

# From Enterprise Models to Scheduling Models: Bridging the Gap

Roman Barták<sup>1</sup>, James Little<sup>2</sup>, Oscar Manzano<sup>2</sup>, Con Sheahan<sup>3</sup>

<sup>1</sup>Charles University in Prague, Faculty of Mathematics and Physics  
Malostranské nám. 2/25, 118 00 Praha 1, Czech Republic  
roman.bartak@mff.cuni.cz

<sup>2</sup>Cork Constraint Computation Centre  
University College Cork, Cork, Ireland  
{j.little,o.manzano}@4c.ucc.ie

<sup>3</sup>University of Limerick, College of Engineering  
Limerick, Ireland  
con.sheahan@ul.ie

**Abstract.** Enterprise models cover all aspects of modern enterprises, from accounting, through management of custom orders and invoicing, to operational data such as records on machines and workers. In other words, all data necessary for running the company are available in enterprise models. The problem is that these data are not in the proper format for some tasks such as scheduling and optimization. This paper deals with the automated translation of data from the enterprise model to a scheduling model and back. In particular, we describe how to extract data from the enterprise model for solving the scheduling problem using constraint-based solvers.

**Keywords:** constraint programming, enterprise modeling, optimization, automatic scheduling.

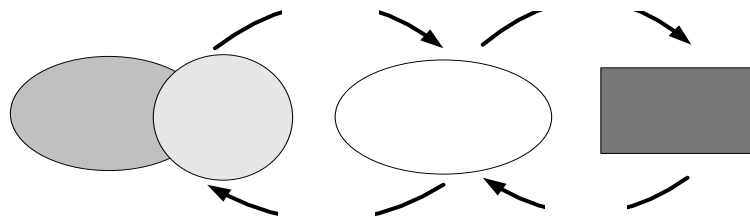
## 1 Introduction

Manufacturing firms have a requirement for Enterprise Modelling and Performance Optimisation tools to enable them to deploy their limited resources for maximum economic yield for their products. Yet, these two areas – modelling and optimisation – are typically addressed separately. In addition, for Small and Medium Enterprises, describing a scheduling model is a severe challenge in terms of their limited budget, capabilities, and time.

We claim that there indeed is enough information within an enterprise model to deduce and automatically generate a scheduling model, which can be run through a scheduler, in order to produce times and resource assignments against the satisfaction of orders. In this way, we are able to combine the two models together and provide easy-to-use, entry level scheduling facilities for SME's. We further claim that the different operating conditions; such as the treatment of lateness for orders and

outsourcing options, can also be automatically incorporated into the scheduling model in the form of constraints and objectives, within an extended enterprise model. The main innovation here is the development of a set of rules to access information from the extended enterprise model and to create the necessary objects for scheduling. These rules also cover the reverse process of converting the scheduling information back into the enterprise model from which the firm's key performance indicators (KPI's) and a schedule of operation can be extracted.

The main contribution of this paper is in the area of knowledge engineering. We introduce the extended enterprise model, which can be seen as a dictionary for translating the notions and concepts between the enterprise models and scheduling techniques, and the translation rules used to obtain the scheduling model from it. We then present an architecture based on Figure 1, to deliver automatically a schedule and implement such a system. The outcome demonstrates the potential for enhancing existing MRP systems to make them more usable to a wider, non-scheduling proficient, audience. This work is being carried out within the EU Framework Programme 6 EMPOSME project which develops an integrated enterprise modelling and scheduling system for SME's in the furniture manufacturing industry.



**Fig. 1.** Integration Architecture between Enterprise Model and Scheduling model.

## 2 The Enterprise Translation Model

In this section we describe and justify the requirements on an enterprise model towards providing sufficient information to generate a scheduling model. Since the extended enterprise model describes the company's operational requirement in non-scheduling terms, we cannot expect to obtain directly such scheduling concepts as activities, resource types, temporal constraints, etc. Instead, we have to focus on related concepts such as work flow, machine availability, outsourcing options and bill of materials. We define the extended enterprise model as a way for describing a scheduling problem in enterprise terms and call it the *translational model*. This model covers a subset of enterprise concepts necessary for scheduling, and also extends traditional enterprise models to provide more scheduling related information, which we expect users to have access to. We use an XML format as an interface language for this translation model. Some examples in this format will be given in the paper, other examples can be found in [2]. This work complements the work done in defining a scheduling model in MaScLib by ILOG [6] and TAEMS formalism [3].

