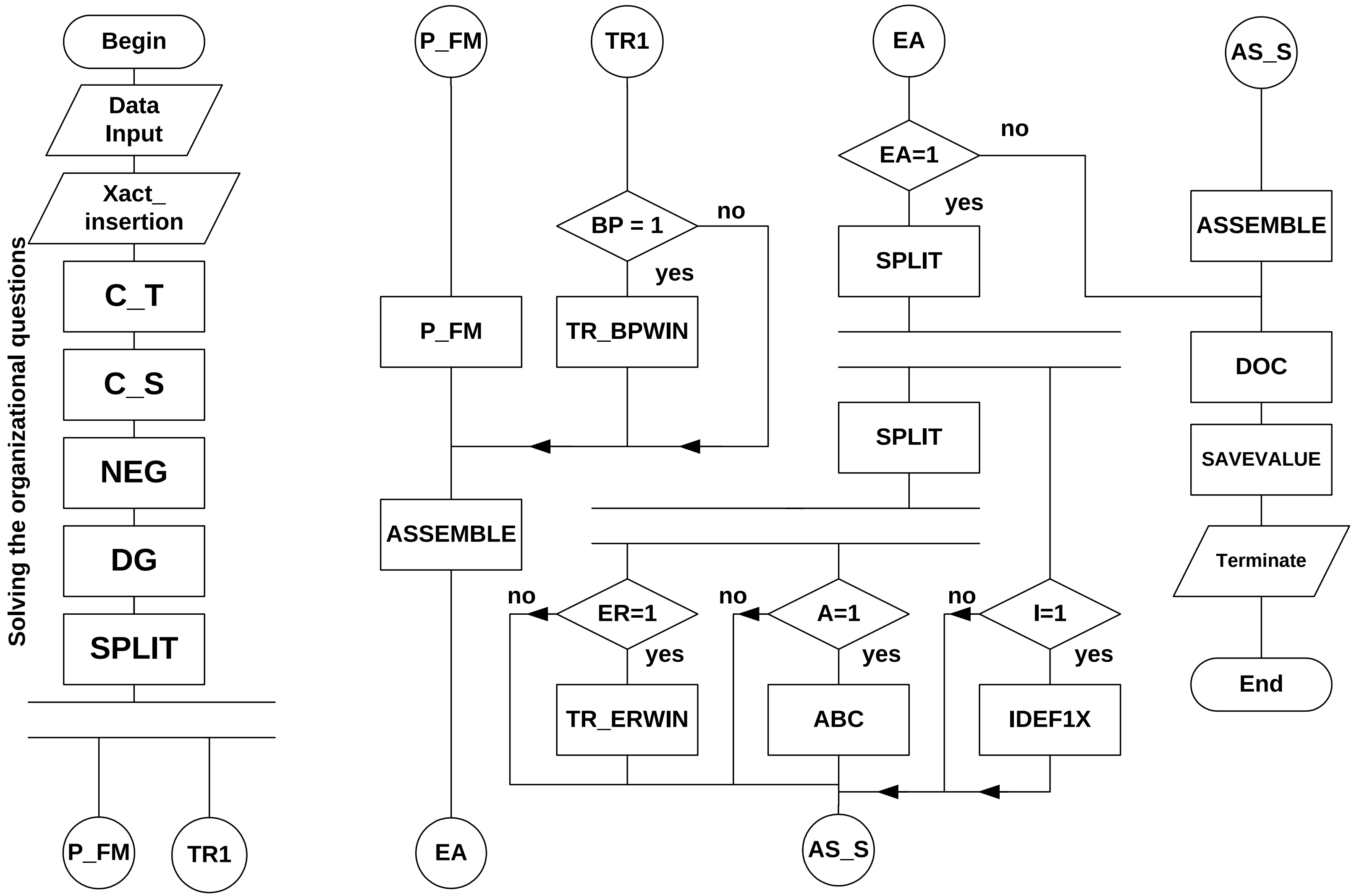


Figure 9.9. BP flowchart (see Tab. 9.1. for notation)

9.1.2. Activity definition

Let us start with elaborating the BP IDEF3 model. First of all we need to assign some fixed costs of a single execution to each leaf node activity. Each activity has certain probability of being included in the overall BP. The leaf node activities provided with the denotations, costs, and probabilities are listed in Tab. 9.1.

Table 9.1. Parameters of the BP model

№	*Activity name*	*Denotation*	*Fixed costs*	*Probability*
1	Casting the team	C_T	6000	1
2	Creating the schedule	C_S	15000	1
3	Negotiating	NEG	9000	1
4	Defining the goals	DG	4500	1
5	Performing the function modeling	P_FM	30000	1
6	Training the staff to use BPwin	TR_BPWIN	15000	0.5
7	Creating the IDEFIx model	IDEF1X	12000	0.3
8	Creating the ABC-model	ABC	24000	0.54
9	Training the staff to use ERWin	TR_ERWIN	18000	0.06
10	Documenting	DOC	24000	1

9.2. Activity costing

Let us estimate the costs of one unit of the BP execution, for example, the costs per project. Suppose the project costs consist of the following two parts:

1. WHOLEPART1: the sum of the fixed costs of the activities that are actually accomplished during the course of the project;

2. WHOLEPART2: the project variable costs with the reference to the time consumed by the team members.

The resulting project costs are: WHOLEPART = WHOLEPART1 + WHOLEPART2.

9.2.1. Assigning activities to executors. Activity cost data

All data required for calculating the project costs are given in Tab. 9.2.

Table 9.2. Data for calculating the cost of the project

Team member hourly rate	***Activity, fixed costs, probability***									
	C_T 6000	**C_S 15000**	**NEG 9000**	**DG 4500**	**P_FM 30000**	**TR_BP WIN 15000**	**IDEF1X 12000**	**ABC 24000**	**TR_ ERW 18000**	**DOC 2400 0**
	1	1	1	1	1	0.5	0.3	0.54	0.06	1
A1 120	12	4	4	4						
A2 100			4	4				80		
A3 100			4		200	40	80			
A4 100			4			40			40	
E1 40			4				80		40	70
E2 40			4		200			80		70
E3 40										70

The first column "Team member hourly rate" represents the hourly rate of each member of the team (e.g. in US$). Note that the analysts are the most expensive employees.

