Comparison of Tools and Languages for Business Process Reengineering

Audris Kalniņš, Dace Kalniņa, Askolds Kālis

The University of Latvia,
Institute of Mathematics and Computer Science,
Rainis blvd 29, Riga, LV-1459, Latvia
email: audris@cclu.lv

Abstract

The paper presents the comparison of main Business Process Reengineering (BPR) tools from the point of view of modeling languages supported by them. One of the tools considered is the GRADE tool developed by IMCS LU. The proposed comparison criteria are language support for the selected basic modeling activities common to most BPR methodologies. The main emphasis is on the semantic properties of modeling languages. The results of comparison are summarised in a table.

1. Introduction

The term **Business Process Reengineering (BPR)** has been actively used for at least 4 years, since the publishing of the book by Hammer and Champi [1]. Nearly every company has declared to do some BPR activities recently. This has created a booming market for various **BPR support tools** - according to the estimates by Gartner Group [2] - about 100 million USD per year, with the annual rate of growth at 30%. There are about 40 BPR tools on the market.

Some industry-level evaluations of BPR tools are performed regularly [2-8], the most respected of which are the regular annual evaluations, performed by Gartner Group [2,3,4,5]. The main goal of Gartner reports is to serve as a tool buyers' guide of high quality to the US market, therefore the main emphasis there is on the tool-technical criteria. Only the tools highly used in USA are included in Gartner reports. The goal of the vast BPR tool survey by Bach, Brecht et al [6] is also an industrial evaluation, however a deeper analysis of modeling facilities, including metamodels, is provided.

The goal of this paper is to compare the underlying principles of modeling incorporated in tools. These modeling principles of a tool determine its **modeling language** and so our main emphasis is to compare modeling languages for BPR. The authors of this paper are within the team developing the BPR tool GRADE [9]. Therefore one of the goals of the paper is to compare GRADE to other BPR tools from the modeling language point of view.

2. Comparison principles

2.1 *The considered area of BPR*

The business process or more precisely the **business system** is understood in this paper in the broadest possible way. BPR is considered as an activity corresponding to its classical definition given by Hammer and Champi[1], namely "fundamental rethinking and redesign of business processes to achieve dramatic improvements in critical contemporary measures of performance such as cost, quality, service and speed." There may be a lot of ways to achieve this goal or, in other words, different BPR methodologies (see e.g. [10-14]). Except for some very specific ones [15], the general scheme is nearly the same for all methodologies:

- the model of the existing business system is built (***as-is* model**). The model should include various aspects of the system, such as:
    - its main business functionality, i.e. the system inputs and outputs and the main functions yielding these outputs
    - organisational structure of the system
    - the exact internal behaviour aspects of business system - when and by whom each function in the system is performed (called also workflow or work practice)
    - the general business principles and goals of the system
    - the low-level economic criteria of the system, mainly timings and costs
- the *as-is* model is analysed and some improvements are proposed
- the proposed system improvements are documented as one or more ***to-be* models**
- the *to-be* models are compared via logical evaluation, static analysis and dynamic simulation
- the best of the *to-be* models is "implemented" in the business system.

Frequently the reengineering of a business system requires also the reengineering of its IT system. Therefore there is another view, that BPR is not a standalone activity, but a **requirements definition** phase for IT system development or redesign. This view puts a little bit different emphasis on BPR efforts - the most important issue is a seamless transfer of requirement data from the business model into a high-level design of the IT system. Due to lack of space we don't consider here this second approach.

During the evaluation of tools we can restrict ourselves to tool (more precisely, their language) support for building models of a business system and evaluation of the models via logical reasoning, static analysis and simulation experiments. The following criteria are selected as crucial for this model building process and are used for the language comparison in section 4: **basic behaviour description,** its **quantitative aspects, allocation of tasks to performers, data based behaviour, basic costing, advanced timing** features and **higher business goals.**

2.2 *About modeling language and its internal qualities*

We have to specify what exactly is meant by a modeling language because in some tools the language elements are mixed up with technical details used to enter these elements.

Nowadays all BPR tools are graphical. We assume, that the language includes all types of **diagrams** together with directly enterable auxiliary elements - **tables, texts** and **objects** enterable via dialog boxes into the tool repository. Derived elements such as reports are not included in the language.

The modeling language has its **syntax** which defines what elements are permitted in each diagram type, how the elements may be linked, what text may be placed within elements etc. The skeleton of such a syntax in most cases can be best represented by a **metamodel** - now most probably as an OMT [16] or UML [17] class diagram. The graphical syntax and metamodels of the considered languages show a great deal of similarity therefore we don't use syntactic aspects in evaluation.

