International Journal of Information Technology & Decision Making
Vol. 3, No. 1 (2004) 179–208

TOWARDS A GENERIC OBJECT ORIENTED DECISION SUPPORT SYSTEM FOR UNIVERSITY TIMETABLING: AN INTERACTIVE APPROACH

SYLVAIN PIECHOWIAK* and CHRISTOPHE KOLSKI
LAMIH, Université de Valenciennes, Le Mont Houy
59313 Valenciennes, France
**sylvain.piechowiak@univ-valenciennes.fr*

Timetable management in universities is a complicated issue which uses up a great deal of human resources and is therefore very costly. This article deals with the analysis and design of an interactive decision support system for timetable management. This tool will be able to take hierarchical data organization into account and maintain coherence of the constraints on this data. Our research which has led to the creation of the EDT^{2002} tool has three aims. The first aim is to provide an open, generic tool which can be developed in many different ways. In order to achieve these aims, we have followed an object-oriented approach and we have defined object classes dedicated for the modelling of the timetabling problem. The second aim is to analyze the needs in timetable manipulation and to provide a generic organization so that the tool can be used in many structures. The third and final aim is to provide a easy to handle tool which can be used by various user profiles on personal computers with a low processing power.

Keywords: Generic timetables model; interactive timetabling; multi points of view; decision support tool.

1. Introduction

Every year, the task of the university educational managers is to organize timetables for the various courses or branches, whilst trying, as far as possible, to meet the "human" constraints of the teachers and students, along with the pedagogical constraints imposed by teaching progression and the "physical" constraints linked to material resources (rooms, equipment, etc.). Until now, managers have tried in vain to find "computerized" solutions for this problem. Some of them have used existing automated timetable generating tools:[6,18,33] However, they criticize this type of tool for the following reasons:

- The tools generally require powerful machines.
- The tools better feet the needs of secondary schools. Their timetables are built on a weekly basis. The development of a timetable for a year only duplicates the weekly structure. These automated generation tools are also well suited to the examinations[8,19] or conferences.[14]

- Fully automated tools are not efficient when the constraints cannot lead to a valid solution (impossibility of building a clash-free timetable). In these situations, the tools do not provide any support in explaining the causes for the lack of solution. Nothing is given to determine which constraints must be relaxed to bring about a solution.
- The quality of these timetables also depends on the exhaustiveness of the constraints. In an university, it is impossible to collect and to formalize all this information. Expertise of timetablers is the key.

Other timetablers have developed their own tools in order to assess the quality of the timetables. These tools make administrative tasks easier (economic or financial). However timetable management is often put apart (design, exploitation, etc.). In addition, solutions provided are too specific to be easily used in other organizations.

During our research work, we have noted that managers generally use an old existing timetable as the starting point. They do not create it from scratch but they adapt a timetable designed for another course or for a previous year. They really need an interactive tool to make adaptations easier.[21] They require "user-centered" tools, according to the terminology found in Refs. 12 or 27. Some limited tasks (like room allocation) can be realized by the machine.

Besides, more advanced techniques have been explored by the constraint programming community for the design of timetables.[25] These techniques allow users to give some guidelines to the system and get quicker or more adequate solution. However, this approach is still focused on systems. We have chosen a different point of view since our work is dedicated to the end user. We aim to help the user (in the sense given in Ref. 31) in his/her decision-making. The tool required must above all be interactive, adaptable and open and must have ergonomic qualities in the sense given by Refs. 4 and 13.

We will describe first the current organization at the University of Valenciennes and Hainaut-Cambrésis (UVHC) and more specifically at the Institute of Sciences and Techniques of Valenciennes (ISTV). We will then identify the generic characteristics of such organizations. The needs for modelling and handling timetables will then be identified. In order to lead to a generic tool, we define various user profiles. We will then present the EDT^{2002} tool, written in DELPHI object-oriented language and we will describe the different objects defined. Finally, we will give results we have obtained using EDT^{2002} and we present perspectives to get the highest level of genericity for this kind of tool and to make it designed for a multi-user configuration.

2. Study of a Representative Case

2.1. *Global analysis of the ISTV*

It was first necessary to analyze the pedagogical university environment into where the interactive timetable management support tool will be integrated. The analysis

was carried out using interviews and observations, along with written information concerning the activities of educational managers, room administration managers, equipment managers, administrative personnel and students. The research is performed at the ISTV (approximately 4,000 students, 800 teachers and 80 rooms).

The ISTV covers various undergraduate and postgraduate level courses. The courses last from one to three years. For example, the first degree and the master's degree last for one year each whereas the undergraduate course lasts for two years and the professional degree for three. The students enrol at the beginning of the academic year, generally in September. The pedagogical syllabus for each course is known *a priori*. The syllabus specifies the subjects to be studied, the number of hours for each subject and some pedagogical information.

According to the pedagogical needs and the physical resource characteristics, each course is structured into groups. Each group can be divided into sub-groups, etc. For example, the number of students for a seminar group (TD) is limited to 30. Numbers also depend on the material resources necessary. For example, the number of students for each computer during practical work sessions (PW) is limited to 2. In addition, each computing room has a limited number of work stations which are either PC type, X terminals, specific or non-specific work stations. So, each computing PW session depends on the size of the groups. If it is impossible to meet all of the constraints, groups may have to be merged or split.

The teachers work according to their subject and their specialist field. It is traditional for teachers to keep their teaching modules from one year to the next. On the administrative side, the teachers have to cover a minimum number of hours. This number depends on teachers grade. Each hour is weighted by a coefficient according to the type of teaching module and the grade of the teacher (Professor, Lecturer, Contract teacher, etc.). When a teacher accepts the responsibility of a class, he/she is obliged to respect the number of hours planned for by the educational manager. In the event of absence, the teacher must organize replacement hours. He/she therefore has to know the exact availability of the resources concerned by the session. This organization ensures that all the students following the same course have the same number of hours of teaching.

Each year, level of a course is organized by an educational manager who creates the timetables with the teachers and also with the managers of other courses. The teachers can perform their teaching hours with different courses. When the timetables are created, the educational managers do not explicitly consider the rooms (except for extremely specific lessons).

Once the timetables have been created, rooms and equipment are allocated by various managers who can be members of the administrative or technical staff, or even teachers, according to the type of room (ordinary, computing PW rooms, etc.). For example, at the ISTV, there is one manager who organizes the lecture theatres and seminar rooms (ordinary). Another manager looks after the computing rooms (a teacher of computer science). A third manager supervises the audiovisual department rooms, etc. These people can ask the educational managers for timetables

modifications if the resources are not sufficient. The timetables are developed in two stages: development of the timetables without taking rooms or equipment into account and then allocation of rooms and equipment. Generally, the first stage involves a scheduling problem; the second involves a resource allocation problem. When aiming to create a generic tool, it is important to deal with these two aspects, as it is also emphasized in Ref. 3. The two stages are performed by different people. It can also be seen that the development of timetables and their management is not a linear process.

The timetables are used by the administrative personnel in order to monitor the teachers' work with regard to their pay, to the purchase of new equipment or maintenance of existing equipment, to back up requests for new premises, etc.

On the pedagogical side, the timetables are studied in order to reveal certain problems such as excessively heavy workloads, or incorrect distribution of teaching sessions. Therefore it makes it possible to correct and improve the timetables for the following years.

2.2. *Analysis of needs*

The quantity of resources available is increasing, although not at as great a rate as the number of courses and their diversification. This makes management an even more delicate problem. The need for a support tool has therefore become primordial. As the aim is not to replace the current actors but to help them, it is important to consider the various profiles of the future users, in particular as regards to their skills in computing[24] and their role (design, manipulation, consultation of the timetables). The tool must therefore be interactive, extremely user-friendly and able to display a great quantity of information. The goal is to manage this information to keep it coherent, to seek and explain incoherent elements as soon as they appear. These properties are characteristic of decision support tools as they are defined in Ref. 16.

Finally, the fact that the study was performed in a real situation makes it to be slightly wary. It is preferable to begin by providing support tools based on the current situation and then to develop them, rather than by imposing tools which will radically change work habits and risk total rejection (cf. Nielsen's acceptability criterion[26]). Now that the context of the research project has been described, we will give a detailed description of the modelling of the problem in order to situate better the constraints exploited subsequently.

3. Modelling of the Timetable Problem

In an abstract view, the timetable problem consists in distributing over time pedagogical activities which require resources (teachers, groups, rooms and equipment). In order to respect the usual vocabulary, these activities are called teaching modules and each occurrence of an activity is called a session. The sessions have a duration. The timetable problem therefore requires the modelling of the activities, the resources and the time.

3.1. *Modelling of the pedagogical activity*

The pedagogical activity is modelled using two entities: the **subjects** and the **teaching modules**.

Each subject is characterized by: its name, its pedagogical description, its type (lectures, seminars, practical work, or others) and its discipline (computer science, mathematics, chemistry, etc.). The type is used during the timetabling step to control the total duration of each subject in a day. The discipline is used during the resource allocation step to ensure that resources are well adapted.

A teaching module is characterized by a name, a total number of hours, a default duration (the duration suggested by default for each module session) and a set of resources. Experience in the field led us to separate the resources into two sets: imposed resources and necessary resources. The distinction made between these two types of resources makes it possible to set certain constraints as soon as the teaching module is described. For example, when the timetables are designed at the ISTV, it is decided that such a module will be taught by such a teacher with such a group. The necessary resources are described as a type. The real allocation of these resources is done once the timetables have been developed. For example, it is possible to specify the type of room required and then to allocate a room belonging to this type later during the development of the timetable.

In addition to these characteristics, each teaching module must be situated in relation to the others. For example, the E2 module may only take place once the E1 has been completely finished. Other dependencies can exist. For example, it is necessary for a student to have passed the first year undergraduate course before enrolling for the second year of this course. These dependencies are not dealt with here.