The table body represents the time usage of each member for each activity. For instance, the row represented below says that the analyst A1 with hourly rate 120$ is occupied:

- for 12 hours in the activity "Casting the team (C_T) ";
- for 4 hours in the activity "Creating the schedule (C_S) ";
- for 4 hours in the activity "Negotiating (NEG) ";
- for 4 hours in the activity "Defining the goal (DG)".

A1 120	**12**	**4**	**4**	**4**						

The upper table rows contain the activities fixed costs and probabilities. For example, the fixed cost of the "Casting the team (C_T)" activity is 6000$ for a single execution and it has the probability of 1, i.e. it is compulsory, unlike the "BPwin training (TR_BPWIN)" activity whose probability equals 0.5.

9.3. GPSS BP model parameters

GPSS BP model outputs are the temporary and the cost metrics of the BP in question. The input and the output parameters of the GPSS program are listed below.

9.3.1. Input parameters

1. The average execution time of each of the ten activities (C_T, C_S, NEG, DG, P_FM, TR_BPWIN, IDEF1X, ABC, TR_ERWIN, DOC) is determined as the first operand of the corresponding ADVANCE block in the GPSS program;
2. The fixed costs of each activity execution are defined as the INITIAL values of SAVEVALUE XC_T, XC_S, XNEG, XDG, XP_FM, XTR_BPWIN, X$IDEF1X, X$ABC, XTR_ERWIN, XDOC. (See the INITIAL command at the head of the GPSS program);
3. The arrival intervals of the orders for the BP research (in hours) are given as the corresponding operand of the GENERATE block;
4. Each team member is represented as a FACILITY in the GPSS program. The hourly rate for each FACILITY occupation is defined at the head of the program by the INITIAL assignment of the corresponding SAVEVALUE (X$PRICEDEVA1, X$PRICEDEVA2, X$PRICEDEVE1, etc.).

9.3.2. Output parameters

1. Each FACILITY demonstrates the percentage of the team members busy time;
2. The Average Queue content of all FACILITies's Queues and the average waiting time in all Queues;
3. The average project execution time T_IME and its standard deviation;
4. The cumulative sum earned by each of the team members. This values are saved in SAVEVALUES X$COSTDEVA1, X$COSTDEVA2 X$COSTDEVA3 X$COSTDEVA4 X$COSTDEVE1 X$COSTDEVE2 X$COSTDEVE3;
5. The average costs of a single project execution WHOLECOST and its fixed and variable parts.

9.4. BP research using GPSS

Now we have the GPSS simulation program that substitutes for the real BP in question. According to the BPR steps, we will now reveal the characteristics of the "As-is" business process.

We assume that the GPSS program follows the initial IDEF3 BP model and, therefore, it represents the BP exactly in the "As-is" fashion.

The BP in question has many characteristics, which define its essential properties. Some of them may be discovered by means of experimenting with the BP GPSS program. First of all, we mean the temporal parameters such as the BP execution time and the utilization of the team members, which show whether there is a possibility of the BP intensification, and, finally, the costs of the BP execution as a universal BP metrics.

Note that both the IDEF3 model and the GPSS program represent the BP as a set of activities. This is the activity point of view at the BP modelling. Another way to create the BP model is to define it as a set of serving systems with an appropriate customer routing.

Having this in mind, we can identify the following two modifications of the BP GPSS program.

9.4.1. GPSS open model

The GPSS open model consists of a transaction generator and a processing system, which belongs to the class of serving systems. A transaction, which enters the system, has to leave it at the end. The amount of transactions staying in the system is not limited and it depends on how fast the transactions are served. Returning to the case study, we consider a transaction as a request (an order, a contract for a new project) to research the BP, so a transaction that is leaving the system is treated as the fact that the request has been fulfilled. This model allows to reveal how the intensity of the requests affects the BP output parameters.

9.4.2. GPSS closed model

The closed model has the constant number of transactions to be served. The incoming request enters the system at the very moment when the request, which has been just fulfilled, leaves the processing system. This program helps us to discover the potential abilities of the team measured in terms of the output parameters introduced in Sec. 9.3.2.

Fig. 9.10 illustrates the core idea of the open and closed GPSS models.