Most BPR methodologies require multiple manual inspection and evaluation of a business system model (especially of *as-is* model) by experts with various background. This requires that any business model should be easy and unambiguously readable by a person. Consequently, there must be a precise **modeling semantics** of a business modeling language, which determines:

- what kind of real world elements may be represented by the given model element;
- what facts about the involved model elements does the given modeling construct express;
- what possible dynamic behaviour of the involved model elements corresponds to the given dynamic modeling construct.

The modeling semantics, especially its precision level, is one of the main evaluation factor for modeling languages. The level **of formality of modeling semantics** of a language evaluates the accuracy of the corresponding language documentation, i.e., how exact answers to the above mentioned questions about the meaning of language constructs can be obtained there.

Since most of BPR tools support simulation, there is also a **simulation semantics** of the business modeling language. As a rule, it is defined only for a subset of the modeling language. This semantics is always precisely defined (by the simulator tool itself!), but two questions arise here:

- how simple and natural is the simulation semantics;
- how exactly it corresponds to the modeling semantics.

We will assess the **consistency of simulation semantics to modeling semantics,** since this consistency actually guaranties that simulation results are meaningful for the model.

Another internal factor of a language is its ability to cope with large models, i.e., the **model structuring** facilities in the language.

2.3 Tools and languages selected for comparison.

As it was already mentioned, this comparison is made by some of the authors of the **GRADE** tool [9], being developed by joint efforts of IMCS LU and Infologistik GmbH (with earlier participation of the company "Dati"). Therefore the comparison inevitably is "GRADE to other tools". The other tools are taken from the top-ranking ones according to the Gartner reports [2,3]. Since we analyse tools from the modeling language point of view, tools supporting the same language are grouped together. The following tools/languages have been selected:

- **ARIS** by IDS Prof. Scheer (the ARIS tool is labelled by Gartner as "far and away leader in the BPR tool market" [3])
- **IDEF** language series (IDEF0, IDEF3, IDEF1x) group: System Architect by Popkin Software, BPwin by Logic Works, Procap/ Prosim by KBSI, Design/IDEF by Meta Software, Wizdom Works by Wizdom System
- **Extend**/BPR by Imagine That.

The size of the paper does not allow to include more tools/languages for comparison. We only want to remark that for some tools ranked high by Gartner, for example Oracle business process modeler [18], the corresponding modeling language is very narrow.

3. Short description of modeling languages used by tools

We have to give at least a very short description of the selected modeling languages to be able during comparison refer to the main concepts of each language. Since the main area of comparison is the dynamic behaviour representation most attention is devoted to this aspect. Fig.1- fig.4 show the brief examples of all languages describing the same fragment of a business system - an initial fragment of order processing in a company.

3.1 GRADE and GRAPES-BM language

A sufficiently comprehensive description of **GRAPES-BM** business modeling language supported by the **GRADE** tool (namely, its version 3.0) has been given in the previous DB&IS conference [9]. From this version the main application area of GRADE has been meant namely the BPR. Now the GRADE version 4.0 has been released. The main business modeling language features in GRADE 4.0 are the same as in GRADE 3.0, though the tool quality, especially its user interfaces, have improved significantly. The main new language feature in this version is the support of **Object Modeling Techniques** with **Class diagrams** either in the classical OMT form [16] or in the new UML 1.0 [17] form. Class diagrams, according to almost all modeling methodologies, should be used for capturing the first overall view of a business system. GRADE 4.0, to a certain degree, has integrated Class diagrams with other modeling aspects. Class diagrams can be "reverse-engineered" from ER models, relevant parts of Class diagrams may be used for generating organisational structure (ORG) or Data Type Definition (DD) diagrams.

Let us briefly remind the key elements of GRAPES-BM. The main type of diagram for describing process behaviour is **Business Process (BP)** diagram, which contains
- **tasks** (depicted as rounded rectangles)
- **events** (depicted as arrows linking tasks)

Tasks are triggered by incoming events, and they produce outgoing events upon their completion. The precise triggering scheme of a task is described by its **triggering condition** (a Boolean expression over incoming event names). **Organisation structure** of the business system is represented in a special **ORG** diagram, but its elements may be referenced as performers of a task via its **performer specification expression**. Tasks may have **decisions** (represented as hexagons) attached to represent branching processes, decisions may be probability based or even event data based. **Timer** symbols represent model dependency upon time points.

