Métodos Formais de Programação II

4.º Ano da LMCC (7008N2)
Academic Year: 1999/2000

Project Proposal — May 10, 2000

Introduction

The main theme of this project it the specification and (partial) calculation of a software package which monitors the activity of a particular kind of organization. Part of this specification is provided as a starting point. Additional information is provided in the form of WWW links to lecture notes or other projects which, if carefully studied, will make the project's overall task considerably simpler.

This project consists mainly of two parts. First, a specification model is provided as a starting point but with no functionality; the task consists in providing such a functionality. Second, part of the overall specification should be subject to calculations leading to a relational implementation.

The structure of this document is as follows: section **Starting Specification Model** describes the problem domain while drafting a formal specification of a collection of basic datatypes which is put forward as a starting point. Section **More Information Available** provides links to additional and useful material available from the WWW. Finally, section **Project Proposal** contains the actual requirements of the project.

Starting Specification Model

The kind of organization we have in mind is structured in departments. Departments may be organized hierarchically in sub-departments and so son. So our model will start by the following datatype

$$OrgSt \cong DepId \rightharpoonup Dept \qquad \text{/* } OrgSt = \text{organization structure */}$$

where $DepId$ (= department identifier) should be regarded atomic (*e.g.* alphanumeric) and

$$Dept \cong DepInf + OrgSt \qquad \text{/* } Dept = \text{department */}$$

where $DepInf$ will record each department (and sub-department) information, most of the details of which (*e.g.* director, employees, address *etc.*) are ignored for the sake of simplicity. Instead, we will focus on the *tasks* which a particular (sub)department accomplishes within the organization and $DepInf$ will record, for each task in which the (sub)department collaborates, the amount of resources (*e.g.* manpower, equipment, materials *etc.*) needed per unit of time (*e.g.* week):

$$\begin{array}{rcll} DepInf & \cong & TaskId \rightharpoonup Reso & \text{/* } TaskId = \text{task identifier */} \\ Reso & \cong & ResId \rightharpoonup \mathbb{N} & \text{/* } ResId = \text{resource identifier */} \end{array}$$

Next, one needs to consider how tasks are performed by the organization. Each task will have a unique identifier ($TaskId$) and some associated information ($TaskInf$). This includes a textual description ($Desc$), how long it takes ($Span$) and which other tasks ($Preds$) need to be completed before it can be triggered:

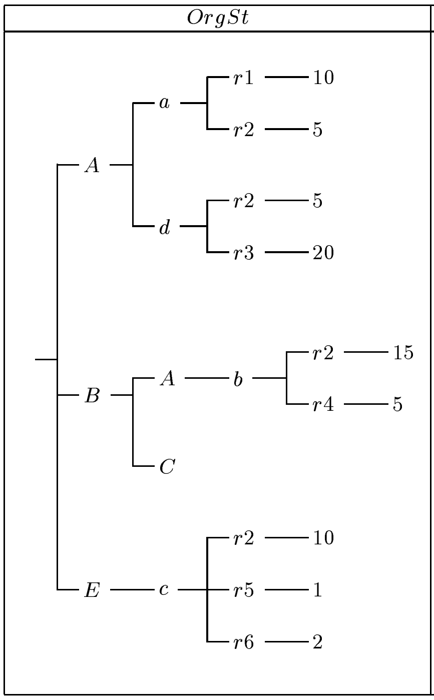
$$\begin{array}{rcll} Process & \cong & TaskId \rightharpoonup TaskInf & \text{/* } Process = \text{tasks' network */} \\ TaskInf & \cong & Desc \times Span \times Preds & \text{/* } TaskInf = \text{task information */} \\ Desc & \cong & Text & \text{/* } Desc = \text{task description */} \\ Span & \cong & \mathbb{N} & \text{/* } Span = \text{task duration */} \\ Preds & \cong & 2^{TaskId} & \text{/* } Preds = \text{predecessor tasks */} \end{array}$$

The model of datatype $Text$ will be left unspecified since no functionality will interfere with it (any suitable model of textual data will do).

Putting everything together, we obtain a model of what we understand about an organization on the whole:

$$Org \cong OrgSt \times Process \qquad \text{/* } Org = \text{overall organization */}$$