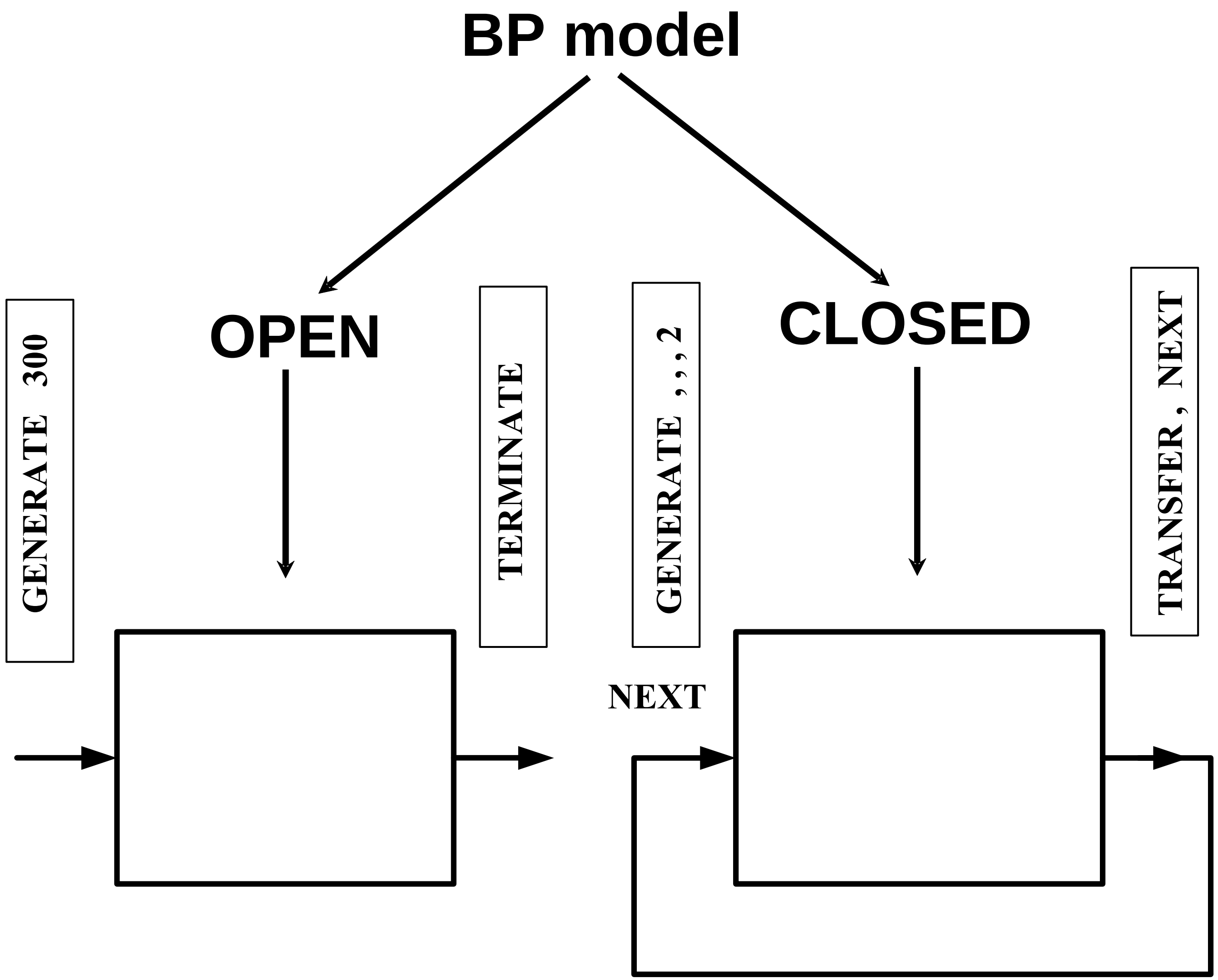


Figure 9.10. Open and closed BP models

Note that the program body remains the same for both models, e.g. compare:

OPEN MODEL	*CLOSED MODEL*
GENERATE 340	GENERATE ,,, 2
PROGRAM BODY	NEXT PROGRAM BODY
TERMINATE	TRANSFER, NEXT

9.4.3. Preventing deadlocks

There are a number of modeling issues crucial for obtaining accurate and valid simulation models. One of these issues is the proper modeling of downtimes. The downtimes of a serving system can be caused by its deadlock due to the improper load management. Let us compare two typical GPSS segments, Tab. 9.3.

Table 9.3. GPSS segments with a possible deadlock and with no deadlocks

With a possible deadlock	*With no deadlocks*
****negotiating****	****negotiating****
QUEUE QDEVA1	QUEUE QDEVA1
QUEUE QDEVA2	QUEUE QDEVA2
SEIZE DEVA1	**TEST E (F$DEVA1+F$DEVA2),0**
SEIZE DEVA2	SEIZE DEVA2
DEPART QDEVA1	SEIZE DEVA2
DEPART QDEVA2	DEPART QDEVA1
ADVANCE 4,1	DEPART QDEVA2
RELEASE DEVA1	ADVANCE 4,1
RELEASE DEVA2	RELEASE DEVA1
	RELEASE DEVA2

In the first segment a deadlock occurs when a transaction captures FACILITY DEVA1 regardless the fact that FACILITY DEVA2 is not yet free. Therefore, FACILITY DEVA1 being SEIZEd wastes its time unless FACILITY DEVA2 is free.

To prevent the deadlock, a block statement TEST is inserted into the GPSS program. The block TEST checks whether both FACILITIES are not busy. As soon as both of them are free, their System Numerical Attributes F$DEVA1 and F$DEVA2 become equal to zero.

The BP GPSS program text for the closed BP model is given in Appendix B.

9.4.4. BP research

Let us perform the BP research based on the open and closed models.

Let N be in the range from 1 to 10, where N defines the permanent transaction number in the closed system (the number of concurrent projects). Let TI be in the range from 300 to

400 (hours), where TI defines the inter-arrival interval of the incoming orders for the BP research.

The total amount of FACILITIES – the serving channels denoting the team members – is equal to seven (four analysts and three experts that differ in their hourly rates and responsibilities). The time that is hold by a FACILITY is assumed to be uniformly distributed.

The average holding time is defined at the head of the GPSS program.

Three series of experiments for several different organizations of the BP have been conducted (all FACILITIES serve transactions with no priorities; 1st correction; 2nd correction). Each experiment includes 100 orders (projects). The outcomes of the models are such that they can be compared with each other.

Examining the upper part of Tab. 9.4, we conclude that load of the team members is unbalanced. Hence, some corrections in terms of prioritizing (introducing the priority-serving discipline instead of the Fist-In-First-Out) and re-assigning the activities have to be performed to balance the workload. As it can be seen from the remaining section of Tab. 9.4, these corrections have indeed improved the BP and reduced its duration.

Let us analyze the results.

- *The 1st series of experiments:*

The interval of 340 hours between the orders (=consequent projects) is the minimum; if the interval is decreased, the duration and the cost of the projects significantly increase. The team members workload is unbalanced. Introducing priorities (the activity with the highest priority is served by a FACILITY while others are waiting) and re-assigning the activities within the group may improve the results. For example, the load rate of the team member E1 is close to 1. It means E1 (1) is overloaded and (2) holds the BP execution process and prevents other team members from being loaded.

- *The 2nd series of experiments (the 1st correction has been implemented):*

The duration of a single BP has decreased from 345 to 281 hours and the workload balance of the team members has slightly improved.

- *The 3rd series of experiments (the 2nd correction has been implemented):*

The duration of a single BP has decreased from 281 to 250 hours and the workload balance of the team members has further improved, although there is still room for further improvements.

This example demonstrates the so-called local research. However, even such a simple model allows to perform rather serious tasks of the BP analysis and synthesis.