First, it should be said that the translation model expresses a snapshot of the enterprise at some given time. Recall that the main motivation for the translation model is to provide necessary data for scheduling. At the first stage, we expect scheduling in batches, where a schedule for some known period is generated. Hence, we expect to know the schedule start and the schedule end and only data important for this period are included in the translational model.

*Orders* for products are the major input for scheduling; these dictate what needs to be manufactured. Typically, the order has some identification and it consists of the ordered product(s), its quantity, and the delivery date. We assume that some of this data, such as dates, have been normalised and converted already by the calendars underlying the enterprise model. In particular, the delivery date is specified in some abstract time units from the start of the schedule rather than an absolute date. At this order level we can also describe the consequences of delivering the order late or early by specifying penalty for being early or late with the delivery. Again some normalised cost is used for the penalty. Finally, we can say that the order can be outsourced (provided that the product is also outsourceable).

Each *product* can be further described in terms of its composition and the possibility of it being outsourced. The final assembly of a product is made up of three stages, the product assembly based on its bill of materials and any additional processes needed before and after that.

The assembly description is effectively a *bill of materials* from which we can determine how a product is assembled from its component parts. At the same time, the description refers to a manufacturing route which describes a *work flow* for each basic part. An example of the manufacturing route is described below in Figure 2. This structure brings together different route pieces (each made up of pairs of linked processes) into a single work flow towards making a part. In the example, we show how two sequential processes MP1 and MP2 need to be carried out, where their relationship is defined as being, MP1 finishes before MP2 can start after a minimum of 1 time unit and a maximum of 100 time units.

```

<Route>
  <ManufacturingRouteID>MR1</ManufacturingRouteID>
  <RoutePiece>
    <ManufacturingProcessID>MP1</ManufacturingProcessID>
    <SuccessorProcesses>
      <SuccessorProcess>
        <ManufacturingProcessID>MP2</ManufacturingProcessID>
        <LinkID>L1</LinkID>
      </SuccessorProcess>
    </SuccessorProcesses>
  </RoutePiece>
</Route>
<Links>
  <Link>
    <LinkID>L1</LinkID>
    <LinkType>finish-start</LinkType>
    <MinDelay>1</MinDelay>
    <MaxDelay>100</MaxDelay>
  </Link>
</Links>

```

**Fig. 2.** Description of a Product Work Flow.

The full workflow associated with a product is made of processes before and after the actual product final assembly. Processes such as setup, packaging and transportation are important to the final schedule, but are not intrinsic to the manufacture. These before and after routes are described similarly to the manufacturing processes and hence we omit their detail description.

For each process, there are one or more *machines, tools, operators*, etc. which are required to carry it out. Figure 3 shows this relationship between a manufacturing process MP1 and various types of resources required to carry it out. In particular, the process MP1 requires machine M1, worker H2 and one of either machine M2 or M3. This process has a duration id DUR1 which is referred to in another part of the model.

```

<Process>
  <ManufacturingProcessID>MP1</ManufacturingProcessID>
  <DurationLinkId>DUR1</DurationLinkId>
  <ResourcesRequired>
    <RequiredMachineResources>
      <MachineResourceID>M1</MachineResourceID>
    </RequiredMachineResources>
    <AlternativeMachineResources>
      <MachineResourceID>M2</MachineResourceID>
      <MachineResourceID>M3</MachineResourceID>
    </AlternativeMachineResources>
    <RequiredHumanResources>
      <HumanResourceID>H2</HumanResourceID>
    </RequiredHumanResources>
  </ResourcesRequired>
</Process>

```

**Fig. 3.** Description of Machine and Labor Requirements for a Process.

The machine resources are further described in terms of their *availability*. In particular, each resource has assigned a set of intervals describing the periods of unavailability. This information is deduced from calendars assigned to resources and it is used during scheduling to restrict allocation of activities to a particular resource.

*Durations* of processes are, where possible, pre-calculated. Durations of processes are determined by the quantity and performance of the resources they require. Therefore, if the resources are known a fixed quantity can be included. Where alternative resources with different performances are available, then we can enumerate the possibilities. This calculation is tied to an order for a product and associated resource ids involved.

The translational model also holds information on the *results of the scheduling*. This comprises of two parts. The first is an actual schedule which is used to organize the work through the factory. The second type of output is a report on the implications on the orders; for example, whether the order is late or not, is being outsourced or will require storage. For the *schedule*, two complementary views are presented; the task based view and the resource based view. The task based view consists of a set of task, where each task is associated with the process id, produced part of a product, an order to which it belongs, and of course the start time and finish time expressed in the same abstract time units as mentioned above. The resource view shows all the tasks using each particular resource. In particular, for each resource there is a set of tasks that use that particular resource together with the time period when they use the resource.