We have identified two types of (temporal) dependence: *global dependence* and *specific dependence*.

The global dependencies situate each teaching module relatively to the others. At each teaching module e is associated a list which contains couples (e_i, dep): e_i is a teaching module and dep is a global temporal dependence between e and e_i. The global temporal dependencies we have provided for are taken from the Allen logic.[2] Each teaching module has a starting date dd (the date of the first session) and an ending date df (the date of the final session). It is thus possible to situate $e1(dde1, dfe1)$ in relation to $e2(dde2, dfe2)$.

To express "*starting the seminars only after 2 or 3 lectures have been given*" is not possible with the global dependencies. In order to deal with these problems, we have defined the "specific dependence". This notion makes it possible to situate a teaching module in relation to another according to the number of sessions already planned and their starting and ending dates. So for each teaching module tm we also manage a list which contains couples $(e, nbSessions)$ in which e is a teaching module and $nbSessions$ is the number of sessions of e which must have taken place before it is possible to plan tm sessions.

3.2. *Modelling of resources*

The resources considered are the physical entities necessary for the preparation of timetables, that is, to say rooms, teachers, groups, students and equipment. In order to take in most of the configurations imaginable, the resources are organized according to a trellis structure. Each resource R has daughter resources and can be the daughter of several other resources. Figure 1 shows an example.

Each reference to a resource R also concerns all the daughters of R, as well as all its "granddaughters", down to the lowest level of the hierarchy. Moreover, as soon as a session concerning R is placed, this limits the degrees of freedom of all its mother resources.

In the ISTV, this organization of the resources into trellis structures is used for the groups and for the equipment.

Four categories of resources are considered: the rooms, the teachers, the groups, the students and the equipment. Each resource is described with data to identify it (like its name) and data to characterize it (like capacity, speciality, etc.).

3.3. *Modelling of time*

Two fields are used in order to represent time: the instant and the duration. In addition, for the timetabling problem, we add the notion of frequency. The notion of frequency can be expressed using two other notions: an event E takes place at a frequency f if the length of time separating two consecutive appearance instants of E is greater than f. For example, to say that "there is a mathematics session once a fortnight" is to say that the length of time separating two consecutive sessions is 15 days. The duration of each session is distinguished from the length of time separating the sessions. It should be noted that this representation of time is more flexible than that generally chosen in secondary schools.[11,22] In classical timetabling problems each day is divided into a set of time slots which are fixed. In this case, each session can only take place in one or several consecutive slots. Here a session can begin at each time in a day.

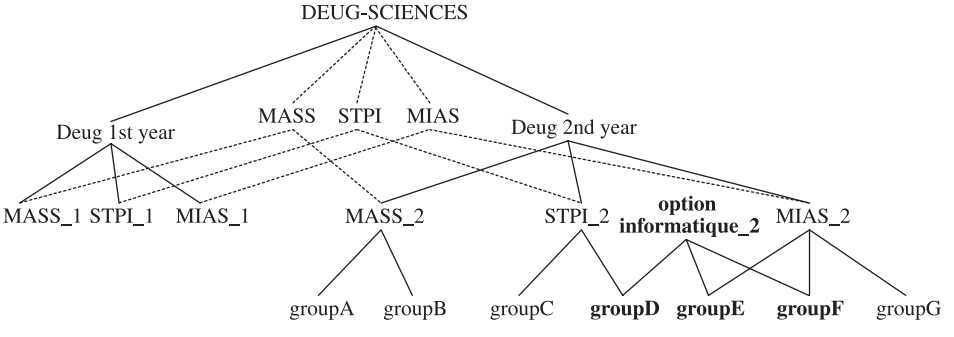

Fig. 1. Example of group hierarchy in the ISTV.

3.3.1. *Granularity of time*

The granularity of time makes it possible to link representations of time with various degrees of sharpness. At the lowest level, we find the basic temporal representation (for example, 15 minutes). At the higher levels, we find complex notions of time (for example, the hour, the half-day, the day, the week, the fortnight, the month etc.). It is important to be able to apprehend time according to different granularities if temporal properties are to be expressed easily. These properties appear frequently in the preference constraints related to the quality of the timetables sought.

In this research project, the representation of time was chosen according to the problem to be dealt with, and according to the observations made on site. We therefore merged time with the axis of real numbers and took an arbitrary date as a point of origin.

3.3.2. *Temporal entities*

In order to model time, the following entities are defined: date, time, duration, slot and calendar.

A date refers to a triplet (day, month, year). Using this triplet, the value associated to it on the day axis is defined. A time is a whole number included between the minimum value $HMin$ (8h00) and the maximum value $HMax$ (19h00). These two numbers correspond to times in relation to a date.

A duration is a number included between $DMin$ and $DMax = HMax - HMin$. $DMin$ represents the smallest temporal unit available. The duration is given by the user.

A time slot refers to a temporal interval during a day. It is characterized by a couple (H, D) in which H represents the starting time of the slot and D its duration.

A calendar is a set of dates. Each date is associated with a state (available or non-available). For practical reasons, other values were used in order to make the tool better adapted to usual practices in the university. For example, for the groups, the values include leave, examination, in-company period, and, on work. Each resource has its own calendar which makes it possible to differentiate the availabilities of all the resources. For example, a room can be made unavailable from 1st to 15th December 2002 for renovation work or from 12th to 16th June 2003 for a conference. A particular teacher can also be made unavailable outside the period from 1st September to 31st December because of a secondment, for example.

Each teaching module is also associated with a calendar to precise when it can be planned during the year. For example, we can decide that a teaching module must be planned on Friday.

3.3.3. *Sessions, schedules and bookings*

A session corresponds to a temporal event of a teaching module on a given date, during a specific slot. The characteristics of a session are: its teaching module,

its date, its slot, its equipment and its room. The schedule of a resource is the set of sessions in which this resource appears. In this way, it is possible to consider the schedule for a room in the same way as that of a teacher. The schedule notion also appears at the teaching module level: in this way, it is possible to manipulate the evolution in time for a specific teaching module, in order to make sure that the total time volume provided for has taken place, for example.

A booking corresponds to an option placed on the occupation of a resource. In relation to a session, a booking is not associated to a teaching module. Bookings give flexibility to the use of the tool, especially during the timetable creation stage performed by the various managers. For example, a room can be booked by a manager for a group, without knowing exactly which teaching module will take place in it. The other managers are aware of the booking and will avoid using the room. If they do use the room, the room manager will have to find an alternative by suggesting another room.

3.3.4. *The courses*

The notion of a course makes it possible to limit the number of resources to be considered and thus to limit the quantity of data to be processed. Indeed, a course manager only needs to know the schedule for the resources he/she uses. A course is therefore characterized by: its name, its manager, a list of the resources working on the course, a reference calendar, a list of teaching modules and a list of subjects. The reference calendar is shared by all the resources in the course.

For example, in the academic year 1999–2000 the quantity of data for the ISTV is approximately 800 teachers, 70 rooms, 600 groups, 1,200 subjects, 3,000 teaching modules, 24,000 sessions. For the IUP Electrical Engineering and Industrial Informatics course (which is part of the ISTV) there are 80 teachers, 20 rooms, 80 groups, 135 subjects, 405 teaching modules, 2,450 sessions. Given this information, it is obvious that it is best for the educational managers for this course to consider only the data relating to it.

However, certain clashes will only be detected when the various courses are collated in a common data base. They are mainly clashes concerning teachers or rooms as teachers frequently work in several teaching modules and rooms are often used by several courses. On the other hand, it is rare to find a group which belongs to several courses.

3.4. *Description of constraints*

The constraints to be respected can be classified in two groups: physical constraints (also called hard constraints) and preference constraints (or soft constraints) which are linked to the pedagogical quality of the timetables. The physical constraints make it possible to be sure that any particular problem has a solution (resolvable character) whereas the preference constraints are generally taken into account when a solution is being improved (optimization character).

3.4.1. *Physical constraints*

These constraints cannot be violated, otherwise clashing situations would arise. There is a physical resource clash between two sessions $s1$ and $s2$ if these sessions have a common resource for a non-null duration.

The physical constraints integrated into EDT^{2002} are the following:

- No resource r may be occupied by two different sessions s_1 and s_2 at the same time (and date): $\forall s_1 \forall s_2 \forall rc(s_1 \neq s_2 \wedge r \in resources(s_1) \wedge r \in resources(s_2)) \Rightarrow (date(s_1) \neq date(s_2))$ where $resources(s_i)$ is the set of all the resources affected by s_i; $date(s_i)$ indicates date and time of s_i.
- As soon as a resource appears in a session, all its daughter resources are also occupied by the same session recursively: $\forall s \forall r(r \in resources(s) \Rightarrow (\forall x(x \in subresources(s) \Rightarrow x \in resources(s))))$.

  In the same way, as soon as the resource appears in a session, none of its mother resources may be occupied at the same time (because all of these resources have resources in common). These are constraints linked to the hierarchical structure.
- It is forbidden to put more students than there are places in a room.
- The total time of the sessions in a teaching module may not exceed the volume planned.
- The calendars are respected for all the resources (sessions may only be placed during resource "working" days).

These constraints are part of EDT^{2002} — they cannot be modified by the users but they can be ignored. Thus, in all the orders available to the user, only actions which respect the constraints can be performed. For example, when a user wants to allocate a room to a session, only the rooms which respect all the constraints (free, suitable size and type) are suggested. This way of proceeding makes it possible, amongst other things, to avoid clashes appearing. We will discuss this further in the paragraph concerning the treatment of clashes. The user can decide to use the automatic timetabler and then to modify the result.