Table 9.4. Results of the BP research

№	***TI***	***N***	***T_IME***	$\sigma(T\_IME)$	***WHOLECOST***	***TEAM MEMBER LOAD (FACILITY UTILITY)***						
						A1	***A2***	***A3***	***A4***	***E1***	***E2***	***E3***
Closed model												
1		1	345	49	159213	0.069	0.106	0.687	0.074	0.262	0.876	0.203
2		2	624	158	160755	0.077	0.120	0.769	0.079	0.390	0.966	0.224
3		3	926	236	162123	0.086	0.132	0.815	0.085	0.430	0.975	0.226
4		10	3097	1035	166755	0.092	0.133	0.837	0.092	0.431	0.977	0.226
Open model												
1	400	-	340	46	154579	0.060	0.070	0.587	0.065	0.216	0.732	0.175
2	370	-	345	47	154578	0.065	0.075	0.634	0.070	0.234	0.791	0.189
3	350	-	360	66	157203	0.067	0.098	0.670	0.088	0.245	0.852	0.199
4	340	-	366	55	157559	0.069	0.102	0.695	0.071	0.261	0.876	0.202
5	340	-	388	84	160476	0.070	0.112	0.709	0.076	0.266	0.892	0.205
1st correction. Activities reassignment. Priority-service discipline. Closed model												
1	-	1	281	60	166991	0.547	0.105	0.596	0.090	0.398	0.802	0.711
2	-	2	535	188	172941	0.575	0.162	0.641	0.092	0.439	0.850	0.747
2nd correction. Activities reassignment. Priority-service discipline. Closed model												
1	-	1	250	58	173849	0.615	0.276	0.669	0.263	0.323	0.779	0.679
2	-	2	475	141	176692	0.644	0.306	0.715	0.278	0.352	0.838	0.731

CHAPTER 10. SAMPLE BP M&S TRAINING TASK

10.1. Defining the BP model

The BP models are defined in the IDEF3 format. According to the IDEF3 syntax, the BP structure mapping onto the IDEF3 model is accomplished with the use of junctions. Without loss of generality, it can be assumed that only two types of junctions are allowed: & and XOR. This constraint improves the conversion of the BP IDEF3 model into the BP simulation model.

10.2. Defining the BP cost model

The BP cost model is the integration of the cost models of the BP activities. Thus, creating the BP activity model is an important problem. According to the Activity Based Costing (ABC) method, we have to start defining the resource drivers for the activities. A resource driver is the time needed for performing an activity. An example of the cost model data is represented in Tab. 10.1.

Table 10.1. Activity assignment and Resource Drivers

		Resource driver $\bar{x}$	***Activity number l***								
			1	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
Resource i	***1***	$\overline{x_{1l}}$	2					3		1	
	2	$\overline{x_{2l}}$				2					1
	3	$\overline{x_{3l}}$		1			2				
	4	$\overline{x_{4l}}$			1				4		

The BP cost model has to be defined by the students themselves (for instance, by following the case study example).

10.3. Creating the BP analytical models

Create a set of the BP analytical models for estimating the temporal and cost parameters of the BP.

10.3.1. Bellman's dynamic programming for the BP scheduling

Compute the BP scheduling parameters for the BP graph with unlimited resources, namely:

t_i^*, the earliest time of the event i occurrence;

t_i^{**}, the latest time of the event i occurrence;

r_{ij}, the float start time for the activity ij;

Assume that the duration of the BP activities is deterministic.

10.3.2. Mathematical programming for the BP scheduling

Define the BP scheduling as a problem of mathematical programming for the deterministic routing case (D_R model). Solve the problem using MATLAB.

10.3.3. BP probabilistic analysis

- Complement the BP IDEF3 model with the information about the distributions of the activities duration. Let the distributions be normal. Define the parameters of the distributions (duration expectation $\overline{x}$ and standard deviation *St.dev.*) on your own. An example is given in Tab. 10.2.

Table 10.2. Serving time parameters

		Mean $\overline{x}$, St. Dev V	***Activity l***								
			1	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
Resource i	***1***	$\overline{x_{1l}}$	2					3		1	
		St.dev	0.71					0.84		0.55	
	2	$\overline{x_{2l}}$				2					1
		St.dev				0.71					0.55
	3	$\overline{x_{3l}}$		1			2				
		St.dev		0.55			0.71				
	4	$\overline{x_{4l}}$			1				4		
		St.dev			0.55				1		

- Compute the statistical characteristics of the BP execution time using the central limit theorem. Compute the probability that the BP execution time exceeds the mean value by more than 30%;

- Obtain the statistical characteristics of the BP execution time using Crystal Ball. Find the probability that the BP execution time exceeds the mean value by more than 30%;
- Compute the BP costs expectation using the central limit theorem. Compute the probability that the BP execution costs exceed the mean value by more than 30% (the BP IDEF3 model has to be supplemented with the necessary the cost data);
- Obtain the BP costs statistical characteristics using Crystal Ball. Find the probability that the BP execution costs exceed the mean value by more than 30%.

10.3.4. Queueing networks for the BP analysis

Create a queueing network for the BP in question. Calculate the customer waiting time in the serving systems of the network. Follow the method described in Sec. 5.4.2.1, specifically formulae 5.10 – 5.12. Compute the expectation of the BP execution time using the critical path method described in Sec. 5.3.

10.4. Creating the GPSS BP model

Create both an open and closed GPSS BP model. Find the temporal and the cost BP parameters. What should be done to improve the BP characteristics?

10.5. Creating the ABC model of the BP

Create an ABC model for the BP in question (on your own).

- Embed some four or five cost objects, for instance, products 1, 2, 3 and 4;
- Assign activities to resources (performers) using resource cost drivers;
- Assign the cost objects to activities using activity cost drivers;
- Calculate the activity based costs for the cost objects using the analytical BP model;
- Find out the activity based costs using the GPSS BP model. Create histograms of the costs for all cost objects.

10.6. Creating the BP model in the form of the timed Petri nets

Create the BP simulation model in the form of a timed Petri Nets (with resource limitations). Find the utilization of the BP serving systems.

10.7. Drawing conclusions

Compare the results based on the created BP models. Draw the conclusions.