Data **store** symbols (parallelograms) and data object symbols represent the links of tasks to data model of the system (described separately via ER and data definition (DD diagram). Tasks have predefined attributes, such as **duration** and labour **cost,** and may have an unlimited number of user-defined attributes.

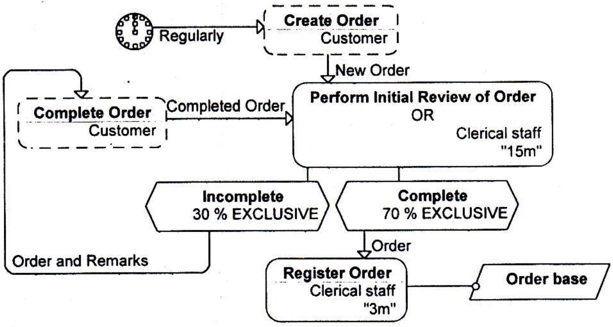


Fig.1. Order processing in GRADE

Complicated tasks are refined via subordinated business process diagrams. The hierarchy of BP diagrams together with other diagrams (ORG, ER, DD etc.) and tables (Event table, Attribute tables etc.) form the business model of a system. The structure of the business model is depicted by model tree.

GRAPES-BM business modeling language has a precise modeling semantics. One of the key elements of the semantics is the concept of **transaction** for natural description of the behaviour of partitioned business activities. Static analysis features of GRADE support the finding of critical path with respect to process execution time. But the main tool for decision making (**what-if** analysis) during BPR is the GRADE simulator, which supports automatic obtaining of main business process statistics: cycle times, total costs, length of queues, utilisation of performers etc.

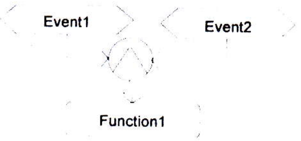
Nearly any kind of a statistical evaluation of a process may be obtained via user defined attributes of tasks and event data. The simulation semantics in GRADE is completely compliant to its modeling semantics.

3.2 ARIS modeling language

The ARIS tool by IDS Prof. Scheer has its own modeling language, which has no specific name. The current evaluation is based on ARIS version 3.1a and fragments of the upcoming version (ARIS Easy Design 1.0).

The modeling language contains about 50 diagram types. Each of the diagram types has a list of permitted objects, many of which are common to several diagram types.. The current number of object types is about 110 and is expected to grow, but they are separated into groups with common properties. Each diagram type defines also permitted links between object types. A link may have types defining its intended semantics and attributes. Any object has a fixed list of attributes (e.g. *function* has about 140 attributes), part of which are common to all objects and part - object-specific. Among the common ones there is a fixed number of user definable attributes. Any object instance has its definition entry (directly in the repository) and may have occurrences in several diagrams. A diagram may be assigned to an object in another diagram as a refinement of this object.

Some of the diagram types have a well-defined semantics. These are diagram types describing system behaviour (**eEPC, PCD** and **event** diagrams), organisation structure (**organisational** chart) and data modeling (several kinds of **ER** models). The most important of them for BPR is the eEPC diagram. It contains **functions** (rounded rectangles) and **events** (hexagons). A function is triggered by one or more events and produces one or more events upon completion. Event and function symbols form an alternating chain linked via control flows - dashed arrows. If more than one event is involved, they must be combined using **rules** - one of the AND (∧), OR (∨) or XOR operators, enclosed in a circle. For example, the following construct represents an AND triggering condition involving events *Event1* and *Event2*:



Similarly, the branching at function completion is represented by an OR or XOR symbol following the function. Rules may form a net, thus complex conditions are expressed. The precise semantics of such constructs is defined only in the simulation context. This semantics does not include data with events - they are just named control flows. Use of performers is specified by "copying" the corresponding elements from the ORG chart into eEPC and linking them to a function.

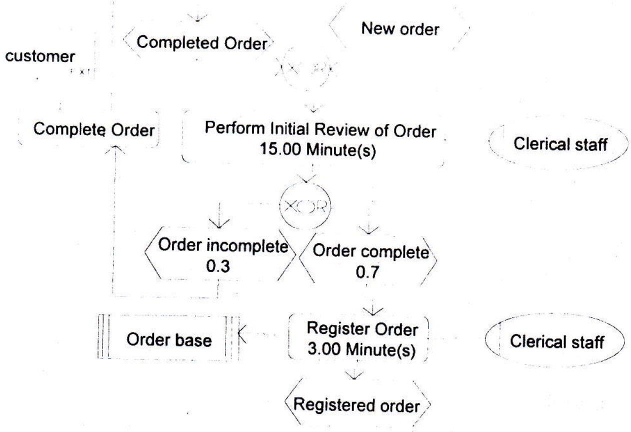


Fig.2. Order processing in ARIS

A number of the object attributes may be made visible in the diagram, e.g. function processing times (durations) and event probabilities after branchings.