3.4.2. *Preference constraints*

During interviews, we were able to identify preference constraints. They are different to the physical constraints because they may be violated: in this case, the timetable obtained is of a lower quality. Typically, these constraints are used to express what a "good" timetable should be for the students and for the teachers. Several examples of these constraints follow:

- To make the duration of the teaching module sessions uniform.
- To avoid "gaps" in the timetable.
- To avoid lectures at the end of the day.

- To divide the work load of the teachers and the students, if possible, over the whole teaching period. The aim is to avoid overloaded periods. To divide up the work load, several levels of time granularity can be considered. For example, the day, the week, the month, the semester and the year can be distinguished.
- To avoid starting before 08.30 and to avoid finishing after 18.30.
- To encourage timetable regularity.

It can be seen that the terms used in the expression of preference constraints are not precise (avoid, should be, if possible, ...). These constraints are more difficult to formalize than the physical constraints and their processing is more delicate.

The preference constraints can be classified according to their degree of importance. The constraints with higher degree are considered with higher priority. In EDT^{2002} the preference constraints are taken into account at two levels.

At the first level, the constraints are taken into account during the interactive placing of sessions by the user. When seeking sessions which the user can place in a specific slot, the sessions found are classified according to criteria linked to the quality of the timetable. The user will find the best suited session at the top of the list.

At the second level, the preference constraints are taken into account during the subsequent analysis of the timetables. In general, the modifications suggested by this analysis are translations in time of the sessions or permutations of resources. For example, if a session can be held at 09.00 instead of 08.00 without causing clashes, it is changed. If it is possible to change $room1$ and $room2$ for sessions $s1$ and $s2$ in order to reduce the movement of groups, then these two rooms are changed.

The modifications brought about by preference constraints are suggested to the user who then decides whether to accept or reject them. In this timetable "improvement" stage, a good number of users expressed the need to be able to come back to previous modification decisions. We noticed this ourselves during another research project on the design of a decision support tool in the field of diagnostics.[20,28] However, the decision making cancellation order has to be accompanied by an "intelligent" backtracking tool which guides the user in his/her reasoning. In this way, the user should be able to find the reasoning applied previously before going back. A detailed illustration of the processing of preference constraints by EDT^{2002} will be given in Sec. 4.3.

4. Architecture of the Interactive Decision Support Tool

The design architecture targeted is shown in Fig. 2. At this step, users are not specialized in a particular task: they have the same tool and can do the same actions on the data. Each user must resolve a local part of the global problem. Each user has different visions (called points of view) of the part of the problem he/she is entrusted with. The points of view the user has enable him/her to see the overall problem he/she is dealing with. On the other hand, each single

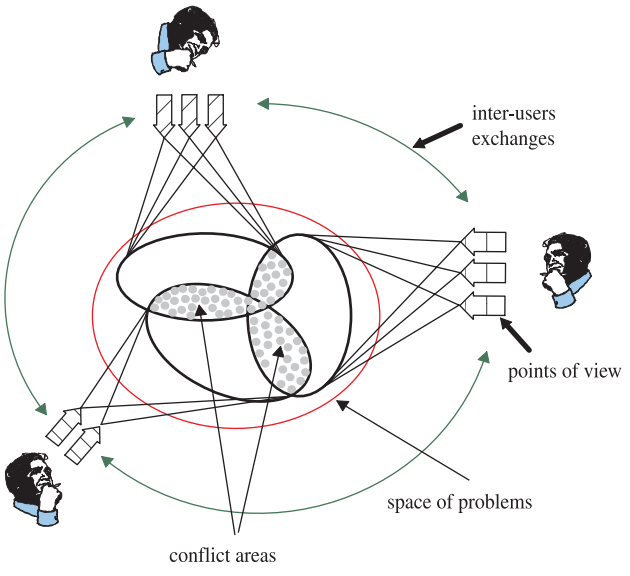

Fig. 2. Design architecture of the interactive tool targeted.

point of view is not sufficient for the user to have a complete view of the part in question.

The global problem consists in designing and managing the timetables of all the resources. Each one of the educational managers only sees a part of this problem. As the teachers, the rooms and the equipment are shared resources, clashes inevitably appear when the timetables are being developed.

In this article, we concentrate our presentation on the interactive tool used by each user. Inter-user exchanges are not considered; the users work sequentially, in turn. The evolution of the tool towards a collective decision support tool is one of the envisaged perspectives of this research project.

4.1. *Visualization of the timetables and points of view*

For our project, we sought a simple graphic representation of the timetables. It immediately seemed important for the representation to be accessible to all. It is also necessary to consider constraints linked to the computers used. For example, most of the users concerned have a 17″ screen, which makes it necessary to privilege light-weight representations.

The two types of view most commonly used are: the weekly graphic view and the yearly graphic view. In both cases, visualizing a timetable consists in representing sessions of one or several resources on a plan. On every view of a timetable, two axes are represented: the resource axis and the time axis. It should be noted that there are no restrictions concerning the resources which appear on the resource axis. In other words, it is possible to visualize on one screen the timetables of

several groups, teachers, rooms and elements of equipment. This has two major advantages. Firstly, during the modification phase, the user is informed on one screen about the availability or occupation of the resources he/she is dealing with. Secondly, it enables the user to understand why the tool refuses to place a session in the slot he/she has chosen (role of explanation).

The manipulation orders are given via screens. They are invariably available in each view. The orders allow the user to "navigate" around the timetables and modify them whilst controlling clashes (and avoiding them as far as possible). In order to make the user's task easier, the constraints are taken into account in a dynamic fashion. Thus, for example, when a session is being moved, the tool begins by finding the moves allowed and then suggests them to the user. This approach limits the user's search space to the areas in which sessions can be placed without creating clashes.

We defined the notion of points of view in the framework of previous research into cooperative reasoning and its application to interactive diagnosis.[20] The basic idea consists in representing a model according to different angles of vision, called points of view. Each point of view only concerns a part of the model, but the set of points of view covers the entire model. The reasoning performed using each point of view is propagated to the other points of view, and this guarantees the coherence of the set of points of view.[28] This approach was validated by the development of an interactive tool in a case in which the reasoning concerned a diagnosis. Although the users of the tool were not used to manipulating computerized tools, the results of real-situation experiments were very convincing. In fact, the diagnoses obtained using the tool were better on average (more accurate, more reliable and obtained more quickly) than those obtained without the tool.

The notion of point of view is used here to represent various elements of information for the same indicator (time, resource) without overloading the screen. The points of view retained here are: the resource point of view (teacher, group, room and equipment), the teaching module point of view and a specific point of view in which the user chooses the information he/she wishes to see. In this last point of view, the user can see, for example, the names of the rooms, their capacity and their occupation rate as well as the names of the teachers who use the rooms in question. A button enables the user to switch from one point of view to another whilst keeping the general appearance of his/her view along with the same orders (addition, removal, etc.).

In Fig. 3, we give an illustration: it is part of the schedule for the *DESS_ICHM* class and its sub-groups 1 and 2, from the point of view of the teachers and the rooms for a period chosen by the user. It is an annual view in which one week is represented by one line. Each line has as many sub-lines as resources selected (here the resources are the groups: *DESS_ICHM*, *ICHM1* and *ICHM2*). The name of the teacher for each session visualized is displayed (teacher point of view).

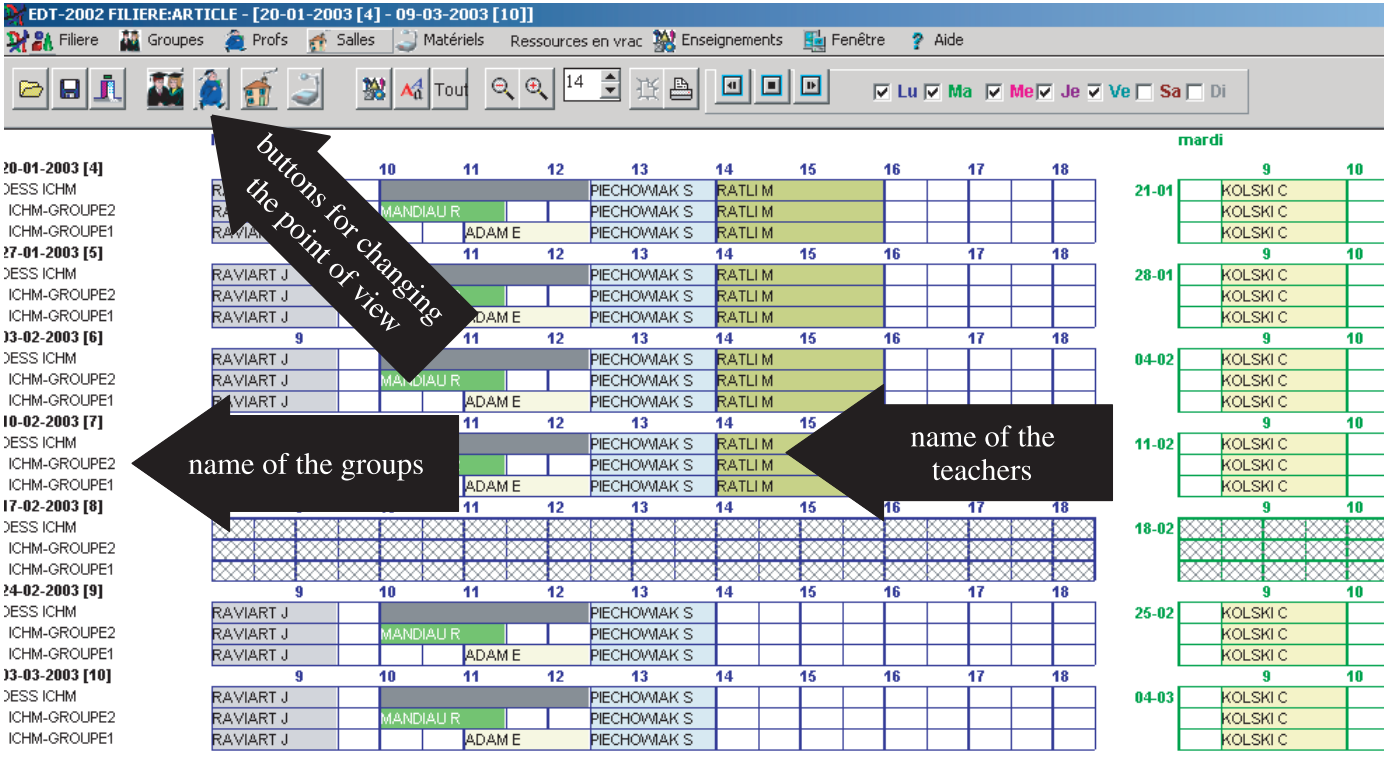

Fig. 3. A timetable seen from the teacher point of view.