REFERENCES

[1] **Hammer, M., Champy, J.,** *Reengineering the Corporation: A Manifesto for Business Evolution,* HarperCollins Publishers, New York, 1993

[2] **Banks, J., Carson, G., Nelson, L.,** *Discrete-event system simulation,* 2nd ed., Prentice Hall, NJ, 1996

[3] **Kaplan, R., Cooper, R.,** *Cost and Effect: Using Integrated Cost Systems to Drive Profitability and Performance,* Harvard Business School Press, Boston, 1998

[4] **Mabberley, J.,** *Managing the Future in Financial Institutions: Meeting the Challenge with Better Information,* Price Waterhouse Pitman Publishing, London, 1997

[5] **Kleinrock, L.,** *Queueing systems. Volume 1: Theory,* John Wiley & Sons, New York, 1975

[6] **Sidnev, A.G., Benderskaya, E.N.,** ***et al.*****,** *Modeling Based on Queuing Theory (Textbook),* 2nd ed., / Ed. D.N Kolesnikov, St. Petersburg State Technical University, St. Petersburg, 2004

[7] **Sidnev, A.G.,** *On the Business Process Analytical Modeling* // Transactions № 482 of St. Petersburg State Technical University, St. Petersburg, 2001

[8] **Fishman, G.,** *Principles of Discrete Event Simulation,* John Wiley & Sons, New York, 1978

[9] **Schriber, T.,** *Simulation Using GPSS,* John Wiley & Sons, 1974

[10] **Peterson, J.L.,** *Petri Net Theory and the Modeling of Systems,* Prentice-Hall, 1981

[11] **Harris, J., Stocker, H.,** *Handbook of Mathematics and Computational Science*, Springer, 1998

[12] **Christofides, N.,** *Graph Theory: An Algorithmic Approach*, Academic Press, 1975

[13] **Bunday, B.D.,** *Basic Linear Programming,* Edward Arnold, 1984

[14] **Gill, P.E., Murray, W., Wright, M.H.,** *Practical Optimization*, Academic Press, 1981

[15] **Shannon, E.R.,** *Systems Simulation: The Art and Science*, Prentice-Hall, 1975

GLOSSARY OF THE KEY TERMS

Activity-based costing – A technique for calculating the expenditure associated with the performance of a task or a group of tasks within an organization or on its behalf

Business process – A set of interrelated tasks that have to be performed to accomplish a business objective

Business process reengineering – An approach of changing the operations within the organization to improve the process of achieving the corporate goals

GPSS – The *General Purpose Simulation System* modeling language and simulation system

IDEF – Acronym for the integrated family of Integration Definition methods. IDEF technology supports the strategy which provides a family of mutually-supportive methods for the enterprise integration

IDEF0 – Integration Definition for Function Modeling

IDEF3 – Integration Definition (IDEF) method for Process Description Capture. IDEF3 is designed to help documenting and analyzing the processes of an existing or a planned system

INDEX

APPENDIX A. GPSS PROGRAMS FOR MODELING EXAMPLE BUSINESS PROCESSES

Program 1. This GPSS program corresponds to the IDEF3 BP model shown in Fig. 8.4. The duration of the activities is deterministic. The total BP execution time is also deterministic and equals $T_{\Sigma} = 6.0$. This result complies with one obtained with the Petri net in Fig. 8.5 (sec. 8.3.1).

```
T_11      EQU       2
T_16      EQU       3
T_18      EQU       1
T_24      EQU       2
T_29      EQU       1
T_32      EQU       1
T_35      EQU       2
T_43      EQU       1
T_47      EQU       4

NUM       EQU       1
BP_TIME   TABLE     M1,0,5,20

          GENERATE  ,,,NUM
S_TART    MARK
          SPLIT     1,ACT3
          SPLIT     1,ACT1
          SPLIT     1,ACT4
          TRANSFER  ,ACT2

ACT1      QUEUE     ACT_1
          SEIZE     ACT_1
          DEPART    ACT_1
          ADVANCE   T_11
          RELEASE   ACT_1
          SPLIT     1,ACT7
          TRANSFER  ,ACT5

ACT2      QUEUE     ACT_2
          SEIZE     ACT_2
          DEPART    ACT_2
          ADVANCE   T_32
          RELEASE   ACT_2
          TRANSFER  ,ACT6

ACT3      QUEUE     ACT_3
          SEIZE     ACT_3
          DEPART    ACT_3
          ADVANCE   T_43
          RELEASE   ACT_3
          TRANSFER  ,ASS_J2

ACT4      QUEUE     ACT_4
          SEIZE     ACT_4
          DEPART    ACT_4
          ADVANCE   T_24
          RELEASE   ACT_4
          TRANSFER  ,ASS_J1
```

```
ACT5      QUEUE     ACT_5
          SEIZE     ACT_5
          DEPART    ACT_5
          ADVANCE   T_35
          RELEASE   ACT_5
          TRANSFER  ,ASS_J1

ACT6      QUEUE     ACT_6
          SEIZE     ACT_6
          DEPART    ACT_6
          ADVANCE   T_16
          RELEASE   ACT_6
          TRANSFER  ,ASS_J2

ACT7      QUEUE     ACT_7
          SEIZE     ACT_7
          DEPART    ACT_7
          ADVANCE   T_47
          RELEASE   ACT_7
          TRANSFER  ,ASS_J4

ACT8      QUEUE     ACT_8
          SEIZE     ACT_8
          DEPART    ACT_8
          ADVANCE   T_18
          RELEASE   ACT_8
          TRANSFER  ,ASS_J1

ACT9      QUEUE     ACT_9
          QUEUE     NODE2
          SEIZE     NODE2
          DEPART    ACT_9
          DEPART    NODE2
          ADVANCE   T_29
          RELEASE   NODE2
          TRANSFER  ,ASS_J4

ASS_J1    ASSEMBLE  3
          TRANSFER  ,ACT9
ASS_J2    ASSEMBLE  2
          TRANSFER  ,ACT8
ASS_J4    ASSEMBLE  2
FIN       TABULATE  BP_TIME
          TRANSFER  ,S_TART
*TIMER SEGMENT*
          GENERATE  10000
          TERMINATE 1
```