### 3 System Architecture

This section describes in more detail the important stages in building a scheduling model from the enterprise description. We initially need to identify the scheduling entities (e.g. activities, unary resources, objectives etc.) from the XML description. After that, some basic checks (pre-processing) can be made on the validity of the data for scheduling and then additional scheduling information will be added.

#### 3.1 Translation rules

The Enterprise model is business centric and so the information a scheduler needs is not necessary explicit and in addition it can be dispersed around the model. The purpose of translation rules is to describe the way a scheduler can examine the enterprise data and realize the different scheduling entities and the relationships between them. These rules can then be encoded in the form of an executable parser. We now present a description of the translation rules necessary for identifying the major scheduling entities.

##### Activities

There are different types of activities found throughout the enterprise model. From the process descriptions (described in Figure 3), we can obtain each activity necessary to manufacture a part. Equally, from the bill of materials, we can identify the points at which an assembly of constituent parts needs to take place. There are also milestone activities such as the schedule-end or product release, which also need to be represented. Figure 4 shows the access points for identifying the assembly activities.

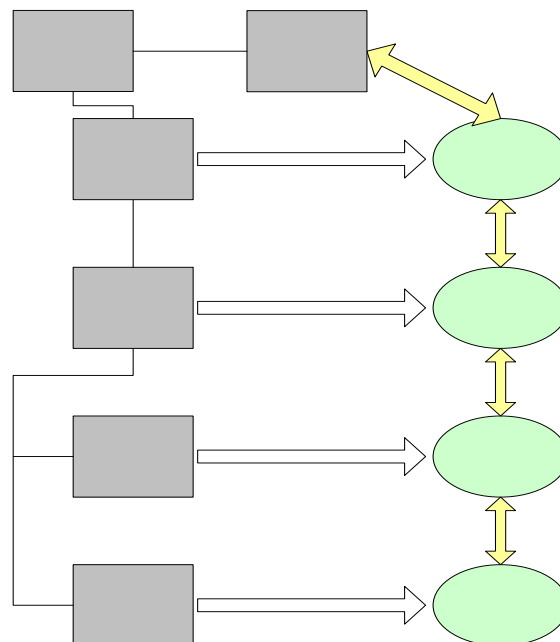


Fig. 4. Temporal Constraints Generated from the Assembly of Product.

### Temporal Relationships

There are many temporal relationships between processes throughout the enterprise model. The work flow description gives us a clear set of relationships between these processes as shown in Figure 5. We can see from Figure 2 how 'Route Pieces' holds the process id's of successive pairs of processes. From this, a temporal relationship is known and from their 'link' information the precise temporal relationship is obtained. From the bill of materials we can also obtain the temporal relationships between the completion of the final process of each constituent part and the start of the assembly activity (Figure 4). Temporal relationships also occur between the schedule-end and each of the final product assemblies.

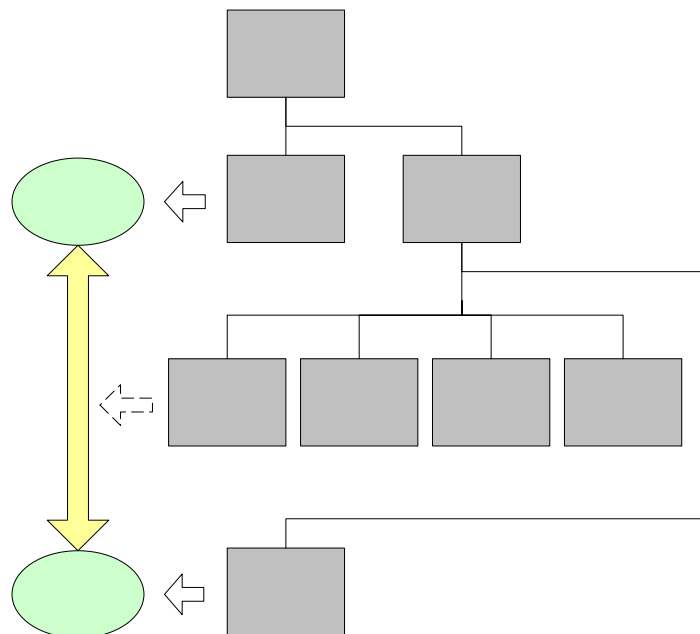


Fig. 5. Temporal Constraints Generated from the Work Flow Description.

