24. How to Model Business Processes with GPN

Günter Schmidt and Oliver Braun
Department of Information and Technology Management, Saarland University, PO Box 15 11 50 D-66041 Saarbrücken, Germany [gs|ob]@itm.uni-sb.de

Organizations today face increasing pressure to reduce time to market, i.e. to improve the design and the operations of business processes in terms of lead time and meeting due dates. Formal analysis using a mathematical graph-based approach can help to achieve this kind of improvement.

We will apply business graphs to scheduling workflows in terms of time-based optimization. We will concentrate on performance measures like completion time, flow time and tardiness. From a business process network we derive two types of directed graphs, one representing the task net (task graph) and the other one representing the resource net (resource graph). In the task graph a node is representing a task and its duration and arcs are representing different kinds of precedence constraints between tasks. The resource graph is similar to a Petri net and represents resource constraints and flows of jobs.

In order to compute optimal or near-optimal workflow schedules the algorithms have to relate to the structure of the business graphs. We will show that a variety of data structures commonly assumed in modern scheduling theory can be represented within the framework of business graphs. Based on these data structures specific scheduling algorithms to optimize time-based performance measures can be applied with the objective to reduce time to market.

1. INTRODUCTION

Modelling languages are required for building models in various application areas. We shall focus on the management of business processes which require the modelling of time-based activities for planning and scheduling purposes. A business process relates to a stepwise procedure for transforming some input into a desired output while consuming or otherwise utilising resources. Some general examples for business processes are: 'Product Development', 'Procurement', or 'Customer Order Fulfilment'; some more special examples would be 'Claims Processing' in insurance companies or 'Loan Processing' in banks. The output of a business process should always be some kind of achievement (goods or services) which is required by some customer. The customer might be either inside or outside the organisation where the process is carried out (Schmidt, 2002).

Two major aspects of business process management are planning and scheduling. Planning is concerned with determining the structure of a process before it is carried out the first time. Scheduling in turn is concerned with assigning resources over time to competing processes. Both planning and scheduling focus on dependencies among transformations within one process or between different processes. Malone and Crowston (Malone and Crowston, 1994) formulated the need

to merge the paradigms of business process planning and business process scheduling concerning the management of dependencies among transformations. The reason is not only to increase the potential of applying results from planning and scheduling theory to the management of business processes but also to consider the relevance of problems arising from business process management for a theoretical analysis within these research areas.

Planning and scheduling require a specialised model of the business process. To build the required process model we base our analysis on Generalised Process Networks (GPN) (Schmidt, 1996), a graphical language related to CPM type of networks (Slowinski and Weglarz, 1998). We will show that GPN are expressive enough to formulate problems related to planning and scheduling of business processes within the same framework. Doing this we use a semi-formal presentation of the syntax and the semantics of GPN.

We start with a short discussion of business processes. Then we introduce a framework for systems modelling to define requirements for business process models. Based on this we describe the different graph models within GPN and discuss its application to business process planning and scheduling. Finally, we use an example to demonstrate the modelling capabilities of the approach.

2. WHAT IS A BUSINESS PROCESS?

A business process is a stepwise procedure for transforming some given input into some desired output. The transformation is time and resource consuming. A business process has some form of outcome, i.e. goods or services produced for one or more customers either outside or inside the enterprise. There are two usual meanings attached to the term 'business process'; a business process may mean a process *type* or a process *instance*.

The process type can be described by defining general rules and structure of a process; the process instance is a real process following the rules and structure of a given process type. A process type can be interpreted as a pattern; the behaviour of a corresponding instance matches with the pattern. A process type might be a pattern called 'Product Development', and the corresponding instance would be 'Development of Product X' carried out according to the pattern of 'Product Development'. In the sequel a process instance will also be referred to as a workflow or a job.

The process type is defined by its input and output, functions to be performed, and rules of synchronisation. The process *input* and *output* are related to tangible and intangible achievements. For example the major shop floor functions in production have as input different kinds of raw materials which are transformed into various types of output called processed material; office functions are mainly transforming data or information into new data or new information. In general input and output will consist of both material and information.

A *function* represents the transformation of some input into some output. Functions are related through *precedence relations* which constrain the possible ways a process can be executed. E.g. a precedence relation requires synchronisation if the output of a predecessor function is part of the input of the successor function. Before a function can be executed certain *pre-conditions* have to be fulfilled and after a function has been executed certain *post-conditions* should be fulfilled.

Starting and ending a function is caused by *events*. In general an event represents a point in time when certain *conditions* come about, i.e. the conditions hold from that time on until the next event occurs. Conditions related to events are described by values of attributes characterising the situation related to the occurrence of an event.