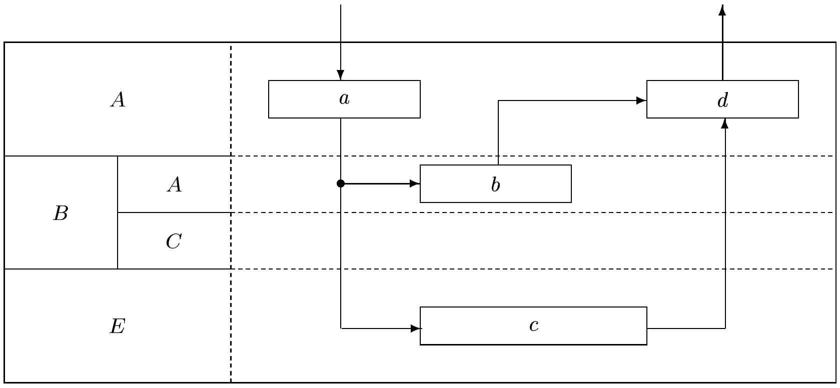
For instance, the following table depicts a particular ("toy") organization. In the $OrgSt$ part, capital letters represent $DepId$ values, lower case letters represent $TaskId$ values and other non-numeric symbols represent $ResId$ values. In the $Process$ part, the data-field $Desc$ is not represented:



$Process$

TaskId	Desc	Span	Preds
a	$\cdots$	2	$\emptyset$
b	$\cdots$	2	$\{a\}$
c	$\cdots$	3	$\{a\}$
d	$\cdots$	2	$\{b, c\}$

As we shall see shortly, the functionality of this model will be targetted at specifying what we mean by a valid task schedule, that is, one which does not violate task precedence. There will be — or so we expect — a "best schedule", that is, one in which tasks are activated as early as possible. For instance, the following diagram pictures such a schedule concerning the "toy" organization above, where task box width is proportional to span (so, it is as if a time axis is considered on the horizontal direction):



A diagram of this kind will also be called a *process* schedule. It is clear from this diagram that sub-department C of department B does nothing in this process. However, it may participate in another process. Because our model does not allow the same organization to perform more than a single process, we have better enhance it in this direction:

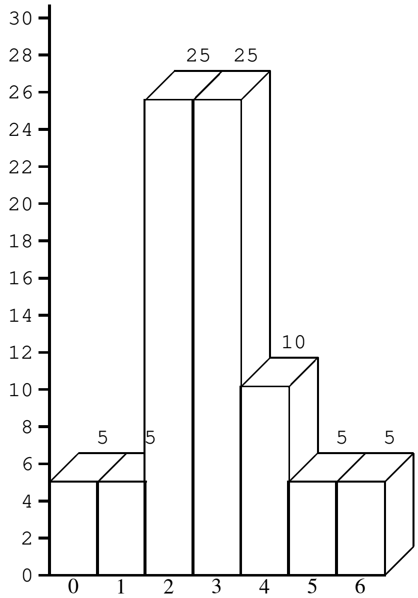
$$Org \quad \cong \quad OrgSt \times Processes \qquad \text{/* New version of } Org \text{ */} \tag{1}$$

where

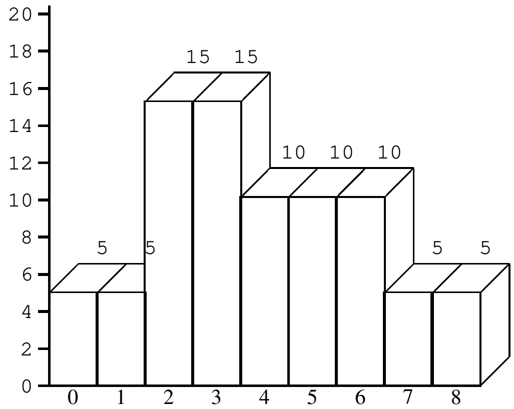
$$Processes \quad \cong \quad ProId \rightharpoonup Process \qquad \text{/* } ProId = \text{process identifier */} \tag{2}$$

Another aspect which will concern us has to do with resource management. Resource consumption will depend upon task scheduling. For instance, the following diagram illustrates the overall evolution of resource $r2$ for the organization and particular schedule depicted above [1]:

[1]Every $0 \leq i \leq 6$ in the x-axis represents time span $[i, i + 1[$.



It may be useful to adopt a schedule other than the "best" if, up to a predefined time limit (deadline), this will spread tasks along the time axis so as to render resource consumption as even as possible. For instance, should it be acceptable to shift tasks c and d later by 2 units of span, so that b and c do not overlap, than the histogram of resource $r2$ would be more even, as follows:



Part of the overall task of this project consists of equipping the datatypes just presented (or any elaboration of these) with functionality capable of express the above operations. Later on, part of the obtained model will be subject to a calculation process leading to a relational implementation.

More Information Available

The model underlying this project is an evolution of the $ActPlan$ and Orc data models which can be found in [Oli97] as exercises [2]. A CAMILA sketch of a project very similar to this one can be found in file `mii9900tp.cam` available from directory

`http://www.di.uminho.pt/~jno/tgz/mii9900mp.tgz`.

It is very strongly recommended that this additional material be carefully analyzed and studied as a preparation for this project.