4.2. *Action possible using the graphic interface*

We have defined three types of user[29] — the designers (pedagogical managers), the analyzers (administrative managers) and the consultors (teachers, students). They have specific needs which lead to the definition of specific action classes. This action is described hereafter.

4.2.1. *Action intended for the designers*

We assume that the creation of timetables is done at first in a localized manner for a course or branch; the various timetables are then collated to reveal clashes and resolve them. Local treatment of timetables does not exclude the incremental use of information. It is in fact the way in which the managers work at the moment as they try to integrate as early on as possible the non-availability of teachers who have already been scheduled in on other timetables.

When a timetable is created, the educational manager wants to place sessions while creating the timetable whilst respecting the constraints or to find any clashes and resolve them or to find improvements for the timetables or to find time slots or to modify the timetable (by swapping two sessions or two groups of sessions, by increasing or reducing the duration of one or more sessions).

At the present time, the educational managers work mainly on paper or using a computerized spreadsheet which enables them to perform graphic manipulations and gives neater printouts than those done by hand. The method starts with a calendar on which the dates of leave, in-company periods and examinations are placed. Then the timetable is created based on the timetable from the previous year.

We have already pointed out that automated generation tools make it possible to obtain timetables, but that they require a fastidious definition of constraints to check. Here, the physical constraints are numerous but they are easy to describe. Moreover, an automated generation method which only uses physical constraints provides a timetable of a very low quality from a pedagogical point of view. This leads to the necessity of considering the constraints which improve the quality of a timetable. However, these constraints are numerous and more empirical in nature, making them difficult to formalize and use. In addition, the constraints can be respected or violated. In our approach, it is the user who decides whether a constraint (which we call a preference constraint) is to be respected or not. Later in the paper, we will present an example of the integration of preference constraints into the phase concerning the allocation of rooms to sessions (Sec. 4.3).

4.2.2. *Action intended for the analysis of timetables*

The timetables are used by two main actors — the educational secretary who moves sessions if a teacher is unable to lead a teaching module, and the educational manager who monitors the running of the course by making regular overall assessments.

In order to use the timetable, the secretary and the manager therefore want:

- To place sessions in order to adapt to the evolution of the timetables throughout the year.
- To find free slots to place new sessions.
- To move one or several sessions.
- To obtain assessments (to report back to administrative managers, especially as concerns the payment of hours worked, to know the occupation rates of the rooms or the equipment and thus anticipate maintenance operations, to show up needs in personnel or to reveal pedagogical shortcomings).
- To circulate the timetables and modifications to the students, the teachers and the various managers.

4.2.3. *Action intended for the consultation and circulation of timetables*

Traditionally, the timetables are circulated on paper — in this case, the framework used by the educational managers to create the timetables is the same as that given to the teachers (in particular when they use a spreadsheet or when they write on a chart they have devised themselves or taken from another department). This remark caused us to keep the same graphic representations for use (using a PC) and for circulation.

We also noticed during our on-site analyses that the choice of representation is often linked to the habits of the manager and the branch. Some people have already attempted to suggest new representations, which are generally graphic and based on the use of a spreadsheet. Unfortunately, these representations are often difficult for other people to read and understand. For example, some managers have used different representations according to type of teaching module on the same schedule. This implicitly translates the placing of constraints on the timetable (for example, set PW days). In the appendix, we provide an example of such a schedule with a representation for lectures and seminars, and another representation for practical work.

We therefore tried to find a compromise adapted to the greatest number of people. This comprise consists in facilitating the visualization of the sessions for the teachers, the groups, the rooms, and the equipment or for a combination of these resources, for a week or for a part of a year. Moreover, these two visualization modes meet the user requirements expressed during our on-site analysis which had revealed the necessity of limiting the number of representations of the information.

4.3. *Allocation of resources in an existing schedule: example of the rooms*

The allocation of rooms is performed by the room manager once the timetables have been created by the educational managers. The rooms are divided into several types — lecture theaters, ordinary rooms, specialized rooms (computing, chemistry, physics, automation, ...). They are characterized by their capacity, that is

the number of people they can accommodate, as well as by their geographical location. The location is limited to proximity zones because the important thing is to know whether one room is close to another or not, more than the actual distance between the two.

Up to now, the manual allocation of rooms has been carried out without too many problems, thanks to the experience acquired over the past twenty years by the various managers. In addition, the allocation of rooms rarely leads to clashes which bring about considerable modifications in the timetables. This can be explained mainly by the approach applied. Before the educational managers even begin developing the timetables, they meet together to pinpoint the critical teaching modules (i.e. those which need critical resources — rooms or equipment). These teaching modules are placed on the various timetables in such a way as to satisfy all requirements and avoid clashes. For example, the lecture theaters are allocated to the various courses so as to cover global needs. This is done using the booking notion explained previously. Thus, for example, for a specific group, a lecture theater is reserved for four half-days every week from 15th September to 30th May.

When the rooms are actually allocated to sessions, the user must choose a room from all the rooms so that the constraints are respected: the room must be free and have a sufficient capacity to accommodate the students concerned by the session. Extra constraints (preference constraints) must also be taken into account not only for the quality of the timetables but also in order to anticipate needs which are unknown when the timetables are developed (for example, organization of work meetings or seminars).

We suggest helping the user to decide which room to allocate to which session in the following manner. Firstly, a list is drawn up of the free rooms which have a sufficient capacity to accommodate the group of students. Then these rooms are classified using a function which depends on preference constraints or heuristics. Thus the user can either trust the tool and choose the first room in the classified list, or choose another room on the list for reasons which are not known to the tool. The parameters of the function used to classify the rooms in the list can be chosen by the user who can decide to apply all of the preference constraints or just some of them.

The preference constraints currently in use were defined in collaboration with the room managers: regularity of room allocation, limitation of group movement between rooms and anticipation of unknown events.

The user can decide whether to take preference constraints into account or whether to ignore them, as is shown in the dialogue window in Fig. 4. In addition, this user can decide to use the automatic allocation module. This module is based on an intelligent backtracking algorithm like backjumping.[32]

5. Treatment of Clashes

In the section concerning constraints, we defined the notion of clashes: a clash is characterized by the sharing of resources between several sessions

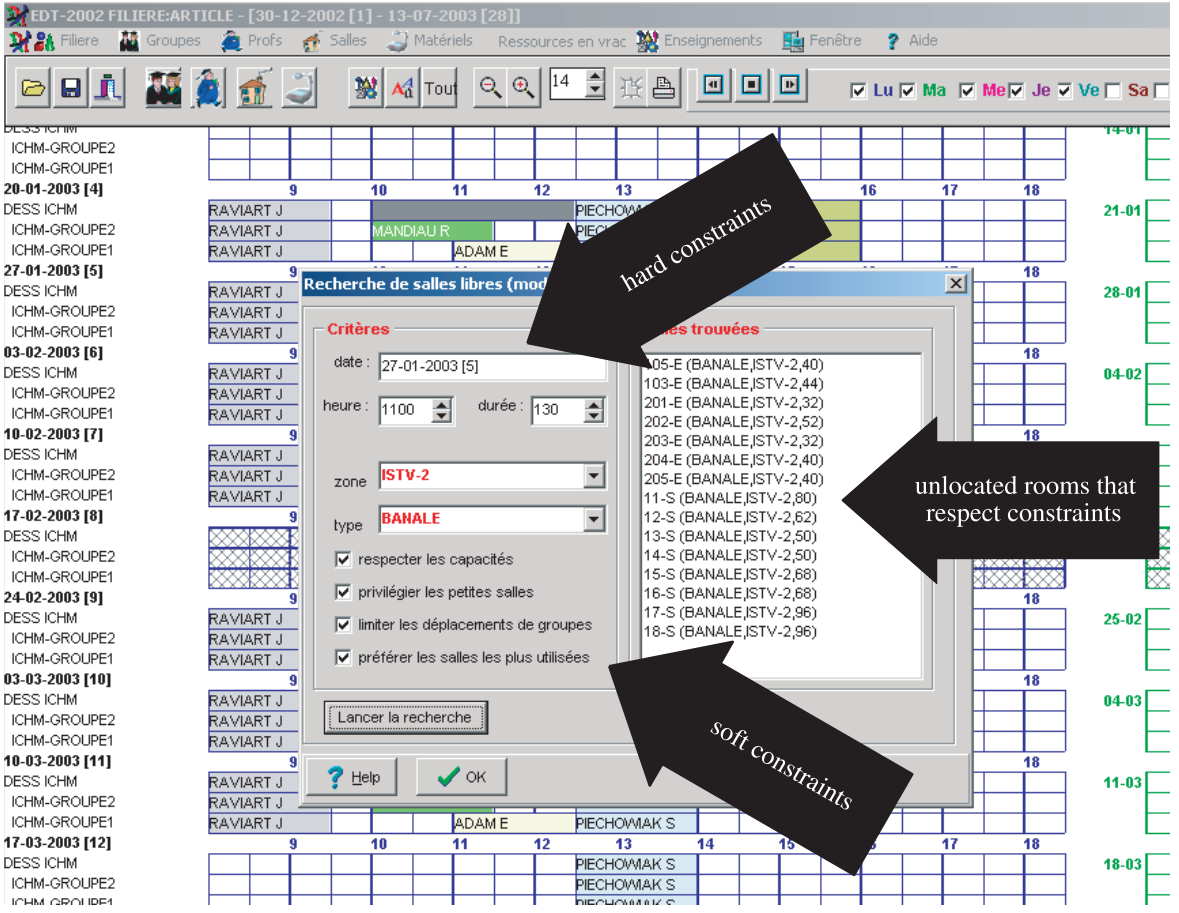

Fig. 4. Dialogue window for the allocation of rooms with consideration of preference constraints.

at the same moment in time: $\forall s_1 \forall s_2, clash(s_1, s_2) \Leftrightarrow ((s_1 \neq s_2) \wedge (overlaped(date(s_1), date(s_2))) \wedge (resources(s_1) \cap resources(s_2) \neq \emptyset))$.