These event values are compared to pre-conditions and post-conditions of functions. Before carrying out some function its pre-conditions must match with the conditions related to its beginning event and after carrying out a function the conditions related to its ending event must match with the post-conditions of the function. *Synchronisation* means that there must be some order in which functions might be carried out over time; in its simplest form a predecessor-successor relationship has to be defined.

To fully determine a process type a number of variables related to the input and output of functions need to be fixed. The input variables define the *producer* who is responsible for the execution of a function, the required *resources*, and the *required data*; the output variables define the *product* generated by a function, the *customer* of the product, and the *data available* after a function is carried out.

Once a process type is defined its instances can be created. A process instance is performed according to the definition of the corresponding process type. The input, output, functions, and synchronisation of a process instance relate to some workflow or real job which has to be carried out in accordance with the regulations documented by the process type definition. The input must be available, the output must be required. Functions that make up a process type have to be instantiated. A function instance is called task. It is created at a point in time as a result of some event and is executed during a finite time interval.

To ensure task execution various scheduling decisions need to be taken considering the synchronisation and the resource allocation constraints as defined by the process type and resource availability. Scheduling process instances or workflows means to allocate all instances of different process types to the required resources over time. The process type represents constraints for the scheduling decision (Blazewicz *et al*, 1996). In terms of modern scheduling theory an instance of a business process is a job which consists of a set of precedence constrained tasks.

Additional attributes to tasks and jobs can be assigned (Schmidt, 1996b). Questions to be answered for process scheduling are: which task of which job should be executed by which resource and at what time? Typically, performance measures for business process instances are time-based and relate to flow time, tardiness or completion time of jobs; scheduling constraints are related to due dates or deadlines.

3. WHAT HAS TO BE MODELLED?

Modelling is a major component in planning and scheduling of business processes. A framework for systems modelling is given by an architecture. An architecture is based on the requirements for building models and defines the necessary views on a system. Many proposals of architectures have been developed and evaluated with the objective to find a generic enterprise reference architecture (Bernus *et al*, 2003).

An architecture which fits in such a framework is LISA (Schmidt, 1999). LISA differs between four views on models:

- the granularity of the model,

- the elements of the model and their relationships,
- the life cycle phase of modelling, and
- the purpose of the model.

According to granularity models for process types (planning) and for process instances (scheduling) have to be considered. Concerning the elements and their relationships models of business processes should represent all relevant inputs (data, resources) and outputs (data, products), the organisational environment (producer, customer), the functions to be carried out, and the synchronisation (events, conditions, dependencies). Referring to life cycle phases of systems different models are needed for analysis, design, and implementation. Finally, concerning the purpose of modelling we need models for the problem description and for the problem solution. The problem description states the objectives and constraints and the problem solution is a proposal how to meet them. Fig. 1 shows the different views to be represented by business process models in the framework of LISA.

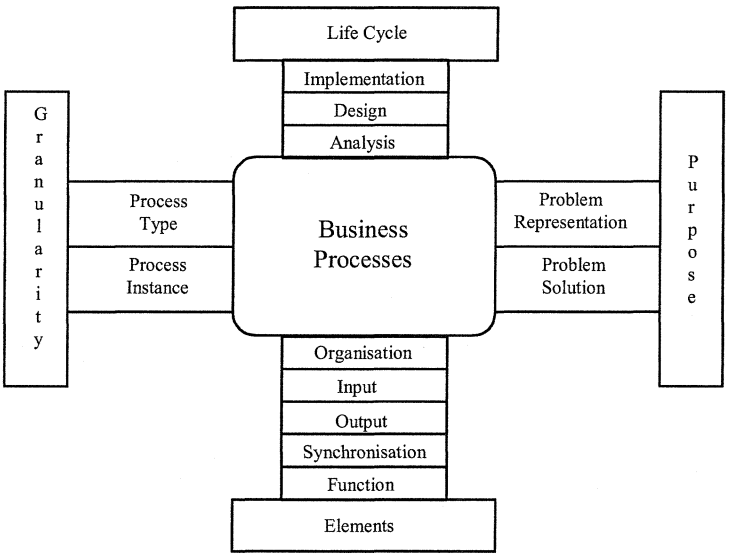

Figure 1. Views on business processes defined by LISA

We will concentrate here on the question how a model for the problem solution can be derived from a model for problem representation. The focus is scheduling of workflows. The modelling language is Generalized Process Networks.

4. GENERALISED PROCESS NETWORKS

There exist many modelling languages to describe business processes. Examples are Petri Nets (Petri, 1962), Data Flow Diagrams (DeMarco, 1978), Event Driven Process Chains (Keller *et al*, 1992), Workflow Nets (Aalst, 1998), Unified Modelling Language (Fowler and Scott, 1998), Dependency Graphs (Kumar and Zhao, 1999) and Metagraphs (Basu and Blanning, 2003). Most of these languages have been developed for planning purposes with a focus on the problem description. Models suited for scheduling purposes in particular for optimisation require a

representation which is suited for combinatorial problem solving (Curtis *et al*, 1992). For this reason Generalized Process Networks (GPN) are developed.