### Resources and Resource Constraints

In scheduling, resources are identified as being able to place restrictions on when and where an activity can take place. Different types of resources have different representations and hence behaviors in the scheduling process; these are unary, discrete and alternative resources. In the translational model they are identified in terms of their real-life manifestation. Therefore, we consider machines, labor and raw materials as resources, which can be further incorporated into groups or provided as alternatives for a single process. Each of them corresponds to a particular scheduling resource entity. Figure 6 identifies the different types of resources in the translational model and how they map through to scheduling resources. At the same time, the figure shows how these resources relate to a process activity which requires them, described also in Figure 3.



### Due dates

Each product order has a due date associated with it. An appropriate constraint on the end of the final assembly activity of each product can be imposed in the scheduling model (Figure 4). The nature of this constraint will depend on the acceptability of tardiness or earliness of each product order.

### Outsourcing

Some products may be produced externally for a known cost and duration. This is an option which many SME manufacturers use to be able to satisfy all their orders within the promised time. Within the translational model, each product has a flag indicating whether it can be outsourced or not. We choose to model this as an alternative manufacturing route in the scheduling model (see Figure 8). For a product with an outsourcing option, the translation will create an outsourcing activity alongside the internal manufacturing activities. A decision variable within the scheduling model allows the search to evaluate both possibilities similarly to selection between alternative resources (Figure 7). The cost and duration calculation are also contained in the translational model from which we can put the appropriate constraints on the variable durations of all the activities involved (whether they are selected or not). Currently we assume only additional cost for outsourcing (in comparison to on site production) and hence no cost of processes is included. Nevertheless, the translation model can be easily extended by assuming the cost of processes which is especially useful when alternative processes are present. To model any type of process alternatives, we developed so called Temporal Networks with Alternatives [1] that formally describe alternative routes in processes. Note also, that to exploit the full potential of alternatives one may need a full cost model, where the (weighted) sum of costs is used as the primary objective.

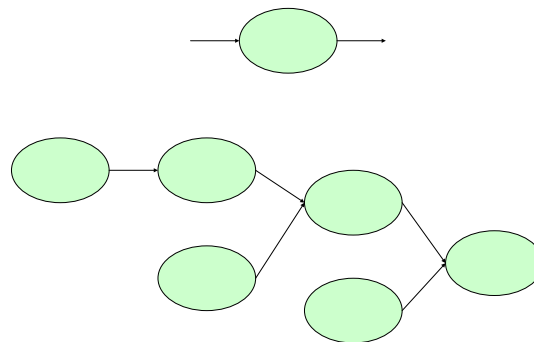


Fig. 8. Alternative Routes for Manufacture using Outsourcing.

### Batching

It is assumed that all batch size calculations have been made for each product order in advance. Therefore, separate orders will be introduced and consequently different activities are created.

### **Schedule Relaxations and Objectives**

On time in full is the common requirement for the schedule by the SME's in the project. We interpret this initially as trying to deliver before all promise dates and to complete the work as quickly as possible, within the given scheduling time window. This in turn is reflected in the model having hard constraints on products being available before the due dates and the objective function being to minimize the end time of the final activity (makespan).

However, there are other aspects permissible in the schedule and different overall objectives that we need to consider. For example, on time in full may be the goal, but if the due dates cannot be met, then a model with hard due date constraints will return no schedule. If we remove the due date restrictions, then assuming availability of resources, we would produce a schedule, but orders would likely be overdue. Therefore, if lateness were allowed, then we should treat it using a goal programming approach, of penalizing lateness in the objective. Equally, earliness from the schedule may also impose a cost and so should be discouraged.

We therefore allow the user to indicate whether to accept orders being late, early or both, and at the same time, an associated objective has to be chosen. Together they indicate the type of model and the set of appropriate constraints we require to reflect this.

In describing the quality of a schedule the user is looking for, there are many factors which can be included. The possible criteria are,

1. Minimise lateness
2. Minimise earliness
3. Minimise lateness and earliness (Just in Time)
4. Minimise outsourcing cost
5. Minimise makespan

Any combination can be considered if appropriate weights are given to each. However, we provide in the current system only single criteria.