The basic idea of our work is to design an interactive decision support tool which will allow the placing of sessions on a schedule whilst guaranteeing that there are no clashes. Nevertheless, several reasons oblige us firstly to accept that there will be clashes and secondly to resolve them.

Firstly, during our on-site tests with the users, it appeared that during the development of timetables, some users prefer to have the possibility of placing clashing sessions temporarily and then correcting these clashes, rather than only being able to consider non-clashing situations.

Secondly, in reality, the timetables are created locally for the branches, without knowing the schedules for other branches. When all the schedules for all these branches are grouped together, clashes generally appear because of teachers who work in several branches, for example.

Finally, our tests revealed that even for users who wish to avoid clashes when placing sessions, it is important for them to be able to inform of the reasons why placing certain sessions is impossible. For example, when the user clicks the mouse on a grid in order to place a session, the tool begins by searching for a list of sessions which may be placed without creating clashes. If there is nothing on this list, it is better to indicate why that is so. Two explanations can be given. The first is that all the sessions have been placed. The second explanation is that for all the sessions which still have to place, there is systematically one of the resources which is already used for another session at the same time. In this case, it is best to give the user the list of sessions which hinder the placing; the user can then decide whether to rethink the choice of place or whether to come back on sessions placed beforehand.

5.1. *1st approach: avoiding clashes*

The method which consists in avoiding clashes is well adapted when the timetable is available and the user wishes to modify it slightly, or when it is available on paper and the user wishes to enter it on a computer. Avoiding clashes then comes down to avoiding data entry errors. The users who appreciate this method are the room managers because they allocate the rooms to the sessions, once the sessions have already been scheduled.

This method is also advantageous for the creation of timetables because it constantly limits the space of the sessions which can be placed in a given slot. In this case, this method alone is not sufficient. We noted that when wanting to place a session in a slot, the user is put off if the session he/she thinks he/she can place is not on the list of sessions possible. To solve this problem, the user would like to be given the reasons for this absence (teacher, room, group or one of the descendants or ascendants already occupied). Another way to tackle this problem is to accept clashes and to resolve them afterwards. The danger of this method is that there

could be a lot of clashes which would be impossible to resolve without starting the work again from the beginning.

By allowing the manipulation of incomplete sessions (no teacher, no room), a different approach is adopted which consists in creating a timetable without resources (or the fewest resources possible) and then allocating the resources afterwards. The problem with this approach is that the constraints are not integrated early enough. At the ISTV, it is this approach which is used, since the rooms are allocated once the timetables have been created.

These two approaches can be compared to the prospective and retrospective approaches which are well known in the field of constraint programming.[32] It is shown that the approaches which involve forward checking through constraints are more efficient than those which check if constraints have been respected after the choices have been made.

5.2. *2nd approach: interactive resolution of clashes*

A clash can be resolved in several different ways.

The first way is to delete certain clashing sessions and to request the user to place them elsewhere on the schedule. This method is only viable if the dependencies between clashes are taken into account and if the number of clashes is reasonable. For example, let us consider the case shown in Fig. 5 in which the sessions $S1$, $S2$ and $S3$ are scheduled on the same day in the same slot.

In this case, two clashes are linked: $S1$ with $S2$ and $S2$ with $S3$. The clashes can be resolved by deleting the three sessions and trying to place them elsewhere, by deleting $S1$ and $S3$ and placing them elsewhere, or by deleting $S2$ only and by placing it in another slot. The choice of solution can depend on various criteria. It is possible to choose the solution which modifies the fewest sessions (just $S2$ in the example). It is also possible to choose the solution which preserves the quality of the timetables.

The second way involves grouping together the resources involved in different sessions. The teaching modules may be shared by several groups in different

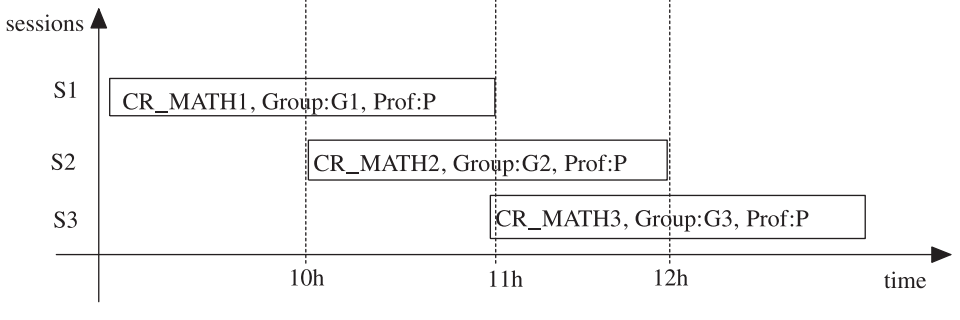

Fig. 5. Example of clash with groups.

branches. When the data relating to the different branches is merged, these teaching modules are declared several times. This can cause clashes with the teachers, for example, because for these shared teaching modules, the same teacher gives a lesson to different groups at the same time. The resolution of these clashes is easy: the groups in the two teaching modules have to be merged.

The third way of resolving clashes is to reduce the duration of the sessions so that the time interval associated to the clashes is eliminated. In the previous example, the two clashes take place in the time intervals [10.00–11.00] and [11.00–12.00]. There are several possibilities: reduce session $S1$ to the [09.00–10.00] interval and session $S3$ to the [12.00–13.00] interval or reduce session $S1$ to the [09.00–10.30] interval, session $S2$ to the [10.30–11.30] interval and session $S3$ to the [11.30–13.00] interval.

The fourth way is to eliminate the "cause" of the clash, that is, the teacher, the group or the room. Once the clash has been eliminated, an attempt can be made to allocate the sessions to other teachers, groups or rooms.

This method can be extended so that it is capable of exchanging the resources for various sessions. For example, the teachers of two teaching modules can be interchanged. More complicated exchanges can also be envisaged, involving more than two teachers. However, searching for these exchanges with an aim to resolving clashes is a laborious process. In Fig. 6, an example of a multiple clash is shown.

By performing the following exchanges: $prof(S1) \leftrightarrows prof(S4)$, $group(S1) \leftrightarrows group(S2)$ and $room(S1) \leftrightarrows room(S3)$, the three clashes are resolved and the configuration shown in Fig. 7 is obtained.

Trials were carried out in real situations. We have worked with the manually made timetables for the academic year 1999–2000. The search for clashes amongst the 2,500 sessions took 3 seconds using a PC equipped with a Pentium III 500 Mz

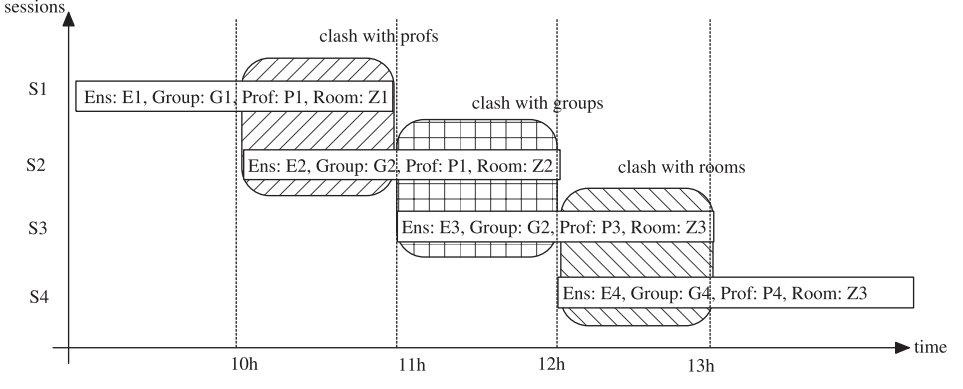

Fig. 6. Example of a multiple clash.

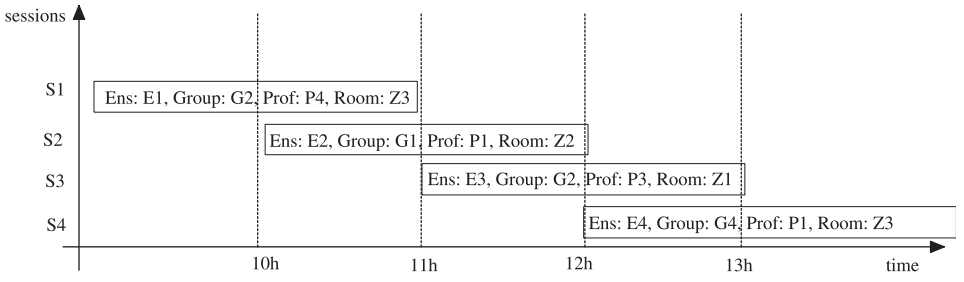

Fig. 7. The clashes have been resolved.

processor. This search revealed 350 group clashes, 50 teacher clashes and 30 room clashes.