The modelling language has to fulfil the following requirements:

- **Completeness and consistency:** all relevant views of a system must be covered and must be defined in a semantically consistent way,
- **Understandability:** the syntax and semantics must be easy to understand and easy to use by the target audience.

The relevant system views for business processes are modelled as defined in LISA. We shall differ between a model for a process type (used for planning) and a model for a process instance (used for scheduling). However, both models are build with the same language.

GPN = (E, F, A, O, I, L) is a directed And/Or graph with two sets of nodes $E=\{e_1, e_2, \ldots, e_n\}$ (events) and $F=\{f_1, f_2, \ldots, f_m\}$ (functions), a set of arcs $A \subseteq \{E \times F\} \cup \{F \times E\}$, sets of logical (AND, OR, XOR) output $O \subseteq \{E \times F\}$ and input $I \subseteq \{F \times E\}$ operators, and various sets of labels L assigned to events, functions, and arcs.

In a GPN graph events are represented by circles and functions are represented by boxes. Arcs are connecting events and functions or functions and events but never events and events or functions and functions. Events represent the dependencies in processing functions. We differ between six possible dependencies: three for beginning events and three for ending events.

- **begin-AND**: all functions triggered by this event have to be processed (default),
- **begin-OR**: at least one function triggered by this event has to be processed (arcs are connected by an ellipse),
- **begin-XOR**: one and only one function triggered by this event has to be processed (arcs are connected by an ellipse with a dot),
- **end-AND**: this event occurs only if all functions ending with this event have been processed (default),
- **end-OR**: this event occurs if at least one function ending with this event has been processed (arcs are connected by an ellipse),
- **end-XOR**: this event occurs if one and only one function ending with this event has been processed (arcs are connected by an ellipse with a dot).

In Fig. 2 a GPN graph is shown where seven events and seven functions are represented.

There are one **begin-XOR** related to event 1 and functions 12 and 14, one **end-XOR** related to functions 36 and 46 and event 6, and one **end-OR** related to functions 57 and 67 and event 7; all other logical input and output operators are of type **AND**.

The sets of labels L are related to six layers. The first layer defines the labels of the nodes and arcs (numbers), the second layer is dedicated to the functional specifications (name, time, cost, performance), the third to synchronisation aspects representing relationships between functions and events (value lists, pre- and post-conditions), the fourth to input and output data, the fifth to required resources and generated products, and the sixth layer describes the customer-producer relationship of a function. Labels related to the six layers are shown in Fig. 3.

Each function is described by the attributes function name, time and cost consumption which are related its execution, and certain performance measures

under which the execution of the function is evaluated. Connected to each function is a list of pre-conditions and a list of post-conditions. The pre-conditions must be satisfied before the function can be carried out; post-conditions are satisfied as a result after performing the function. Additional labels may be assigned to the function:

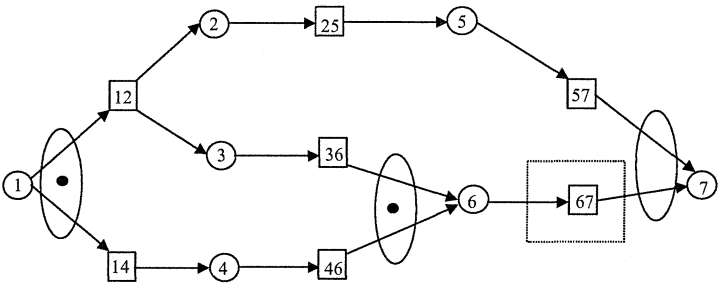

Figure 2. GPN example graph

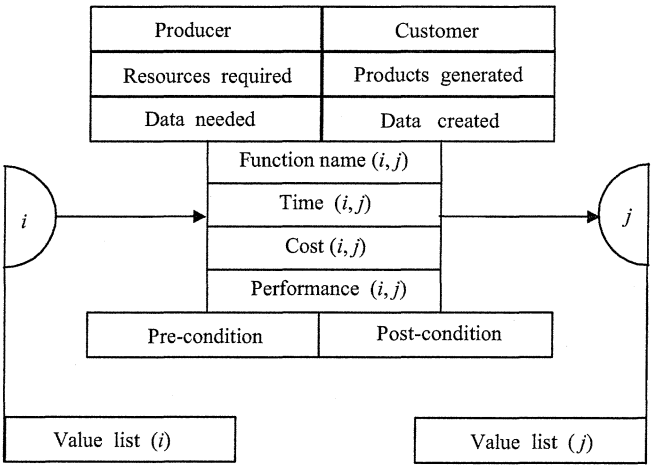

Figure 3. Labels of GPN

- **Producer-Customer:** the producer is responsible for carrying out some function and the customer needs the results from this function. The inputs of the function are transformed under the responsibility of the producers, and the output of the function is consumed by the customers. A producer and a customer might be two distinct organisational units of the same enterprise.
- **Resource-Product:** resources are the physical inputs of the function, products are its physical outputs (resources required, products generated). Resources might be specific machines or employees with certain qualifications as well as material or incoming products to be processed. Products might be types of goods or services.
- **Data-Data:** input data represent the information required for performing a function and output data represent the information available after performing it (data needed, data created).