Within the translational model the Performance Indication value identifies the objective criteria which the schedule tries to improve on and the Treatment Of Lateness value indicates whether the orders can be late or not. Only valid combinations of objective and handling of lateness are accepted by the system. For example, outsourcing can be selected for consideration and made available, but will only be considered when the objective is to minimize makespan.

### **Scheduling Preferences**

How a schedule looks and feels is quite subjective to the user. Indeed the EMPO system is designed only to give a starting-point schedule for further manual enhancements. However, we can incorporate certain preferences or features into the schedule, if these are required by the user. The preference of alternative machines is an example which can be incorporated into both models. Here, the user prioritizes one machine over another for reasons of utilization. In the XML description, an implicit priority is assumed through the declared order of the alternative machines. Hence in Figure 3, machine M2 is prioritized before machine M3. We reflect this in the scheduling model by including an aspect of the search strategy which considers the preferred resource first, and only if improvements can be made to the objective of the schedule, will a reverse assignment be considered.



### 3.2 Pre-processing

Once the scheduling information has been extracted and assembled, simple “schedulability” checks can be made before proceeding. This is called ‘pre-processing’. Apart from type and range errors in the data, there are some scheduling logic tests which can be applied. This allows us to deduce early whether or not the orders can be possibly scheduled or not. Indeed, current scheduling systems do not provide explanations as to why a schedule cannot be produced and this is not helpful to users. The current experimental explanation techniques can for example explain the failure of a particular scheduling constraint [8] but this may be too hard to interpret by users that are not proficient in scheduling. On the other hand, in many cases the lack of a schedule may be for a very simple reason and the pre-processing module both prevents unnecessary time being spent going ahead in scheduling and gives early clear feedback on why a schedule cannot be made. The pre-processing we have implemented is not exhaustive; there are still cases where a schedule is found to be infeasible only after trying to schedule, but it demonstrates the idea of recognising common scheduling ‘mistakes’. There are currently three types of pre-processing activities carried out within the EMPO system.

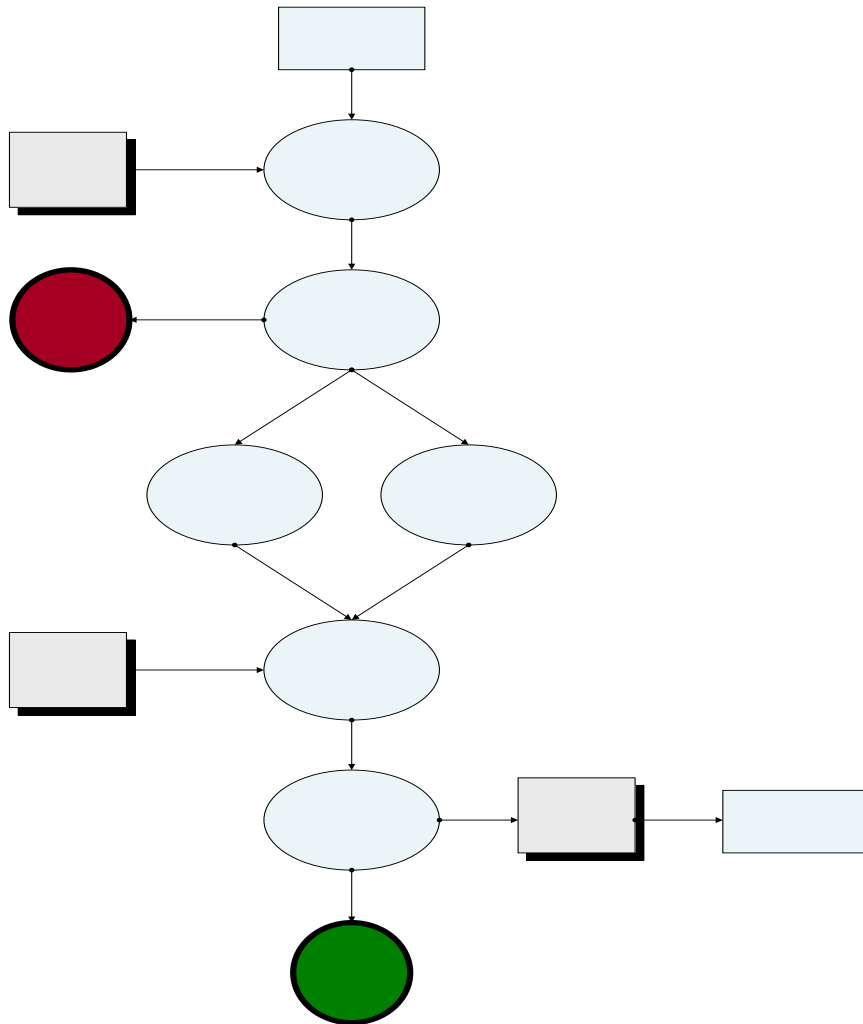
**Check for infeasible scheduling data** – this covers type and range discrepancies in the data which would not be handled correctly inside the scheduler. Since the translational model is described as an XML file, we are able to generate and use an XML Schema to verify the input structure and data types as well.