It should be noted that this test was performed using real data from the manually-made timetables used for the university year, which should in theory have been clash-free! This mean that clashes was not detected during the conception phase of timetables: they have been treated during the year by teachers themselves.

Figure 8 shows how the clashes are displayed (group clashes in the case of the figure). There is a 9-column chart (the 9th column is hidden because of the size of the window). Each line represents a clash. For example, the first line shows a clash involving the $SMPC$ group on 04-02-2002. In fact there are two overlapping sessions. The first one, ALG_CR_DEUG, starts at 08.30 and lasts for 2 hours, while the second, ALG_CR_DEUG, starts at 09.00 and lasts for 1.30 hours. The overlap is therefore 1.30 hours. By using the $Voir$ button, the user sees the schedule for this group for the entire week which shows up these two sessions. The user can then decide how to correct the clash (move the sessions, eliminate them, etc.).

Figure 9 shows the view a user can have when he/she has decided to request a graphic view of a clash (the first clash in our example). The two clashing sessions are shown up by a color. It is necessary to modify (eliminate, move or reduce) the duration of one of the two sessions to resolve the clash.

The user can view the schedule for a set of resources, showing the teachers as well as the groups, rooms and equipment. This type of view is particularly well-suited to understanding why the choice of a place or move of session is impossible. For example, let us suppose that the graphic view shown concerns the timetable of a group $G1$ and that the user wishes to move a session S for this group. By clicking on the session, the user can choose the instruction move in the day. $\texttt{EDT}^{2002}$ calculates the list of slots corresponding to the possible moves. If the user does not find the slot he/she had been thinking of *a priori* on this list, the tool enables him/her to understand the reasons why by providing the occupation of each resource involved in the session.

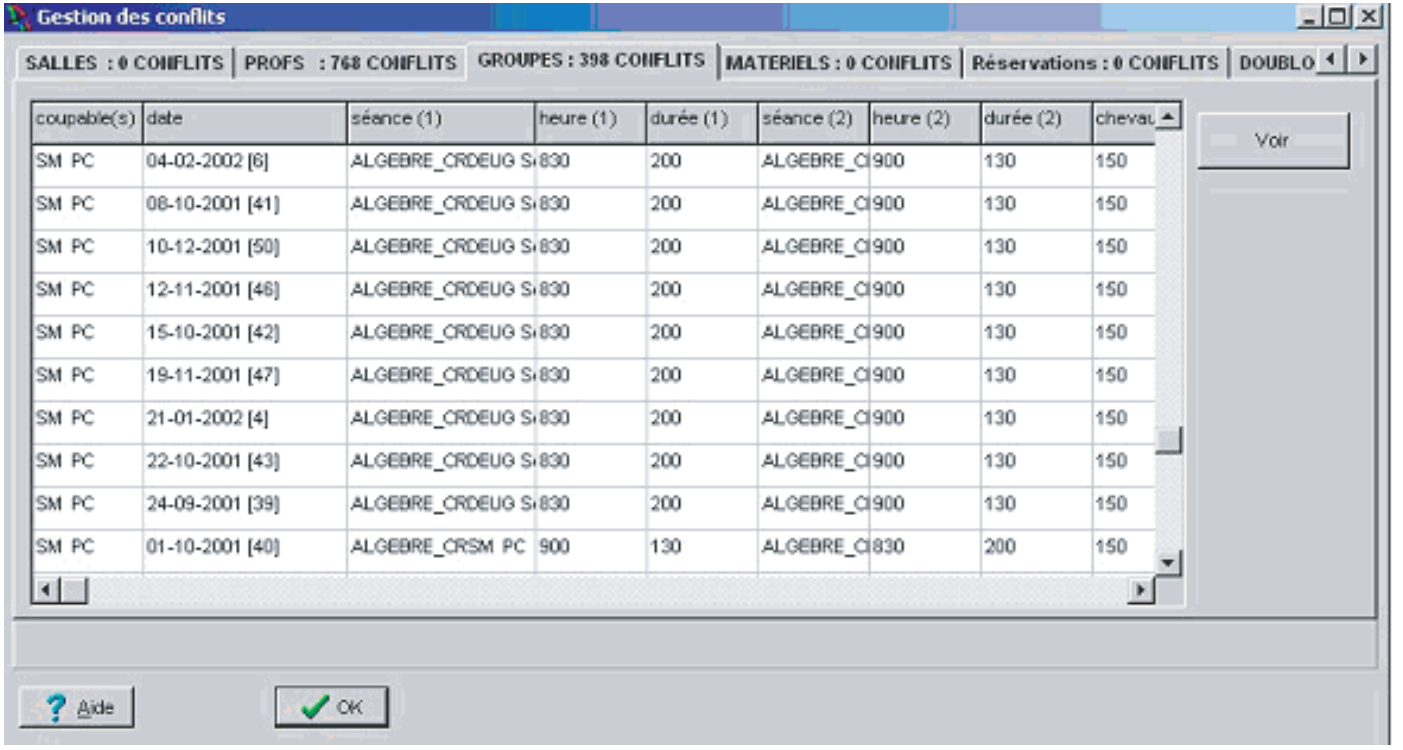

Fig. 8. Window displaying the clashes detected.

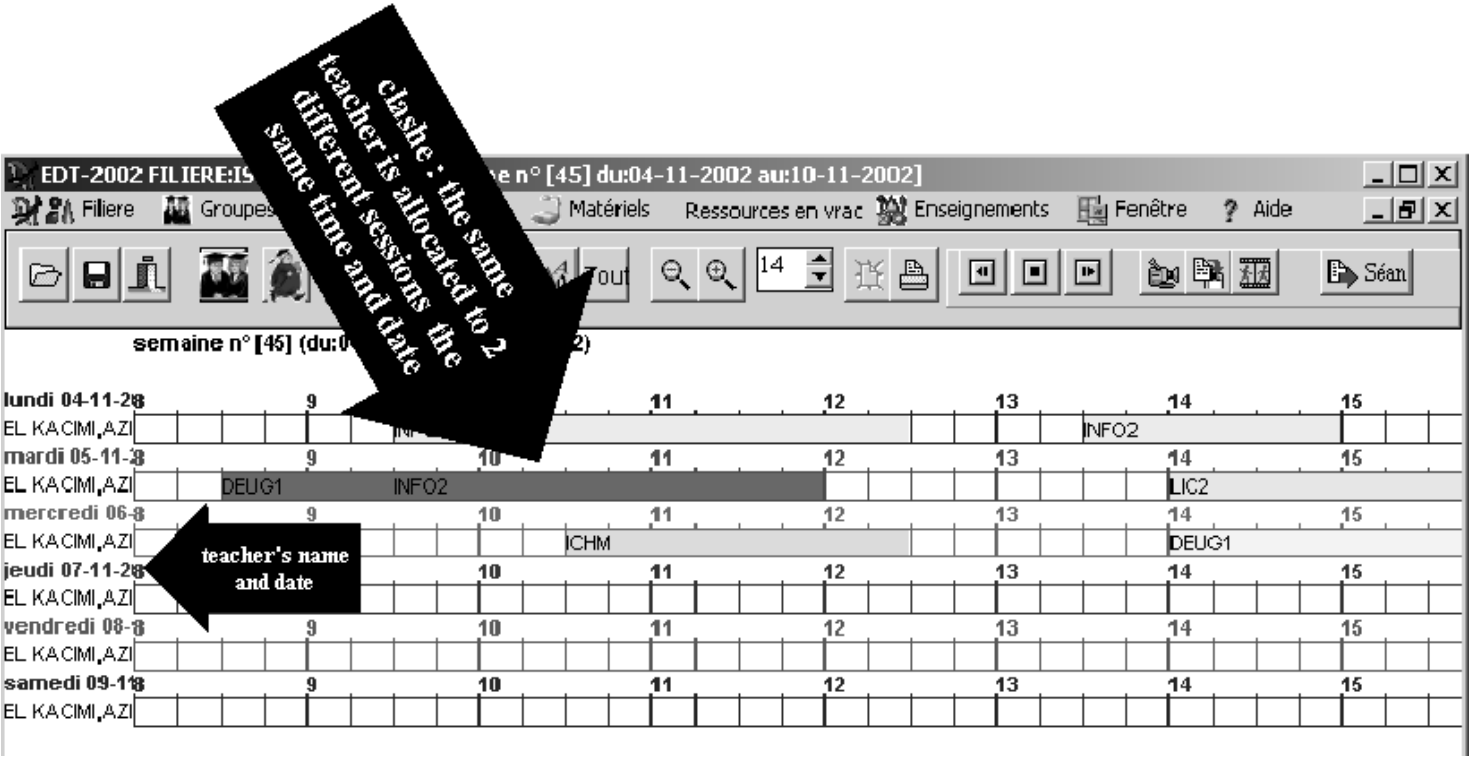

Fig. 9. Visualization of one of the clashes.

5.3. *Semi-automatic resolution of clashes*

The problem of resolving clashes can be tackled in a generic way, but experience has led us to the same remark as that made by Foulds and Johnson:[16] the clashes generally concern few sessions and can be resolved by backtracking a little way.

Clashes are detected according to each type of resource (teacher, group, room and equipment) and are classified according to different criteria. These criteria were defined on the basis of the experience of the various educational managers.