There are at least two events connected to each function; one represents its start and the other one its end. Events define constraints for the synchronisation of functions.

An event separating two functions represents the constraint that both functions can be executed only in a certain sequence. Functions which have no separating event can be performed in parallel. The occurrence of an event is a necessary condition to perform a function. Each event is described by a value list defining the environmental conditions represented by the event. The occurrence of an event is also a sufficient condition for performing a function if its value list meets the pre-conditions of the function adjacent to this event. An example of a GPN labeling schema is shown in Fig. 4.

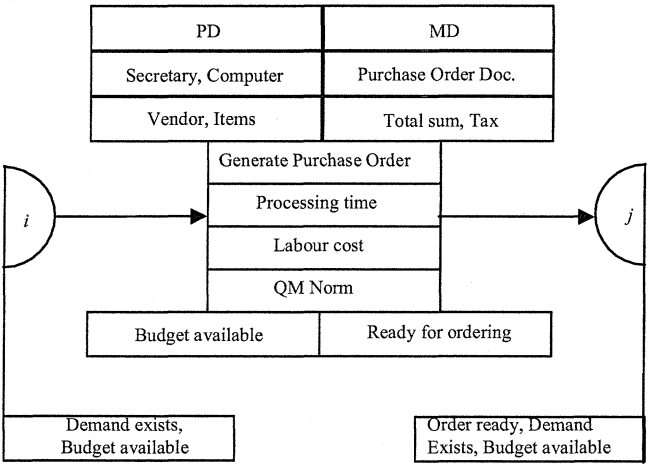

Figure 4. An example for GPN labeling

The function 'Generate Purchase Order' can be interpreted as an activity of a procurement process. Pre-conditions represent the assumption that there must be some 'Budget Available' for purchasing. The post-condition 'Ready for Ordering' which should be fulfilled after the function 'Generate Purchase Order' is processed. The meaning is that the purchase order is ready for sending out. Both conditions match with some values of the list of the beginning and the ending events. Data needed for preparing a purchase order are the 'Vendor' (address of vendor) and the 'Items' (list of items) to be purchased; data created are all purchase order related: 'Total sum' or 'Tax' (to be paid). Required resources might be a 'Secretary' and a 'Computer'; the product generated is a 'Purchase Order Document'. The manufacturing department 'MD' (the customer) asks the purchasing department 'PD' (the producer) to process the function 'Generate Purchase Order'. It can only be carried out if event i occured and not only demand for the items exist but also the budget is still available.

There are certain properties of a GPN related to a workflow:

- A workflow is *well structured, iff* all activities are represented by functions and events and there is only a single workflow instance for each situation.
- Redundancy measures the number of alternatives a workflow can follow.
- Each GPN representing a workflow has a critical path depending on the time consumption of the functions.

4.1 Resource Graph

In the process planning phase all required attributes of a business process are defined; their values are determined once an instance of a business process, i.e. a workflow is created. For example data for 'Vendor' or 'Items' might be 'Vendor ABC' and 'Item 123'. The emphasis of models for workflows is to find answers to scheduling questions, such as timing and resource allocation, taking into account competing process instances or workflows (jobs).

In order to model the resource setting we derive from the GPN graph a resource graph RG = (GPN, R). Doing this we take the defined GPN graph representing the business process and add a set of resource markings $R = \{r_1, r_2, \ldots r_k\}$ where each r_i represents a scarce discrete resource. A resource marking r_i is assigned to these functions f_j which require resource i if they are carried out. Parallel execution of these functions might induce a resource conflict. An example of a resource graph is shown in Fig. 5.

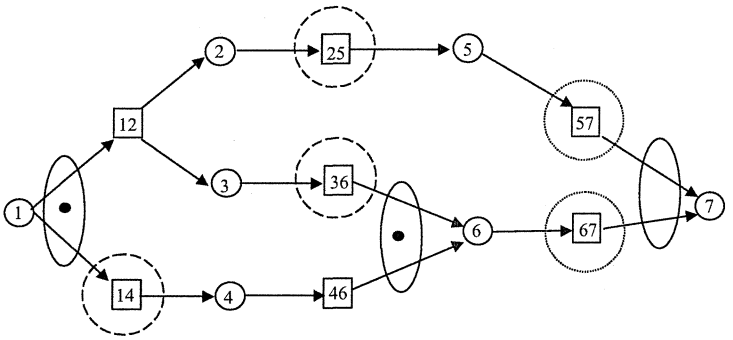

Figure 5. Resource graph example

There are two scarce resources which lead to markings of functions 14, 25 and 36 by r_1 and of functions 57 and 67 by r_2.