Program 2. This GPSS program corresponds to the IDEF3 BP model shown in Fig. 8.6 and simulates the closed nonlinear queueing network with the constant number of customers equal to one. The total BP execution time is $T_{\Sigma} = 7.0$, which complies with the result obtained with the Petri net in Fig. 8.7 (sec. 8.3.2).

```
T_11      EQU       2
T_16      EQU       3
T_18      EQU       1
T_24      EQU       2
T_29      EQU       1
T_32      EQU       1
```

```
T_35     EQU      2
T_43     EQU      1
T_47     EQU      4

NUM      EQU      1
BP_TIME  TABLE    M1,0,5,20

         GENERATE ,,,NUM
S_TART   MARK
         SPLIT    1,ACT3
         SPLIT    1,ACT1
         SPLIT    1,ACT4
         TRANSFER ,ACT2

ACT1     QUEUE    ACT_1
         QUEUE    NODE1
         SEIZE    NODE1
         DEPART   ACT_1
         DEPART   NODE1
         ADVANCE  T_11
         RELEASE  NODE1
         SPLIT    1,ACT7
         TRANSFER ,ACT5

ACT2     QUEUE    ACT_2
         QUEUE    NODE3
         SEIZE    NODE3
         DEPART   ACT_2
         DEPART   NODE3
         ADVANCE  T_32
         RELEASE  NODE3
         TRANSFER ,ACT6

ACT3     QUEUE    ACT_3
         QUEUE    NODE4
         SEIZE    NODE4
         DEPART   ACT_3
         DEPART   NODE4
         ADVANCE  T_43
         RELEASE  NODE4
         TRANSFER ,ASS_J2

ACT4     QUEUE    ACT_4
         QUEUE    NODE2
         SEIZE    NODE2
         DEPART   ACT_4
         DEPART   NODE2
         ADVANCE  T_24
         RELEASE  NODE2
         TRANSFER ,ASS_J1

ACT5     QUEUE    ACT_5
         QUEUE    NODE3
         SEIZE    NODE3
         DEPART   ACT_5
         DEPART   NODE3
         ADVANCE  T_35
         RELEASE  NODE3
         TRANSFER ,ASS_J1

ACT6     QUEUE    ACT_6
         QUEUE    NODE1
         SEIZE    NODE1
```

```
          DEPART    ACT_6
          DEPART    NODE1
          ADVANCE   T_16
          RELEASE   NODE1
          TRANSFER  ,ASS_J2

ACT7      QUEUE     ACT_7
          QUEUE     NODE4
          SEIZE     NODE4
          DEPART    ACT_7
          DEPART    NODE4
          ADVANCE   T_47
          RELEASE   NODE4
          TRANSFER  ,ASS_J4

ACT8      QUEUE     ACT_8
          QUEUE     NODE1
          SEIZE     NODE1
          DEPART    ACT_8
          DEPART    NODE1
          ADVANCE   T_18
          RELEASE   NODE1
          TRANSFER  ,ASS_J1

ACT9      QUEUE     ACT_9
          QUEUE     NODE2
          SEIZE     NODE2
          DEPART    ACT_9
          DEPART    NODE2
          ADVANCE   T_29
          RELEASE   NODE2
          TRANSFER  ,ASS_J4
```

Program 3. This GPSS program corresponds to the same IDEF3 model, but the duration of the activities is uniformly distributed (the parameters are given in Tab. 5.5). The network is open. The equivalent network is studied analytically in Sec. 5.4.2.2. The analytical estimate of the BP execution time is $T_\Sigma = 10.7$, whereas the GPSS simulation gives $T_\Sigma = 16.6$.

```
T_11min   EQU       0.77
T_11max   EQU       3.23
T_16min   EQU       1.55
T_16max   EQU       4.45
T_18min   EQU       0.05
T_18max   EQU       1.95
T_24min   EQU       0.77
T_24max   EQU       3.23
T_29min   EQU       0.05
T_29max   EQU       1.95
T_32min   EQU       0.05
T_32max   EQU       1.95
T_35min   EQU       0.77
T_35max   EQU       3.23
T_43min   EQU       0.05
T_43max   EQU       1.95
T_47min   EQU       2.27
T_47max   EQU       5.73

RMULT     555,123,456,792,141,654,37
BP_TIME   TABLE     M1,0,5,20
```

```
         GENERATE  (Exponential(1,0,10))
S_TART   MARK
         SPLIT     1,ACT3
         SPLIT     1,ACT1
         SPLIT     1,ACT4
         TRANSFER  ,ACT2

ACT1     QUEUE     ACT_1
         QUEUE     NODE1
         SEIZE     NODE1
         DEPART    ACT_1
         DEPART    NODE1
         ADVANCE   (Uniform(2,T_11min,T_11max))
         RELEASE   NODE1
         SPLIT     1,ACT7
         TRANSFER  ,ACT5

ACT2     QUEUE     ACT_2
         QUEUE     NODE3
         SEIZE     NODE3
         DEPART    ACT_2
         DEPART    NODE3
         ADVANCE   (Uniform(3,T_32min,T_32max))
         RELEASE   NODE3
         TRANSFER  ,ACT6

ACT3     QUEUE     ACT_3
         QUEUE     NODE4
         SEIZE     NODE4
         DEPART    ACT_3
         DEPART    NODE4
         ADVANCE   (Uniform(4,T_43min,T_43max))
         RELEASE   NODE4
         TRANSFER  ,ASS_J2

ACT4     QUEUE     ACT_4
         QUEUE     NODE2
         SEIZE     NODE2
         DEPART    ACT_4
         DEPART    NODE2
         ADVANCE   (Uniform(5,T_24min,T_24max))
         RELEASE   NODE2
         TRANSFER  ,ASS_J1

ACT5     QUEUE     ACT_5
         QUEUE     NODE3
         SEIZE     NODE3
         DEPART    ACT_5
         DEPART    NODE3
         ADVANCE   (Uniform(6,T_35min,T_35max))
         RELEASE   NODE3
         TRANSFER  ,ASS_J1

ACT6     QUEUE     ACT_6
         QUEUE     NODE1
         SEIZE     NODE1
         DEPART    ACT_6
         DEPART    NODE1
         ADVANCE   (Uniform(7,T_16min,T_16max))
         RELEASE   NODE1
         TRANSFER  ,ASS_J2

ACT7     QUEUE     ACT_7
```

```
          QUEUE     NODE4
          SEIZE     NODE4
          DEPART    ACT_7
          DEPART    NODE4
          ADVANCE   (Uniform(1,T_47min,T_47max))
          RELEASE   NODE4
          TRANSFER  ,ASS_J4

ACT8      QUEUE     ACT_8
          QUEUE     NODE1
          SEIZE     NODE1
          DEPART    ACT_8
          DEPART    NODE1
          ADVANCE   (Uniform(2,T_18min,T_18max))
          RELEASE   NODE1
          TRANSFER  ,ASS_J1

ACT9      QUEUE     ACT_9
          QUEUE     NODE2
          SEIZE     NODE2
          DEPART    ACT_9
          DEPART    NODE2
          ADVANCE   (Uniform(3,T_29min,T_29max))
          RELEASE   NODE2
          TRANSFER  ,ASS_J4

ASS_J1    ASSEMBLE  3
          TRANSFER  ,ACT9
ASS_J2    ASSEMBLE  2
          TRANSFER  ,ACT8
ASS_J4    ASSEMBLE  2
FIN       TABULATE  BP_TIME
          TERMINATE

*TIMER SEGMENT*
          GENERATE  10000
          TERMINATE 1
```