- The clash represents a **double**. Two sessions are a double (session duplication) if they are sessions in the same teaching module (so they have the same teachers and the same groups, but not necessarily the same equipment or the same rooms) or if they take place on the same day, at the same time and for the same duration. This problem is resolved by deleting one of the sessions. If the total volume of sessions of this teaching module is not equal to the total volume planned, the user will have to place the deleted session elsewhere.
- Sharing of a teaching module over several groups. For example, conferences are organized for two courses represented by $G1$ and $G2$. The managers of these courses create their timetables. Then, when all the timetables are collated for the allocation of rooms and detection of clashes, the conferences are detected as being clashes because they share the same teacher, on the same date, at the same time and for the same duration. Experience shows that these cases of teaching modules shared over several courses is done with small groups and according to the level (1st, 2nd or 3rd cycle). The problem is solved by deleting one of the two teaching modules and adding the teaching module groups from the deleted module to the groups of the remaining module.
- Resolution of clashes by giving preference to moving sessions involving small groups and short sessions (in duration).

6. Implementation

The work presented in this article required the creation of an interactive tool which went beyond the prototype stage. In order to perform test campaigns with people in work situations, it was important for the tool not to appear to be a laboratory product requiring heavy means, but as a truly usable and user-friendly tool.

The EDT^{2002} tool was completely developed using DELPHI (version 5). This language makes it possible to create software for middle-range PCs, thus fulfilling our initial intention to work on PCs with modest performance capacities.

A considerable effort was made concerning the definition of the objects in order to make it easier to adapt the tool not only to other similar contexts (universities, secondary schools), but also to contexts in which the activities to be scheduled depend on different shared resources (personnel planning) and in which an "interactive decision support" approach is preferable to an entirely automated approach.

It is not possible to give a detailed description of all the classes defined in this paper. In the next section, we will therefore concentrate solely on the classes relevant to the problem of creating timetables according to a decision support approach. Amongst these classes, we can distinguish those which are linked to the modelling of the problem and those which concern the interfaces between the models and the users.

6.1. *Definition of the objects linked to the modelling of the timetable problem*

The abstract class *TAbstract* makes it possible to define characteristics common to the classes *TFiliere* and *TResource* which model the notion of branch and resource. The attributes of these classes are given in the previous paragraphs of this article. The *TResource* class is generic in the timetable problem. All the specific resource classes stem from the *TResource* class: we find the classes *TProf*, *TGroup*, *TRoom*, *TMaterial* and *TStudent* (teacher, group, room, equipment and student) which model the classes dealt with by EDT^{2002} . All the objects have an ID attribute which has a unique value. Each object in the *TResource* class has the attribute *daughterResources* and *motherResources* used to memorize the hierarchical organization of resources. Only the direct mother or daughter resources are memorized. The set of all the mother and/or daughter resources of a specific resource are calculated recursively using these two attributes.

In the tool, we have treated the students as groups limited to one single person. The class *TStudent* has therefore been defined as being a sub-class of *TGroup*. The advantage of this approach enables each student to consult his/her timetable without having to remember which groups he/she belongs to. This choice proved to be very pertinent in reality, even though it is questionable from a modelling point of view.

It should be emphasized that for the use of EDT^{2002} at the ISTV only the organization of the groups and the equipment require the *daughterResources* and *motherResources* attributes (there is no similar organization for the teachers or the rooms). However, it would be perfectly possible to organize the teachers into groups according to their subjects, for example.

The classes *TTeachingModule*, *TSession* and *TSubject* (teaching module, session and subject) are also sub-classes of *TAbstract*. These classes make it possible to describe the activities and are therefore generic for the modelling of the timetable problem.

Figure 10 shows the organization of the previous classes.

In parallel to the definition of the classes which make it possible to model individual entities, we have also defined other classes to model sets of entities. The organization of these new classes is shown in Fig. 11.

The modelling of the resource types which were presented during the paragraphs on treatment of equipment made it necessary to define specific classes (Fig. 12).

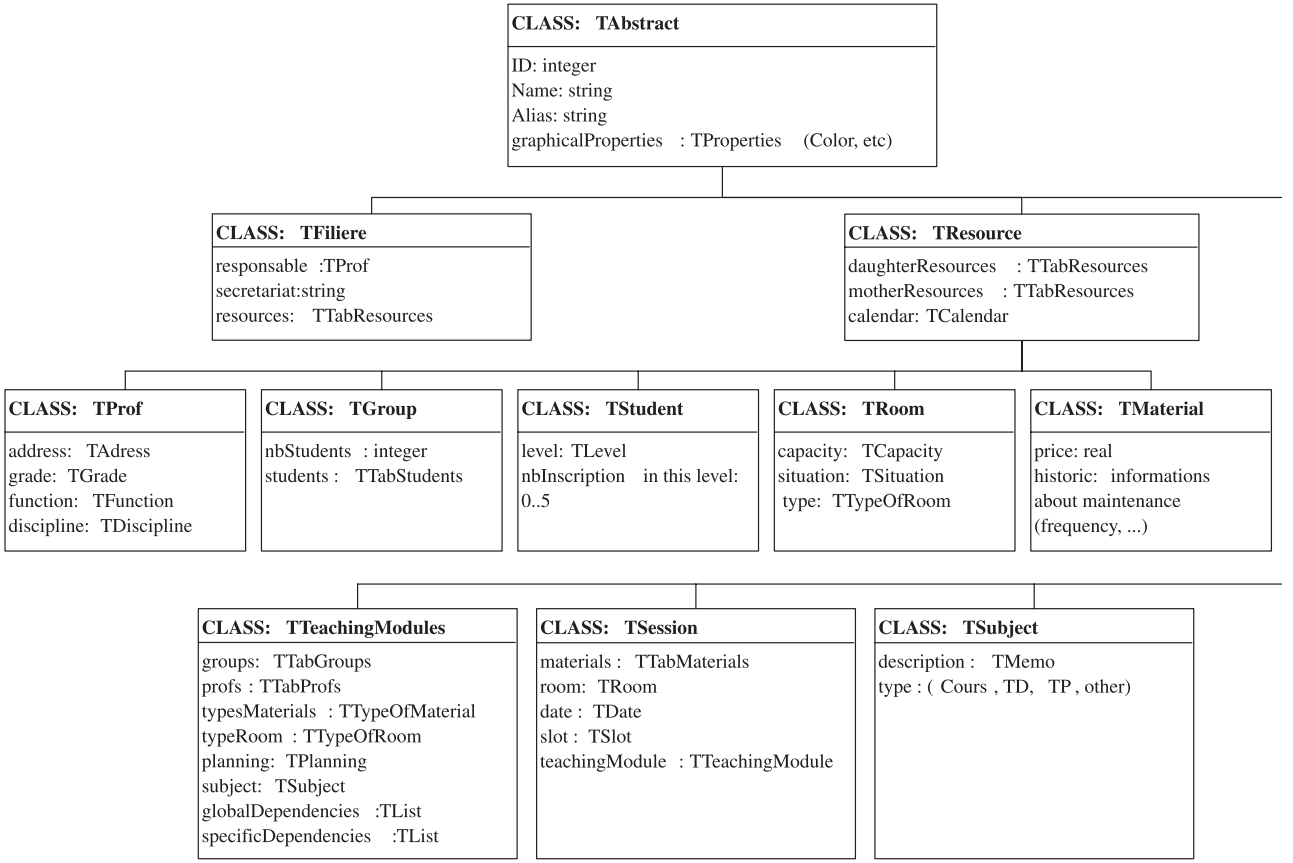

Fig. 10. Organization of basic classes (individual entities).

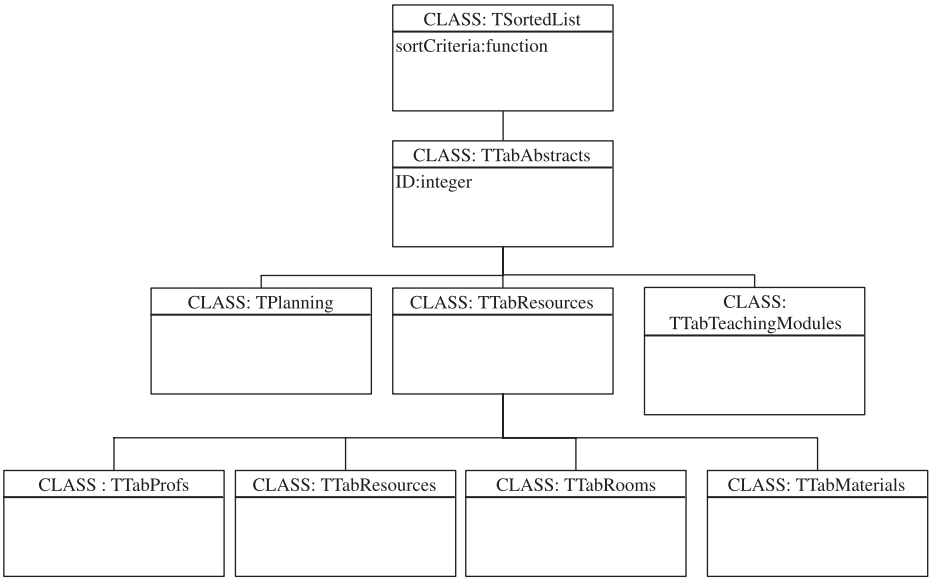

Fig. 11. Organization of basic classes (groups of entities).

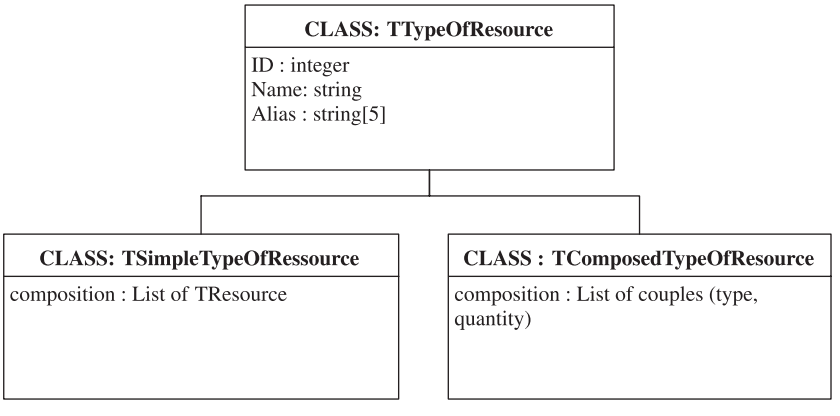

Fig. 12. Classes defined for the modelling of resource types.