There are certain properties of a RG related to a workflow:

- A workflow is *resource restricted, iff* there is at least one resource marking in RG.
- A workflow is *resource constrained, iff* one resource can only be assigned to one function at any time.
- A workflow is *function constrained, iff* one function can only be assigned to one ressource at any time.

4.2 Task Graph

When it comes to operations there are many process instances or workflows (jobs) which have to be carried out in parallel. We will assume that these instances come from workflows which are not only resource restricted but also resource and function constrained.

To model these workflows we define a task graph $TG = (T, A_w, A_r, L)$. $T = \{T_{ij} \mid i=1,..., n; j=1, ..., m\}$ is a set of tasks T_{ij} where index i relates to the function f_i of the workflow and index j relates to different workflows (jobs). $A_w \subseteq \{T_{kj}\} \times \{T_{lj}\}$ is a set of precedence arcs derived from the GPN (all chains of length two connecting predecessor-successor pairs). $A_r \subseteq \{T_{ik}\} \times \{T_{il}\}$ is a set of edges

between tasks T_{ij} which cannot be executed in parallel due to resource constraints. L is a set of labels assigned to tasks and arcs.

In case two or more tasks cannot be processed simultaneously, a hyperedge is introduced between the corresponding tasks. Tasks associated with the same hyperedge create conflicts concerning the usage of resources. The scheduling decision has to resolve these conflicts such that a resource-feasible schedule can be generated (compare(Schmidt, 1989) and (Ecker *et al*, 1997)). To solve the problem all conflicting tasks have to be put in some sequence such that a resource-feasible schedule can be constructed. Algorithms to solve this kind of problem are given in (Ecker and Schmidt, 1993).

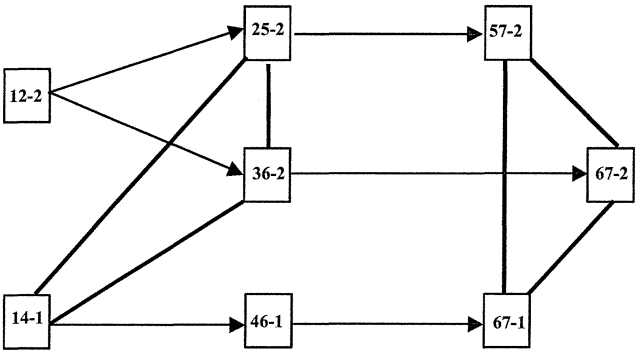

Figure 6. Task graph example

An example of a task graph with two workflow instances related to the GPN graph from Fig. 2 and the resource graph from Fig. 5 is shown in Fig. 6. Tasks in the task graph relate to functions in the GPN graph. The instance number is added to the function number. Resource markings r_1 and r_2 lead to two sets of edges A_1 = {(14-1,25-2), (25-2,36-2), (36-2,14-1)} and A_2 = {(57-2,67-2), (67-2,67-1), (67-1,57-2)}.

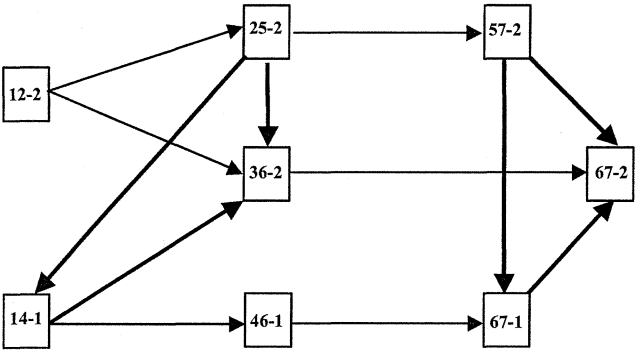

Figure 7. Scheduling solution related to task graph example

There are certain properties of a TG related to a workflow:

- TG is *acyclic*, *iff* each path build from A_w does contain each task node at most once.
- TG is *executable*, *iff* all paths build from A_w do contain each task node at most once.

4.3 Mathematical Programming

Once an acyclic task graph TG is set up the scheduling problem can be treated by means of mathematical programming. Although there is more than one alternative to formulate such a model for a scheduling problem we follow here the formulation given in (Adams *et al*, 1988).