APPENDIX B. GPSS PROGRAM FOR MODELING THE ENTERPRISE CONSULTING RESEARCH

This GPSS program (closed model) of the business process describes the enterprise consulting research, which precedes the implementation of the ERP system.

```
          INITIAL X$PRICEDEVA1,120
          INITIAL X$PRICEDEVA2,100
          INITIAL X$PRICEDEVA3,100
          INITIAL X$PRICEDEVA4,100
          INITIAL X$PRICEDEVE1,40
          INITIAL X$PRICEDEVE2,40
          INITIAL X$PRICEDEVE3,40
          INITIAL X$C_T,6000
          INITIAL X$C_S,15000
          INITIAL X$NEG,9000
          INITIAL X$DG,4500
          INITIAL X$P_FM,30000
          INITIAL X$TR_BPWIN,15000
          INITIAL X$IDEF1X,12000
          INITIAL X$A_B_C,24000
          INITIAL X$TR_ERWIN,18000
          INITIAL X$DOC,24000
          INITIAL X$NUMB,2

T_IME     TABLE  M1,0,100,30
          GENERATE ,,,X$NUMB
*         begin solving the organizational questions        *
****casting the team****
 TOP      MARK
          QUEUE WHOLE
          *QUEUE QDEVA1
          SEIZE DEVA1
          *DEPART QDEVA1
          ADVANCE 12,3
          RELEASE DEVA1
          SAVEVALUE UC_T+,X$C_T
****creating the schedule****
          QUEUE QDEVA1
          SEIZE DEVA1
          DEPART QDEVA1
          ADVANCE 4,1.5
          RELEASE DEVA1
          SAVEVALUE UC_S+,X$C_S
****negotiating****
          *QUEUE QDEVA1
          *QUEUE QDEVA2
          *QUEUE QDEVA3
          *QUEUE QDEVA4
          *QUEUE QDEVE1
          *QUEUE QDEVE2
          TEST E F$DEVA1,0
          TEST E F$DEVA2,0
          TEST E F$DEVA3,0
          TEST E F$DEVA4,0
          TEST E F$DEVE1,0
          TEST E F$DEVE2,0
          SEIZE DEVA1
          SEIZE DEVA2
```

```
          SEIZE DEVA3
          SEIZE DEVA4
          SEIZE DEVE1
          SEIZE DEVE2
          *DEPART QDEVA1
          *DEPART QDEVA2
          *DEPART QDEVA3
          *DEPART QDEVA4
          *DEPART QDEVE1
          *DEPART QDEVE2
          ADVANCE 4,1
          RELEASE DEVA1
          RELEASE DEVA2
          RELEASE DEVA3
          RELEASE DEVA4
          RELEASE DEVE1
          RELEASE DEVE2
          SAVEVALUE UNEG+,X$NEG
****defining the goals****
          *QUEUE QDEVA1
          *QUEUE QDEVA2
          TEST E F$DEVA1,0
          TEST E F$DEVA2,0
          SEIZE DEVA1
          SEIZE DEVA2
          *DEPART QDEVA1
          *DEPART QDEVA2
          ADVANCE 4,1
          RELEASE DEVA1
          RELEASE DEVA2
          SAVEVALUE UDG+,X$DG
*         finish solving organizational questions *         *
*         begin the basic analysis   *
          SPLIT 1,TR1
****performing the function modeling****
          *QUEUE QDEVA3
          *QUEUE QDEVE2
          TEST E F$DEVA3,0
          TEST E F$DEVE2,0
          TEST E F$DEVA1,0
          TEST E F$DEVE3,0
          SEIZE DEVA3
          SEIZE DEVE2
          SEIZE DEVA1
          SEIZE DEVE3
          *DEPART QDEVA3
          *DEPART QDEVE2
          ADVANCE 130,20
          RELEASE DEVA3
          RELEASE DEVE2
          RELEASE DEVA1
          RELEASE DEVE3
          SAVEVALUE UP_FM+,X$P_FM
          TRANSFER ,NO_TR1
****training the stuff to use BPWin ****
     TR1 TRANSFER 0.5,,NO_TR1
          *QUEUE QDEVA3
          *QUEUE QDEVA4
          TEST E F$DEVA3,0
          TEST E F$DEVA4,0
          SEIZE DEVA3
          SEIZE DEVA4
          *DEPART QDEVA3
```

```
          *DEPART QDEVA4
          ADVANCE 40,10
          RELEASE DEVA3
          RELEASE DEVA4
          SAVEVALUE UTR_BPWIN+,X$TR_BPWIN
   NO_TR1 ASSEMBLE 2
*         finish the basic analysis  *
          TRANSFER 0.6,,NO_EA
*         begin the extra analysis   *
****creating the IDEF1X model ****
       EA SPLIT 1,TR2
          SPLIT 1,ABC
          TRANSFER 0.5,,NO_X
          *QUEUE QDEVA3
          *QUEUE QDEVE1
          TEST E F$DEVA3,0
          TEST E F$DEVE1,0
          SEIZE DEVA3
          SEIZE DEVE1
          *DEPART QDEVA3
          *DEPART QDEVE1