PCD diagram actually is the same as eEPC, but with a predefined layout. When a function is refined, the refinement eEPC is assigned to it. The inter-level link is via equally named events (precisely defined at simulation level), it is up to user to define the links correctly.

eEPC and PCD diagrams may be analysed statically to find a maximum time or cost (equivalents of critical path). Simulation takes into account eEPC, PCD, event and organisational diagrams with main statistics obtained automatically. Only the predefined time and cost attributes (including various their components) are taken into account. Any user attributes are simply ignored at simulation. Thus the part of the large modeling language used in simulation is relatively small.

3.3 Family of IDEF languages

The family of IDEF modeling languages has been promoted by KBSI in the USA. The IDEF0 language is an updated version of the SADT techniques [22] proposed by D.Ross in 1976 for structured analysis of systems. It is accepted in the USA as a federal standard [23]. IDEF1x language is an extended version of the traditional ER modeling, including some object-oriented elements. It is also accepted as a US standard [24]. The IDEF3 language [25], the most relevant for BPR goals, is not a US standard.

All three languages are supported by a number of tool vendors. System Architect [26] by Popkin Software, BPWin by Logic Works, PROSIM toolset [27] by KBSI support all of them, Design/IDEF by MetaSoftware and Wizdom Works by Wizdom Systems support IDEF0 and IDEF1x. The language standards are well defined for each of the IDEF language separately, but links between them, though much needed for BPR activities, are left to tool vendors.

The IDEF0 language contains one type of diagram, which is a sort of data flow between functions. It may be used as an overview of a business system. Normally functions in IDEF0 are refined further by IDEF3 diagrams to express the behaviour level.

IDEF3 has two types of diagrams, of which the most used are **process diagrams**. These diagrams contain **units of behaviour (UOB)** - a direct equivalent of tasks in GRADE or functions in ARIS. UOBs are linked by arrows - **precedence links**. According to the semantics of IDEF3, the precedence is a pure control flow - it expresses the fact, that the second UOB cannot start before the first is completed. As in ARIS, links may be only combined using **junctions** - AND, OR and XOR symbols. If junctions are in the role of a triggering condition (*fan-in* junctions), the natural triggering semantics is used. The symmetric *fan-out* junctions (placed after UOB) express either branching (OR or XOR case) or parallel subprocessing (AND case). Any diagram in IDEF3 may have only one input or output. Therefore junctions must be combined in a "structural" way (each *fan-out* must have the corresponding *fan-in*). If a loop in the event flow is present, it must be coded by a **referent** (an equivalent of *goto* or *call*). There are more exotic features in the formal definition of IDEF3 [25], but all tools use just this. On the other hand, to have a meaningful simulation feature, there must be at least some attributes of objects - durations of

UOBs, probabilities of OR branchings, input frequencies. Since these things are absent at IDEF3 definition level, they are simply added as simulation attributes in IDEF3 tools. UOBs may be refined in a natural way (very simple inter-level links because of one input and output).

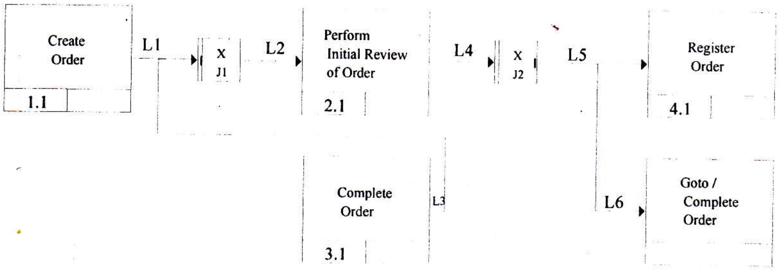


Fig.3. Order processing in IDEF3

Durations and probabilities actually can be entered in the model, but they are not visible in the diagram. Performer specifications are absent in the language, in tools they are shown as swimlanes. Thus IDEF3 language, even with typical adds-on in tools, cover only the very basics of business processes.