Let

- the tasks from TG be numbered by $T_1, ..., T_{n+m}$
- p_i be the processing time of task T_i
- T_0 and T_{n+m+1} be two dummy tasks *start* and *end* with zero processing time
- P be the set of k resources
- A_w be the set of workflow precedence arcs related to each task pair (T_i, T_j)
- E_i be the set of all pairs of tasks that are resource constrained
- t_i be the earliest possible starting time of task T_i

minimize t_{n+m+1} (1)

subject to

$t_j - t_i \geq p_i$ for all (T_i,T_j) from A_w (2)

$t_j - t_i \geq p_i$ or $t_i - t_j \geq p_i$ for all (T_i,T_j) from E_i for all r_i from P (3)

$t_i \geq 0$ for all T_i from TG (4)

(2) ensures that the workflow order of tasks is obeyed; (3) ensures that there is only one task occupying each resource at a time; (4) ensures that each workflow instance is carried out.

The objective of the mathematical program (1)-(4) is related to the completion time of the last task of all workflow instances, i.e. the makespan. But also other time-based performance measures of workflow execution can be modelled. E.g., in case flow time is the objective we have to minimize $(t_{n+j} + p_{n+j} - t_0)$ with t_{n+j} as the earliest possible starting time of the last task n of workflow j and p_{n+j} as its processing time; if tardiness is a criterium we have to minimize $MAX\{0, t_{n+j} + p_{n+j} - d_j\}$ with d_j as the due date of the j-th workflow.

In Fig. 7 a feasible solution is given showing an executable TG.

5. EXAMPLE PROBLEM

We shall now demonstrate how GPN can be used for modelling a business process for planning and for scheduling purposes. The example is related to a procurement process. It deals with purchasing goods and paying corresponding bills. Let us start to explain how to build a model on the planning level considering the following setting.

If the manufacturing department (MD) of a company is running out of safety stock for some material it is asking the purchasing department (PD) to order an appropriate amount of items. PD fills in a purchase order and transmits it by E-mail or fax to the vendor; a copy of the confirmed purchase order is passed to the accounts payable department (APD). The vendor is sending the goods together with the receiving document to the ordering company; with separate mail the invoice is also sent.

Once the invoice arrives PD compares it with the purchase order and the goods sent via the receiving document. The documents are checked for completeness and

for correctness. If the delivery is approved APD will pay the bill; if not PD complains to the vendor. Invoices for purchased goods come in regularly and have to be processed appropriately.

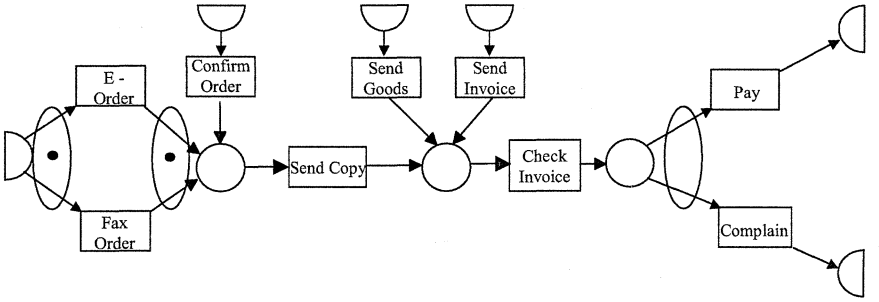

Figure 8. Procurement process

This process is shown in Fig. 8. Arcs leading from left to right represent the functions of the purchaser's process and arcs leading from the top to the bottom represent functions of the vendor's process. The purchasing order can be sent either by e-mail or by fax. This is represented by the two functions 'E-Order' and 'Fax Order'. Once the order is confirmed by the vendor a copy of the order is sent to APD represented by the function 'Send Copy'. If the ordered goods and the corresponding invoice have arrived the function 'Check Invoice' can be carried out. Depending on the outcome of the checking procedure the functions 'Pay' or 'Complain' are performed. In case there are complaints only about parts of the delivery both functions are carried out.

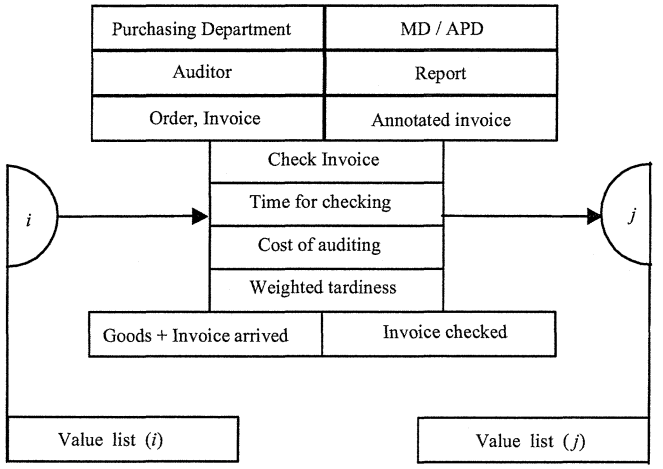

Figure 9. GPN representation of a selected function

In Fig. 8 the labels for most of the layers were omitted. In order to give a small example how labelling is done we concentrate on the function 'Check Invoice' using all six GPN layers. The result is shown in Fig. 9. We assume that PD is taking over the responsibility for this function and MD and APD need the results. The resource needed is an auditor who is generating a report. Data needed for the 'Check Invoice'