          ADVANCE 80,20
          RELEASE DEVA3
          RELEASE DEVE1
          SAVEVALUE UIDEF1X+,X$IDEF1X
NO_X      TRANSFER ,AS_S
****creating the ABC-model****
 ABC      TRANSFER 0.1,,NO_ABC
          *QUEUE QDEVA2
          *QUEUE QDEVE2
          TEST E F$DEVA2,0
          TEST E F$DEVE1,0
          TEST E F$DEVE2,0
          SEIZE DEVA2
          SEIZE DEVE1
          SEIZE DEVE2
          *DEPART QDEVA2
          *DEPART QDEVE2
          ADVANCE 60,10
          RELEASE DEVA2
          RELEASE DEVE1
          RELEASE DEVE2
          SAVEVALUE UA_B_C+,X$A_B_C
NO_ABC    TRANSFER ,AS_S
 TR2      TRANSFER 0.9,,NO_TR2
****training the stuff to use ERWin ****
          *QUEUE QDEVA4
          *QUEUE QDEVE1
          TEST E F$DEVA4,0
          TEST E F$DEVE1,0
          SEIZE DEVA4
          SEIZE DEVE1
          *DEPART QDEVA4
          *DEPART QDEVE1
          ADVANCE 40,5
          RELEASE DEVA4
          RELEASE DEVE1
          SAVEVALUE UTR_ERWIN+,X$TR_ERWIN
NO_TR2    TRANSFER ,AS_S
AS_S      ASSEMBLE 3
NO_EA     TRANSFER ,DI
*         finish the extra analysis  *
*         begin documenting          *
```

```
          *QUEUE QDEVE1
          *QUEUE QDEVE2
          *QUEUE QDEVE3
          TEST E F$DEVE1,0
          TEST E F$DEVE2,0
          TEST E F$DEVA2,0
          TEST E F$DEVA4,0
          TEST E F$DEVE3,0
DI        SEIZE DEVE1
          SEIZE DEVE2
          SEIZE DEVE3
          SEIZE DEVA2
          SEIZE DEVA4
          *DEPART QDEVE1
          *DEPART QDEVE2
          *DEPART QDEVE3
          ADVANCE 40,10
          RELEASE DEVE1
          RELEASE DEVE2
          RELEASE DEVA2
          RELEASE DEVA4
          RELEASE DEVE3
          SAVEVALUE UDOC+,X$DOC
*         finish documenting                *
          DEPART WHOLE
          SAVEVALUE WHOLETIME+,M1
NUM       TABULATE T_IME
          TRANSFER ,TOP
* TIMER SEGMENT                             *

          GENERATE  300000
          SAVEVALUE NUMBCYCLE,N$NUM
          SAVEVALUE COSTDEVA1,(Cost(FR$DEVA1,X$PRICEDEVA1))
          SAVEVALUE COSTDEVA2,(Cost(FR$DEVA2,X$PRICEDEVA2))
          SAVEVALUE COSTDEVA3,(Cost(FR$DEVA3,X$PRICEDEVA3))
          SAVEVALUE COSTDEVA4,(Cost(FR$DEVA4,X$PRICEDEVA4))
          SAVEVALUE COSTDEVE1,(Cost(FR$DEVE1,X$PRICEDEVE1))
          SAVEVALUE COSTDEVE2,(Cost(FR$DEVE2,X$PRICEDEVE2))
          SAVEVALUE COSTDEVE3,(Cost(FR$DEVE3,X$PRICEDEVE3))
          SAVEVALUE WHOLECOST,(Cost1())
          SAVEVALUE WHOLEPART1,(Cost2())
          SAVEVALUE WHOLEPART2,X$WHOLECOST
          SAVEVALUE WHOLEPART2-,X$WHOLEPART1
          SAVEVALUE ALNEGIME,(AC1/X$NUMBCYCLE)


          TERMINATE 1

****SAVEVALUE description******
****COSTDEVA1-E3 - payment of time spend by DEVA1-E3 per project****
*Arg1 - serviceman (facility) utilization muNEGiplied by 1000*
*Arg2 - cost per working time unit (hour)*
*AC1  - working time *
PROCEDURE Cost(Arg1,Arg2) BEGIN

    RETURN (((Arg1#Arg2#AC1)/(1000#X$NUMBCYCLE)));

END;

PROCEDURE Cost1()BEGIN
```

```
      RETURN
(X$COSTDEVA1+X$COSTDEVA2+X$COSTDEVA3+X$COSTDEVA4+X$COSTDEVE1+X$COSTDEVE2+X$
COSTDEVE3+(X$UC_T+X$UC_S+X$UNEG+X$UDG+X$UP_FM+X$UTR_BPWIN+X$UIDEF1X+X$UA_B_
C+X$UTR_ERWIN+X$UDOC)/N$NUM);


END;



PROCEDURE Cost2()BEGIN

    RETURN
((X$UC_T+X$UC_S+X$UNEG+X$UDG+X$UP_FM+X$UTR_BPWIN+X$UIDEF1X+X$UA_B_C+X$UTR_E
RWIN+X$UDOC)/N$NUM);


END;

; Some blocks of the GPSS program responsible for gathering the statistics
are made inactive using asterisks (*) to meet the constraint of the GPSS
World Student Version on the number of blocks (no more than 150)
```

Helsinki University of Technology

Industrial Information Technology Laboratory Publications

TKK-INIT-1	Isto, P. Adaptive Probabilistic Roadmap Construction with Multi-Heuristic Local Planning September 2003
TKK-INIT-2	Sidnev, A., Tuominen, J., Krassi, B. Business Process Modeling and Simulation December 2005

ISBN: 951-22-7928-2 (printed version)

ISBN: 951-22-7929-0 (electronic version)

ISSN: 1459-6458