Should you decide to run the `mii9900tp.cam` CAMILA source, the following sequence of operations is recommended (any problems should be reported to the course lecturers):

1. Load `mii9900tp.cam` + all other auxiliary `*.cam` files from `http://www.di.uminho.pt/~jno/tgz/mii9900mp.tgz`.
2. At MS/DOS or UNIX prompt level, type

[2] See, for instance, Exercise 2.62 (p.127) and Exercise 2.70 (p.133).

```
camila mii9900tp.cam
```

3. Once in the CAMILA shell, initialize the prototype you've just loaded via command line

   ```
   INIT();
   ```

   and load a built-in demo database by typing command line

   ```
   UPLOAD();
   ```

4. You are ready to run the demo: just type

   ```
   DEMO("Demo");
   ```

   and follow the instructions. In case your environment includes LaTeX, you may produce a report of the involved resource histograms, task schedule diagram *etc.* by typing

   ```
   LATEX();
   ```

   In case you don't have LaTeX available, you may inspect a particular report output by browsing file `_mii9900tp.cam.ps` available from the same directory as above.

Project Proposal

The following requirements should be taken into account in your project report:

1. Specify a datatype invariant

$$\text{inv-}Process \quad : \quad Process \longrightarrow 2$$

   ensuring not only that all tasks which precede some given task are are known tasks, but also acyclicness of the task precedence graph.

2. Specify datatypes $Time$ and $Schedule$ and, on top of these and of the other datatypes already specified in section **Starting Specification Model**, provide a formal specification of (at least) four of the following six functions, defining preconditions and/or auxiliary functions wherever convenient:

$$\begin{array}{rcl} bestSchedule & : & Process \longrightarrow Schedule \\ bestSchedule(db) & \stackrel{\text{def}}{=} & \ldots \end{array}$$

/ This function should compute, for each task, its earliest possible start, which is dependent upon the start of its antecedents and so on. Invariant inv-$Process$ is assumed. */*

$$\begin{array}{rcl} checkSchedule1 & : & Schedule \times Process \longrightarrow 2 \\ checkSchedule1(s,p) & \stackrel{\text{def}}{=} & \ldots \end{array}$$

/ This function checks the validity of schedule s relative to the task precedences of process p. */*

$$\begin{array}{rcl} checkSchedule2 & : & Schedule \times Process \times Time \longrightarrow 2 \\ checkSchedule2(s,p,t) & \stackrel{\text{def}}{=} & \ldots \end{array}$$

/ This function checks whether schedule s for process p meets deadline t. */*

$$\begin{array}{rcl} shiftSchedule & : & Schedule \times Process \times (TaskId \rightharpoonup I\!N) \longrightarrow Schedule \\ shiftSchedule(s,p,m) & \stackrel{\text{def}}{=} & \ldots \end{array}$$

/ This function gets schedule s relative to process p and shifts all tasks in m by the relevant time delays. The outcome should be the best schedule which accommodates such delays. */*

$$\begin{array}{rcl} getDepts & : & OrgSt \longrightarrow DepId^\star \\ getDepts(o) & \stackrel{\text{def}}{=} & \ldots \end{array}$$

/ This function lists the "paths" which describe all departments of a given organization o. */*

$$\begin{array}{rcl} genHistogram & : & Org \times ProdId \times Schedule \times DepId^\star \times ResId \longrightarrow (Time \rightharpoonup I\!N) \\ genHistogram(o,p,s,d,r) & \stackrel{\text{def}}{=} & \ldots \end{array}$$

/ Let o be an organization, p be one of its processes, s be a valid schedule for p, d be a department of o and r be a resource of d. This function should compute the evolution in time of the consumption by department d of resource r in process p, as implied by schedule s. */*

3. Build a CAMILA prototype for the whole project structured and documented in the same way as `sae.cam` [3].
4. (Optional) Port your prototype to WINDOWS and *hide* it behind a VISUAL BASIC user interface (see the CAMILA website for details how to interface the two technologies).
5. Calculate the relational model implementation of the Org datatype. Record the abstraction invariant (AI) relative to each step.
6. (Optional) Describe the outcome of your calculations in a database language or diagrammatic notation, *e.g.* Entity-relationship diagrams. (You have a lot about this in section 8.9 of [Oli97].)
7. Chose one of the operations you have specified above or invent a new one, and calculate its implementation at relational level.

[3] Also vailable from `http://www.di.uminho.pt/~jno/tgz/mii9900mp.tgz`.

References

[Oli97] J. N. Oliveira. *Métodos Formais de Programação*. University of Minho, 4^{th} edition, 1997. Textbook (489 p. in Portuguese [4]). English version under preparation at the time of writing.

[4] 846K gzipped PS file available from `http://www.di.uminho.pt/~jno/ps/mfp.ps.gz`.