function are the order and the invoice data; the function creates 'Annotated Invoice' data. Before the function can be carried out the ordered goods and the invoice should have arrived; after carrying out the function the condition holds that the invoice has been checked. The remaining parts of the process have to be labelled in an analogous way.

We now investigate the workflows resulting from the procurement process with focus on the scheduling decisions to be taken. We want to assume that with each individual invoice discount chances and penalty risks arise. A discount applies if the invoice is paid early enough and a penalty is due if the payment is overdue.

Let us focus again on the function 'Check Invoice'. The corresponding tasks require some processing time related to the work required for checking a current invoice. Moreover, for each instance two dates are important. One relates to the time when the invoice has to be paid in order to receive some discount, the other relates to the time after which some additional penalty has to be paid. For the ease of the discussion we assume that discount and penalty rates are the same. Let us furthermore assume that there is only one auditor available to perform these tasks and that the auditor is the only constrained resource of the procurement process. The resource graph is shown in Fig. 10.

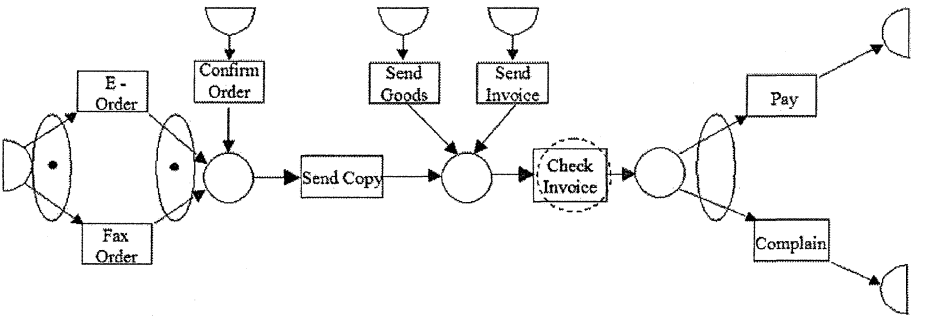

Figure 10. Resource graph of the procurement process

There are three invoices waiting to be processed. It is obvious that the sequence of processing is of major influence on the time of payment considering discount and penalty possibilities. Table I summarises the scheduling parameters showing invoice number (J_j), total sum of the invoice (w_j), time required to check an invoice (p_j), discount date (dd_j), penalty date (pd_j), and the rate for discount and penalty (r_j), respectively.

Table I Scheduling parameters of the procurement workflow

J_j	w_j	p_j	dd_j	pd_j	r_j
J_1	200	5	10	20	0.05
J_2	400	6	10	20	0.05
J_3	400	5	10	15	0.05

In general there are n invoices with n! possibilities to process them using a single resource. The range of the results for the example data is from net savings of 30

units of cash discount up to paying additional 10 units of penalty depending on the sequence of processing. The task graph is shown in Fig. 11.

The data required for scheduling relate to the processing times p_j, the amount of the invoice w_j, the discount and penalty rates r_j, the discount dates dd_j, and penalty dates pd_j; the scheduling objective is assumed to be to maximise the sum of cash discount minus the penalty to be paid. The mathematical program (1)-(4) has to be revised in an analogous way. An optimal schedule is represented in Fig. 12.