The *TSimpleTypeOfResource* class (simple type) makes it possible to attribute a type to each resource. The *TComposedTypeResource* class (complex types) enables us to define more complex types using the objects in the class *TSimpleTypeOfResource* or from its own class. These two classes complete the *TResources* and *TTabResources* classes. The latter are necessary for the planning phase whereas the former are vital for the phase in which missing resources are allocated.

In addition to all these classes necessary for the modelling of the timetable problem, further classes were defined to manipulate the time aspect. These classes

are called *TDate*, *TTabDates*, *TSlot*, *TTabSlots* and *TCalendar* (date, slot and calendar). They are not described in detail here.

7. Conclusion and Future Work

In this article, the problem of timetable management has been looked at from a viewpoint centred on the user, and not, as in many other research projects, centred on the problem (or on the algorithms available). This led us to design the EDT^{2002} decision support tool which helps the user when faced with solving a problem. The tool has been tested successfully at the University of Valenciennes and Hainaut–Cambrésis in large scale real situations.

EDT^{2002} is intended for different user profiles, going from the designers of timetables to users who have varied degrees of computing skills. It makes it possible to manage resources organized into a hierarchy (groups, teachers, rooms and equipment). It guarantees data coherence in a dynamic way. When clashing situations appear, various resolution approaches are suggested.

The aim was to design a generic interactive decision support tool for timetable problems on the basis of a real case study which was truly representative of the problem. An object-oriented approach was chosen and the generic classes were defined.

Work is now continuing in two directions according to the suggestions made in Ref. 5 concerning the evolution of decision support systems, especially in the present context with the considerable expansion of internet based activities.

In the first direction, the aim will be to help the user to express constraints and to take these constraints into account in an interactive manner. For this, we intend to revise the work performed on constraint programming to provide support in the design and generating of timetables. In this field, most existing work deals with automated generation. On the other hand, very few research projects concentrate on support in the description of timetables.[9] We feel that an interesting project would be to combine the automated and interactive generation of timetables. This direction was explored successfully in Refs. 1 and 17 which deal with other problems based on scheduling. For example, we envisage performing an automatic allocation of rooms after the timetables have been created interactively with the users (thus making it easier to take into account the pedagogical constraints which are difficult to express and formulate). A project similar to Ref. 10 based on approaches from the constraint programming field has already been started on this subject.

In the second direction, the problem of timetable management and generation will be tackled from the collective software angle or using multi-agent approaches.[15] For this, it is necessary to design a distributed architecture as a base for decision support tools for cooperative timetable creation by various agents. This type of perspective is also studied in Ref. 23. We envisage an abstract architecture able to support cooperative reasoning involving several agents responsible for developing one or several timetables. Two main situations can be seen.

In the first situation, each agent is responsible for building one timetable. In this situation, each agent begins by generating a specific clash-free timetable. Modifications are then negotiated with other agents when the resources needed are in clash. Several negotiating levels can be distinguished. The first level involves agent-to-agent negotiations. The second level corresponds to negotiations agent-to-agent within the same interest group. The final third level concerns negotiation between groups of agents.

In the second situation, the agents are specialized in specific fields. For example, some agents are specialized in the allocation of rooms, others are capable of judging the pedagogical quality of a timetable, yet others are budget specialists and will push towards sharing resources between several courses, etc. Unlike the previous situation, the agents do not generate a timetable completely, but work on various timetables according to their speciality.

Acknowledgements

Authors would like to thank Prof. Bertrand David, Ruben Gonzalez Rubio and Prof. Philippe Trigano for their help in improving the quality of this paper.

References

1. S. Abdennadher and H. Schlenker, INTERDIP, an interactive constraint based nurse scheduler, in *Proceedings PACLP'99*, London, 1999.
2. J. Allen, Time and time again: The many ways to represent time, *International Journal of Intelligent Systems* **6** (1991) 341–355.
3. R. Bartak and H. Rudova, Integrated modelling for planning, scheduling, and timetabling problems, in *Proceedings of PLANSIG 2001*, Edinburgh, UK, 2001.
4. J. M. C. Bastien and D. L. Scapin, Évaluation des systèmes d'information et Critères Ergonomiques, (dir.) C. Kolski, Environnements évolués et évaluation de l'IHM, Interaction homme-machine pour les SI 2, Hermes, Paris, 53–80, 2001 (in French).
5. M. Beynon, S. Rasmequan and S. Russ, A new paradigm for computer-based decision support, *Decision Support Systems* **932** (to appear, 2002).
6. P. Boizumault, Y. Delon and L. Peredy, Constraint logic programming for examination timetabling, *Journal of Logic Programming* **26-2** (1996) 217–233.
7. P. Boufflet, Emplois du temps dans un environnement fortement contraint: Exemple de l'UTC, PhD Thesis, Compiègne University (1992) (in French).
8. M. W. Carter, G. Laporte and J. W. Chinneck, A general examination scheduling system, *Interfaces* **24**(3) (1994) 109–120.
9. T. B. Cooper and J. Kingston, A program for constructing high school timetables, Technical Report n° 496, University of Sydney, Australia, 1995.
10. S. Deris, S. Omatu and H. Ohta, Timetable planning using the constraint-based reasoning, *Computers and Operations Research* **27** (2000) 819–840.
11. M. Dimopoulo and P. Miliotis, Implementation of a university course an examination timetabling system, *European Journal of Operational Research* **130** (2001) 202–213.
12. A. Dix, J. Finlay, Abowd and R. Beale, *Human-Computer Interaction* (Prentice Hall, Europe, 1998).
13. A. Drouin, A. Valentin and J. Vanderdonckt, Les apports de l'ergonomie à l'analyse et la conception de systèmes d'information, (dir.) C. Kolski, Analyse et Conception

de l'IHM, Interaction homme-machine pour les SI 1, Hermes, Paris, 51–83, 2001 (in French).
14. R. W. Eglese and G. K. Rand, Conference seminar timetabling, *Journal of the Operational Research Society* **38** (1987) 591–598.
15. J. Ferber, *Les systèmes Multi-agents: Vers une Intelligence Collective* (Interéditions, Paris, 1995) (in French).
16. L. R. Foulds and D. G. Johnson, SlotManager: A microcomputer-based decision support system for university timetabling, *Decision Support Systems* **27** (2000) 367–381.
17. H. J. Goltz and D. Matzke, Combined interactive and automatic timetabling, in *Proceedings PACLP'99*, London, 1999.
18. H. J. Goltz, On methods of constraint-based timetabling, in *Proceedings PACLP'00*, Manchester, 2000.
19. D. G. Johnson, Timetabling university examinations, *Journal of the Operational Research Society* **41** (1990) 39–47.
20. D. Jouglet, S. Piechowiak and F. Vanderhaegen, Cooperative diagnosis tool including a multi-point of view modelling: The case of France Telecom's phone system network troubleshooting, in *Proceedings Analysis, Design and Evaluation of Man-Machine Systems*, Kyoto, Japan, 1998.
21. C. Kolski, *Interfaces Homme-Machine* (Editions Hermès, Paris, 1997) (in French).
22. L. M. Kwork, S. C. Kong and Y. Y. Kam, Timetabling in Hong Kong secondary schools, *Computers Education, Elsevier Science* **28:3** (1997) 173–183.
23. B. M. Lawal, D. Gilbert and A. Letichevsky, A web course scheduler in constraint logic programming: Interactive computing with constraints, *Workshop on Constraint Reasoning on the Internet, 3rd International Conference on Principles and Practice of Constraint Programming (CP'97)*, 1997.
24. R. E. Mayer, From novice to expert, in *Handbook of Human-Computer Interaction*, eds. M. G. Helander, T. K. Landauer and P. V. Prabhu (North Holland, 1997) 781–796.
25. T. Muller and R. Bartak, Interactive timetabling, in *Proceedings of ERCIM Workshop on Constraints*, Prague, 2001.
26. J. Nielsen, *Usability Engineering* (Academic Press, Boston, 1993).
27. D. A. Norman, Cognitive engineering, *User Centered System Design: New Perspectives on Human Computer Interaction*, eds. D. A. Norman and S. W. Draper (Hillsdale, Erlbaum, NJ, 1986).
28. S. Piechowiak, D. Jouglet and F. Vanderhaegen, A multi-point of view for phone network system diagnosis, in *Proceedings ICSSSE'98*, Bengjing, China, 1998.
29. S. Piechowiak and C. Kolski, Analyse et conception d'un outil interactif d'aide à la gestion des emplois du temps basé sur les points de vue, *Journal of Human-Computer Interaction*, in press (in French).
30. A. Schaerf, A survey of automated timetabling. Technical Report n° CS-R9567, Centrum voor Wiskunde en Informatica, 1995.
31. R. H. Sprague and E. D. Carlson, *Building Effective Decision Systems* (Prentice-Hall, Englewood Cliffs, NJ, 1982).
32. E. Tsang, *Foundations of Constraint Satisfaction* (Academic Press, Computation in Cognitive Science, 1993).
33. M. Yoshikawa, K. Kaneko, Y. Nomura and M. Watanabe, A constraint-based approach to high-school timetabling problems: A case study, in *Proceedings of the 1995 International Joint Conference on AI (IJCAI'95)*, Montreal, Quebec, Canada, 1995.