3.4 *Extend language*

Extend by Imagine That is a very specific tool with the main emphasis on simulation [12,28]. It is based on a textual low-level simulation language ModL, and its graphical user interface in essence is a set of graphical macros over the ModL language. These graphical macros, called **blocks** in Extend, are organised in problem-oriented **libraries** (supporting both continuous and discreet event simulation). There is only one type of diagrams, where blocks and lines connecting them are placed. Each block type has a fixed set of input and output **connectors**, and an output connector of a block may be linked to an input connector of another one. Since Extend probably has been earlier mostly used for continuous simulation, an Extend diagram strongly reminds a preliminary design of a device in an electronics lab, with component sockets linked via temporary cables. A linked group of blocks may be defined as a hierarchical block (to be stored also in a library), thus refinement is available.

Only the behaviour aspect of a business system may be described. Blocks such as *Import* (generating incoming stream of work), *Operation* (an equivalent of task), *Decision* (branching) are available. The *Operation* supports also the simplest default triggering condition AND for up to three inputs. A more complicated triggering condition may be defined by *Batch* block (and its pair *Unbatch*). An OR triggering condition is expressed by *Merge* block. There are no built-in queues in operations, queues must be specified explicitly by *Queue*, *Stack* or *Repository* blocks (with similar properties).

The simulation semantics is a typical **item** (discrete event) flow. Items are generated by *Import* blocks, stored in *Queue* or *Stack*, processed (delayed for the specified interval) in an *Operation* and so on. Items may have user-defined attributes.

Performers are specified via *Labour Pool*, but their use in tasks is defined dynamically according to the same item semantics (as an additional "triggering event"). After task completion performers must be "unbatched" and returned to pool. There are few predefined statistics, but nearly any kind of statistics can be programmed and displayed dynamically, including the chart form.

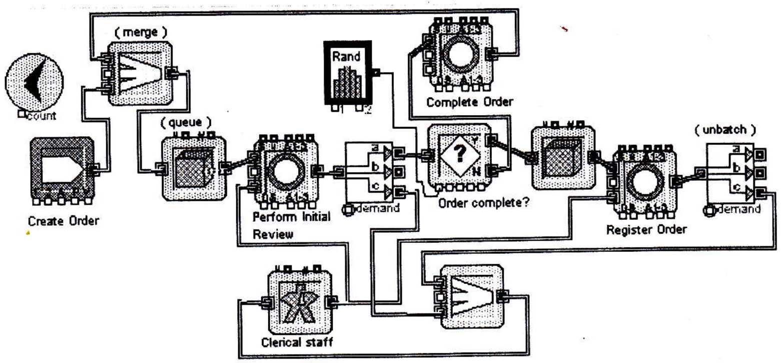


Fig.4. Order processing in Extend

Process models of limited size seem to be quite readable in Extend, but representing large business systems may be problematic this way.

4. Comparison of modeling languages

In this section we perform the actual comparison of the selected modeling languages according to the criteria, found out in section 2. Each subsection is devoted to one or more related criteria. The section concludes with a table which contains a row for each criterion and where authors have tried to perform some formal evaluation.

4.1 The degree of formality of the modeling language

The level of formality is a quality defined not only by the tool, but also by its documentation (manuals and on-line helps). The main issue here is the **formality of modeling semantics** of language since the syntax is well-documented as a rule.

For **GRADE** the modeling semantics of its modeling language is described in language reference manual [19], with approximately the same topics available in on-line help, and in a more introductory language guide [20]. The modeling semantics of all GRAPES-BM concepts is completely defined in these documents, both in cases where just relations between objects are described in diagrams and where the intended dynamic behaviour is described.

In **ARIS** the modeling semantics is described in Methods manual [21] and in on-line helps of ARIS tool. Since largest part of the great number of ARIS diagram types are descriptive, the objects, their possible connection and their attributes for each diagram type are described in tables - in a sort of tabular equivalent to the metamodel. A number of semantic questions remain uncovered - for example, what is the meaning of various connection types between org-units in org-charts or connection types between functions and org-units in *eEPC*. The few diagrams which do have the dynamic semantics - *eEPC* and *PCD* (with event diagrams and org-charts as a support) have their precise semantics described only in the context of ARIS simulation [29]. For example, the placement of queues, the precise semantics of inbound (triggering) AND-rule, the semantics of use of performers by functions (i.e. org-objects linked via *executes* link to functions) are clearly described only for simulation. However, these concepts are essential for understanding the dynamics of any model. Thus, from the complete set of the ARIS documentation, the intended modeling semantics of most model elements can be guessed, but not always so easy.