**Check for incorrect scheduling data** – this is data, which although correct in isolation, is not consistent with other data in the enterprise description from a scheduling point of view. For example, we have implemented tests on the minimum possible delivery date for a product and compared this against the requested release date. If we do not allow lateness, and the sum of durations is greater than the release date, then no schedule can be obtained. Another test is in comparing the initial availability of raw material with what will be required during the scheduling period. Assuming no further delivery of raw material during the scheduling period, then if the requirement is greater than what is available no schedule can be produced. Notice also that in each of these cases an exact reason can be given for the failure of the schedule. It is then up to the user to correct this.

**Reduce problem complexity** – the complexity of the resulting scheduling problem is often a consequence of the number of activities generated. There are opportunities in which activities can be aggregated together at the product level rather than at the part level in order to provide some scheduling feedback. It is possible to deduce this information right before the scheduling stage for example by identify similar substructures in process networks [5], but this information is easier to access at the translation stage where original semantics (such as names of products) is still present.

## 4 Implementation

The EMPO system is based on enterprise scheduling information represented in an XML format. This data is accessed by the translator which generates an instance of a scheduling model which in turn is run through a scheduling engine. The translator is written in C#, while the scheduling engine is one of either Ilog OPL 3.7.1 [4] or Sicstus Prolog 4.0.1 [7]. All are contained with a .NET environment. Each engine has its own characteristics in terms of performance and modeling capabilities, and for this project both were evaluated. The translator is able to use the same rules to provide the same scheduling information to both engines; albeit in different formats. In Figure 10, we illustrate as a data flow, how the various modules of the EMPO system interact.



**Fig. 10.** Data Flow Diagram for the EMPO translator system.

## 5 Conclusions

We have shown how to extend and use an existing enterprise model, say from an MRP system, provide sufficient company information for a scheduling model to be realized and run. The impact of this is to give users with no scheduling experience, the benefits of state of the art scheduling systems, without having to learn much about scheduling or the underlying technologies. Note that almost all data in the translational model are deduced automatically from the enterprise model. The user only needs to specify the scheduling period and to select particular objectives. Actually, the users may experiment with different objectives to produce various schedules that can be later compared using other performance indicators. This makes scheduling much easier for users with no scheduling experience which is typical for SMEs. Features such as pre-processing and scheduling intelligence also help the user experience. The EMPO system will enable those firms in the project to compete more effectively, since it will enable them to manage the increasing complexity of their enterprises without increasing their cost base.

## References

1. Barták, R.; Čepek, O.; Surynek, P.: Modelling Alternatives in Temporal Networks. In Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Scheduling (CI-Sched 2007), IEEE Press (2007), 129–136
2. EMPOSME Translation Model, 4C, <http://4c.ucc.ie/web/projects/empo/index.html>
3. Horling B.; Leader V.; Vincent R.; Wagner T.; Raja A.; Zhang S., Decker K., and Harvey A.: The Taems White Paper, University of Massachusetts, (1999) <http://mas.cs.umass.edu/research/taems/white/taemswhite.pdf>
4. ILOG OPL Studio, ILOG, <http://www.ilog.com/products/oplstudio/>
5. Kovács, A.; Váncza, J.: Progressive Solutions: A Simple but Efficient Dominance Rule for Practical RCPSP. In Proc. of CPAIOR 2006, the 3<sup>rd</sup> Int. Conf. on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, LNCS 3990, Springer Verlag (2006), 139–151
6. Nuijten, W.; Bousonville, T.; Focacci, F.; Godard, D.; Le Pape, C.: MaScLib: Problem description and test bed design, (2003) <http://www2.ilog.com/masclib>
7. SICStus Prolog 4.0.1, SICS, <http://www.sics.se/sicstus>
8. Vilím, P.: Computing Explanations for Global Scheduling Constraints. In Principles and Practice of Constraint Programming – CP2003, LNCS 2833, Springer Verlag (2003), 1000