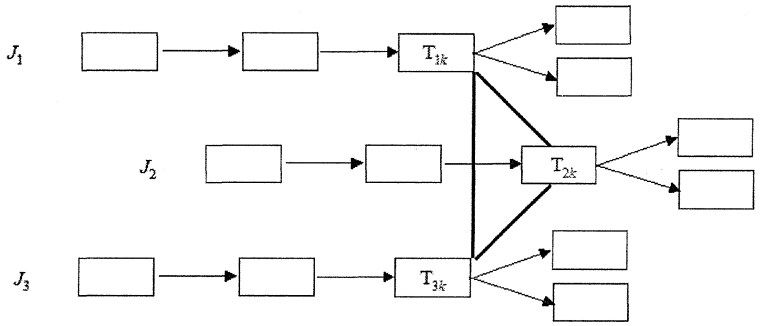

Figure 11. Task graph of the procurement process

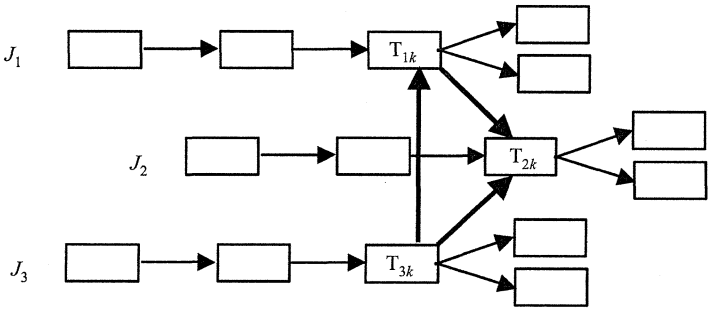

Figure 12. Scheduling solution of the procurement workflow

6. CONCLUSIONS

We have presented business graphs to describe business processes and to schedule corresponding workflows within a single model. Besides GPN resource graphs and task graphs are used as a formalism to create appropriate data structures for the application of mathematical programming formulations. The approach has the capabilities to structure problems from a descriptive point of view and to optimise workflows based on time-based criteria. It is easy to understand and easy to use, and it is especially suited for modelling sequence dependent decision problems having a combinatorial structure.

We have not presented algorithms to solve the arising scheduling problems. The scope of this contribution is to demonstrate that planning and scheduling problems can be modelled using a common and easy to use notational framework. We have illustrated this by an example. There are many business processes which can be analysed and optimised using the notational framework of business graphs.

There are promising areas for future research. One is to develop business graph-based workflow optimisation tools. Another one is a theoretical analysis of various workflows with respect to models investigated in modern scheduling theory. A third area might be the integration of business graphs in commercially available business process modelling tools.

REFERENCES

Adams, J., Balas, E., Zawack, D. (1988) The shifting bottleneck procedure for job shop scheduling, *Management Science* 34, 391-401

Aalst, W.M.P. (1998) The application of Petri nets to workflow management, *J. Circuits Systems Comput.* 8(1), 21-66

Basu, A., Blanning, R. (2003) Synthesis and decomposition of processes in organizations, *Information Systems Research* 14(4), 337-355

Blazewicz, J., Ecker, K., Pesch, E., Schmidt, G., Weglarz, J. (1996) *Scheduling Computer and Manufacturing Processes*, Springer, Berlin

Bernus, P., Nemes, L., Schmidt, G. (eds.) (2003) *Handbook on Enterprise Architecture*, Springer, Berlin

Curtis, B., Kellner, M. I., Over, J. (1992) Process modeling, *Communications of the ACM* 35(9), 75-90

DeMarco, T. (1978) Structured Analysis and System Specification, New York

Ecker, K., Gupta, J., Schmidt, G. (1997) A framework for decision support systems for scheduling problems, *Eur. J. of Operational Research*, 101, 452-462

Ecker, K., Schmidt, G. (1993) Conflict resolution algorithms for scheduling problems, in: K. Ecker, R. Hirschberg (eds.), *Lessach Workshop on Parallel Processing, Report No.* 93/5, TU Clausthal, 81-90

Fowler, M., Scott, K. (1998) UML Distilled: Applying the Standard Object Modelling Language, Addison-Wesley, Reading

Keller, G., Nüttgens, M., Scheer, A.-W. (1992) *Semantische Prozeßmodellierung auf der Grundlage Ereignisgesteuerter Prozeßketten (EPK)*, Veröffentlichungen des Instituts für Wirtschaftsinformatik Nr. 89, Saarbrücken

Kumar, A., Zhao, J. L. (1999) Dynamic routing and operational controls in workflow management systems, *Management Science* 45(2), 253-272

Malone, T. W., Crowston, K. (1994) The interdisciplinary study of coordination, *ACM Computing Surveys* 26(1), 87-119

Petri, C.A. (1962) Kommunikation mit Automaten, *Schriften des Instituts für Instrumentelle Mathematik* Nr. 3, Bonn

Schmidt, G. (1989) Constraint-satisfaction problems in project scheduling, in: Slowinski, R., and Weglarz, J. (1989), 135-150

Schmidt, G. (1996) Scheduling models for workflow management, in: B. Scholz-Reiter, E. Stickel (eds.), *Business Process Modelling*, Springer, 67-80

Schmidt, G. (1996b) Modelling production scheduling systems, *Int. J. Production Economics* 46-47, 109-118

Schmidt, G. (1999) Informationsmanagement - Modelle, Methoden, Techniken, Springer, Berlin

Schmidt, G. (2002) *Prozeßmanagement - Modelle und Methoden*, Springer, Berlin

Slowinski, R., Weglarz, J. (eds.) (1989) *Recent Advances in Project Scheduling*, Elsevier, Amsterdam