For **IDEF** languages, namely, those of main interest for BPR - IDEF0 and IDEF3, there are very precise syntax and semantics definitions [23,25]. But the language version (especially, IDEF3) implemented in main tools differs from the standard. The IDEF3 standard [25] contains a number of types of *precedence links* between UOBs (tasks), but the main tools implement another, more practical types. The precise low-level description of UOBs - *elaboration* uses a logic based language in the standard, while in practice a new description kind - the *Simulation information* is added (only informal elements are retained in elaboration). Vital elements of process behaviour - use of performers, task durations etc. are included in simulation information, which slightly differs in various tools. Therefore the actual semantics description of IDEF3 for tools is not so clear.

There is no modeling language other than simulation one in the **Extend** tool, therefore the Extend language is too narrow for modeling purposes.

A strongly related issue is the **consistency of modeling and simulation semantics**. For **GRAPES-BM** the simulation semantics is explicitly defined as a part (a very large one) of the modeling semantics. For **ARIS** only a relatively small part is simulatable, but there are no inconsistencies because a lot of details are defined only in the simulation context. For **IDEF** languages the formal definitions of modeling semantics are completely not simulation oriented, therefore the *Simulation information* added in tools sometimes leads to a slightly inconsistent semantics. For **Extend** there is only simulation semantics.

4.2 Description of basic behaviour aspects

The **basic behaviour elements** - tasks, events linking them, triggering conditions of tasks, behaviour branching after a task - are described in a similar manner in all languages considered. Namely, these aspects are illustrated for all the languages in the simple example present for all discussed languages in section 3. The description facilities for the basic behaviour seem to be acceptable for all the languages.

Only one difference should be pointed out. In GRAPES-BM a triggering condition of a task is defined by textual expression inside the task symbol, but in ARIS and IDEF3 special graphical connector symbols are used to express triggering conditions. In simple cases the graphical notation for triggering condition may be more readable, but it completely prohibits correct definition of semantics when data are to be carried along with events.

4.3 Quantitative aspects of basic behaviour

The next question refers to quantitative aspects of behaviour - such as duration of each task and probabilities of output branching of a task. They are adequately provided as basic modeling elements both in **GRAPES-BM** and **ARIS**. In the definition of **IDEF3** language standards there are no such aspects, but IDEF3 based tools provide these things as simulation attributes in a manner specific to a tool, not always all features are provided. Basic blocks in **Extend** provide these characteristics, however sometimes with a little "programming".

The specific subtopic is the **basic costing**. In **GRAPES-BM** there is a predefined task-duration-dependent *cost* component. In addition, an appropriate task type (Attribute Table) for a model may be defined, containing an arbitrary number of numeric attributes. In particular, these attributes may represent cost components according to the selected costing scheme. Then attribute values (constants or expressions) corresponding to the relevant cost components may be assigned to each elementary task in the model. The automatic totalling of cost attribute values at simulation occurs for meaningful fragments of the business model - transactions, thus total costs of the relevant business processes are obtained. The actual formula for cost totalling may also be defined by the user, by means of another attribute table used for transactions. In **ARIS** each function has a very rich fixed list of cost attributes, which can be used to obtain total values either via static analysis (makes sense for processes with simple behaviour) or via simulation (for any kind of processes). Though GRADE and ARIS approaches are totally different, they both provide the necessary support in practice. For **IDEF** languages another popular costing scheme - Activity Based Costing (ABC) is frequently implemented in tools, but not for IDEF3 - the main business modeling language in IDEF (ABC feature is typically defined only for IDEF0). Task costs can be "programmed" in **Extend**, via *item* (message) attributes, but there are no provisions in block libraries to support this, which makes this programming rather cumbersome.

4.4 Allocation of tasks to performers

Both in **GRAPES-BM** and **ARIS** this feature is based on one or more org-charts in the business model and references from tasks to necessary performers in *Business Process* (GRAPES-BM) and *eEPC* (ARIS). In basic cases the support is adequate for both languages. There is one essential difference. In **GRAPES-BM** the reference is via a textual performer expression inside the task, in **ARIS** a link to the occurrence of the corresponding org-element in the *eEPC* is used. The textual performer expression can be quite complicated, e.g. *P1 AND P2 OR P1 AND P3*. The graphic form in **ARIS** can support more relations beside the standard *executes*, but it is impossible, e.g. to specify even alternative performers such as *P1 OR P2*. There is no performer specification in **IDEF3** (the *resource* used in some IDEF3 simulators is something similar to it). In **Extend** you can specify performers dynamics for simulation via *Labour pool* block, but it is not a very readable way.

4.5 Advanced data-based behaviour description

In **GRAPES-BM** this style of description is based on event attributes which can be used in formula-based decisions, variable task durations and costs etc. The main use of this feature is for simulation, but it has meaning also for modeling, e.g. for looping processes. There is no such possibility in **ARIS** where event links to data objects in *eEPC* have different semantics, not taken into account for function/event dynamics. The control-flow nature of links in **IDEF**, naturally, forbids this feature in **IDEF3.** Data-based behaviour description is available in **Extend** where *items* can have a dynamically defined list of attributes, used in decisions, computations etc.

It should be noted, that just the use of data in the description of business process control enables one to model and simulate adequately an important class of business systems such as manufacturing, logistics and transportation, where the system behaviour significantly depends on its "control data".

Another aspect of data is time control facilities - **advanced timing.. GEAPES-BM, ARIS** and **Extend** - all support interval timing for incoming events. But only **GRAPES-BM** supports full calendar-based timing in process description (fixed time in **ARIS** *eEPC(instance)* is something another).

4.6 Description of higher business goals.

This feature is best developed in **ARIS**. There is *Objective* diagram type containing *Objectives* (with their hierarchy), *critical factors* (again with hierarchy) related to *objectives* and *functions* supporting *objectives*. Though this diagram is just an "object diagram" in the sense of OMT, it nevertheless gives a description of the situation. Other languages support the feature at a quite low level. In **GRAPES-BM** there are *Objectives* and *Constraints* sections of task details diagram, however there are no references between them. If necessary, one or more *Class* diagrams may be used to link tasks, objectives, critical factors and constraints by some meaningful relations. In **IDEF3** elaborations of *UOB* (tasks) also have textual *Facts* and *Constraints* section. A special language **IDEF5** has been planned for descriptions of business system goals but not implemented in the tools considered here. **Extend** has nothing in this area.

4.7 Facilities for description of large systems

All the languages considered have **model structuring** facilities for behaviour description - any task may be refined by a subordinate diagram. The main issues to be discussed are the organising of diagrams and inter-level consistency.

In **GRAPES-BM**, the diagrams of a model are organised by the model tree. Normally it depicts also. the natural task refinement hierarchy. Inter-level consistency of business process diagrams is strictly defined via *event/referenced task* pairs in subordinated diagrams. It seems very acceptable from the reader's point of view, but sometimes create difficulties at modifications of large models.

In **ARIS** subordinate diagrams are assigned or linked to objects in higher level diagrams. This scheme, in general, is applicable to all types of diagrams. For *eEPC*, this yields the natural hierarchy of function refinements. There is also a function hierarchy diagrams describing function hierarchy tree-wise, but there is no requirement for this hierarchy to correspond to the *eEPC* hierarchy (normally it should be so). In addition there is an external hierarchy of diagrams in the repository, organised via groups, but this is mere a technical grouping, similar to Windows95 folders. The inter-level links for *eEPC* is defined via corresponding events, while the role of *process interfaces* (an object similar to referenced tasks in GRAPES-BM) is unclear. All this makes structuring features powerful, but a little unsafe, dependent on the good will of the model builder.

In **IDEF3** there is a strict concept of decomposition of *UOB* by another process diagram. Interfaces are trivial since each function has one entry/exit.

There is no structure organising facilities in **Extend**. Inter-level links are defined via named connectors.

4.8 Summary of comparison results

The table 1 gives the formal comparison results ("+" - good support, "*" - medium support, "-" - bad support).

	GRAPES-BM	ARIS	IDEF	Extend
Formality of modeling semantics	+	+	+	-
Consistency of modeling and simulation semantics	+	+	*	*
Basic behaviour description	+	+	+	+
Quantitative aspects of basic behaviour	+	+	*	+
Basic costing	+	+	*	-
Allocating tasks to performers	+	+	-	*
Data based behaviour	+	-	-	+
Advanced timing	+	*	-	*
Higher business goals	-	+	*	-
Model structuring	+	+	+	*

Table 1. Results of language comparison

The results show that from the language point of view **GRAPES-BM** and **ARIS** are quite similar and both are better than **IDEF** and **Extend**. The main difference between **GRAPES-BM** and **ARIS** seems to be that

- **ARIS** tends to cover a broader area but with lesser semantic precision
- **GRAPES-BM** is the most semantically rich and concise language in the BPR area.

5. Conclusions

The comparison of modeling languages for the main BPR has revealed many similarities, but also several different approaches to BPR language design. Yet the main conclusion may be a couple of issues having not found a good solution in any of languages:

- though OMT (now UML) techniques is supported by nearly all tools, no language has offered natural links between classic Business modeling and Object modeling
- higher level business goals have not got a real semantic link with other parts of modeling
- no language ensures a true business rule support (see, e.g., [30]). There has been an attempt to use rule based principles in GRAPES [31], but it also has had a rather procedural approach.

References

[1] Hammer M., Champy J. Reengineering the Corporation, Harper Collins, New York, 1993.

[2] Kleinberg K. BPR Tool Market - 4Q97: Visionaries and Niche Players [October 30, 1997 • Gartner Analytics •00052235 •.]

[3] Kleinberg K. BPR Tool Market - 4Q97: Leaders and Challengers [October 30, 1997 • Gartner Analytics • 00052218 •]

[4] Kleinberg K BPR Tool Market: Mid-1996 Update, Part 1,2 [July 01, 1996 • Gartner Analytics • 00030549 •]

[5]. Kleinberg K. The BPR Tool Market: Broadening and Rising [July 07, 1995 • Gartner Analytics • 00024641 •]

[6]. Bach V., Brecht L., Hess T., Oesterle H. Enabling Systematic Business Change, Verlag Vieweg, Wiesbaden, 1996.

[7] REUSE-M. A Project from the European Commission. EBIT, ESSI, IGD, October 1996, http://www-a5.igd.fhg.de/reuse_m/reuse_m_home.html

[8] BPR International Survey. ESPRIT project GLOBE, 1996, http://www.txt.it/globe/survey.htm

[9] Kalnins A., Barzdins J. et al. - Business Modeling Language GRAPES-BM and Related CASE Tools - .Proceedings of Baltic DB&IS'96, Institute of Cybernetics, Tallinn, 1996.

[10] Davenport Th. Process Innovation. Reengineering work with Information Technology. Harvard Business School Press, Boston, 1993

[11] Ferstl O.K., Sinz E.J. Der Ansatz des Semantischen Objectmodells (SOM) zur Modellierung von Geschaeftsprozessen,in: Wirtschaftsinformatik Vol. 37,1995, No.3,p.209

[12] Hansen G.A. Automating Business Process Reeingineering, Prentice Hall, 1997

[13] Oesterle H. Business in the Information Age. Heading for new processes, Springer, Heidelberg, 1995

[14] Brenner W., Keller G. (Eds): Business Reengineering mit Standartsoftware, Campus Verlag, Frankfurt, 1995.

[15] Action Inc., Action Workflow Analyst User's Guide, version 1.0, Alamanda/CA 1993

[16] Rumbaugh J. Object-Oriented modeling and Design, Prentice-Hall, 1991

[17] Unified Modeling Language (UML) Notation Guide v. 1.1 Rational Software

Corporation, 1997, http://www.rational.com/uml/documentation.html
[18] Designer-2000. A Guide to Process Modeling. Oracle Corp., 1995.
[19] Kalnins A. GRADE Version 4.0 Business Modeling Language Reference Manual.Infologistik GmbH, Munich, 1997
[20] Barzdins J., Tenteris J., Vilums E. Business Modeling Language GRAPES-BM (Version 4.0) and its Application. Infologistik GmbH, Munich, 1997
[21] ARIS Methods Manual, version 3.1a,IDS Prof. Scheer GmbH,1996
[22] Marca D.A., McGowan C.L. SADT: Structured Analysis and Design Techniques, McGraw-Hill, NY, 1988
[23] Integration Definition for Function Modeling (IDEF0). FIPS Publication 183, National Institute of Standards and Technology, 1993
[24] Integration Definition for Information Modeling (IDEF1x). FIPS Publication 184, National Institute of Standards and Technology, 1993
[25] Information Integration for Concurrent Engigeering (IICE) IDEF3 Process Description Capture Method Report, KBSI, Texas, 1995
[26] SA/BPR Product Data. Popkin Software & Systems, 1997 http://www.popkin.com/prods/sa_bpr/sa-bpr.html
[27] ProCap/ProSim Process Modeling and Simulation Support., KBSI, 1997, http://www.kbsi.com/products/prosim.html
[28] Extend Performance modeling for decision support Demonstration version of Extend 3.1, Extend+BPR 3.1, and Extend+Manufacturing 3.1, Imagine That, Inc, 1995 http://www.imaginethatinc.com/itdemo.htm
[29] ARIS Simulation Manual, version 3.1a, IDS Prof. Scheer GmbH,1996
[30] GUIDE Business Rules Project, Final Report, GUIDE International, 1997, http://www.guide.org/ap/apbrall.htm
[31] Barzdins J., Barzdins G. and Kalnins A. Rules-Based Approach to Business Modeling, Proceedings of SEKE'95, pp. 164-165, 1995.