Semantic Components for Timetabling

Nele Custers, Patrick De Causmaecker, Peter Demeester
and Greet Vanden Berghe

KaHo Sint-Lieven, Information Technology
Gebroeders Desmetstraat 1, 9000 Gent, Belgium
{nele.custers, patrick.decausmaecker, peter.demeester,
greet.vandenberghe}@kahosl.be

Abstract. Automated timetabling is a research domain that has occupied many researchers over the last 50 years. Several algorithms have proven to be applicable to timetabling but they are nearly all designed to address specific problems. The framework presented in this paper is a step towards a generic semi-automatic timetabling tool. The basis of the framework is an ontology for timetabling that we designed after research on different types of timetabling problems.
We developed a calculation component that can deal with the search space based on problem characteristics passed on by the semantic components. A first step towards solving general problems consists of mapping their data representation to the ontology. In the second step, the tool assists in determining the constraints and objectives of the problem. The semantic components have three sources of information: meta data about the database, domain knowledge about timetabling problems and external, non-domain specific knowledge.

Keywords: semantics, ontology, timetabling

1 Introduction

Timetabling problems have been under intensive investigation [4,5,6,7,8]. Resulting from this research, there are numerous approaches generating good quality solutions. All these applications are based on algorithms that arrive at satisfactory solutions within a specific model. For each specific situation, the model is redesigned [12]. This lack of software reuse hinders fast and reliable development of timetabling tools.

While studying different timetabling problems, it is possible to identify a common set of characteristics. In Fig. 1, we have grouped some key characteristics of four different timetabling (or scheduling) problems with varying terminology such as lecture, game, qualification, operation, etc. In [25], Wren defines timetabling as "the allocation, subject to constraints, of given resources to objects being placed in space-time, in such a way as to satisfy as nearly as possible a set of desirable objectives." Hard constraints are those that must be satisfied at all time. Soft constraints can be violated but that will affect the quality of the solution. Reducing the number of violated soft constraints is a frequently occuring objective in timetabling. Often, the objective of timetabling problems is expressed in terms of a mathematical function, called the objective function, reflecting a degree of constraint violation.

We developed a framework for timetabling applications with a central domain ontology [10], which in this case is timetabling. Ontologies enable the sharing of information between researchers in a specific domain. They contain at least some machine-interpretable definitions of domain concepts and their relations. Our timetabling ontology is based on the general OZONE [22] scheduling ontology and is expressed in the DAML+OIL [9] ontology language. This ontology language is based on web standards such as XML [3] and RDF [15]. RDF provides a format to describe data using XML as a serialisation syntax. All these languages and standards fit in the development of the Semantic Web [1].
The developed timetabling ontology allows the application of one single approach to solve a multitude of different problems. The ontology acts as an intermediate language between the data layer - in our case this will be a data base containing specific information about the timetabling problem at hand - and the calculation component. Supposing that the problem data is stored in a problem specific database, all that is needed is a mapping - some kind of semantic translation - of the database schema to the ontology. The mapping can be performed semi-automatically - user assistance is still needed - by the presented semantic mapping component. We do not believe that fully automatic mapping is possibe: input will be needed from a domain expert during the mapping process. The calculation component, which is aware of the ontology, can compute a solution with the data extracted from the data layer. Since the calculation component has to be able to solve different timetabling problems, it will certainly not outperform timetabling applications that were built with a specific problem in mind. As in nearly all pratical applications we do not want to produce the optimal solution but we will rather assist in making decisions.
In Fig. 2 a schematic overview of the tool is depicted. All depicted parts will be discussed in the next sections.

The focus of this paper will be on the semantic part of the developed application and less on the calculation component. In Section 2, we introduce other recent contributions to semantic mapping tools. The semantic mapping component for timetabling is discussed in Section 3. In Section 4, we describe the user support for defining the timetabling problem. Section 5 summarises the results and indicates interesting challenges for further work.

2 Related Work

The ontology mapping tool described by Prasad et al. [21] uses explicit information. It must be provided by 'exemplars' that describe the meaning of the concepts in both ontologies. By using a text classifier, a model is built for each ontology. Afterwards, the models of the ontologies are compared with each other and the concepts with the highest similarity scores are considered for mapping.

	School Timetabling	**Sports Timetabling**	**Nurse Rostering**	**Job Shop Scheduling**
Scheduling Object	Session (Lecture)	Game	Shift (duty assignment)	Operation (of a job)
Problem Dimension	Time Rooms	Sports-day	Time People	Start time of operation
Attributes of Scheduling Object	Lecturer (group) Student (group)	Home team Visitor team	Shift type Qualification	Machine Job
Examples of Hard Constraints	Lecturer @ place/time <= 1 Student @ place/time <= 1 Full curriculum No Sessions on holidays	Double round robin (A at B and B at A) No repeaters (A at B, followed immediately by B at A) Not more than three consecutive home or road games for any team	Nurse Qualifications Coverage Constraints	Tool(s) Precedence constraints Capacity Resources Stock
Examples of Soft Constraints	Capacity constraints Room facilities Precedence lectures Lecturer preferences Student preferences	Play home match if local bank holiday Broadcasting preferences	Contract constraints: overtime, workload, consecutive shifts, ... Personal preferences Patterns	Due date Machine constraints
Goal: minimise	Violated soft constraints	Distance travelled	Violated soft constraints	Tardiness Processing time

Fig. 1. Characteristics of sample timetabling problems

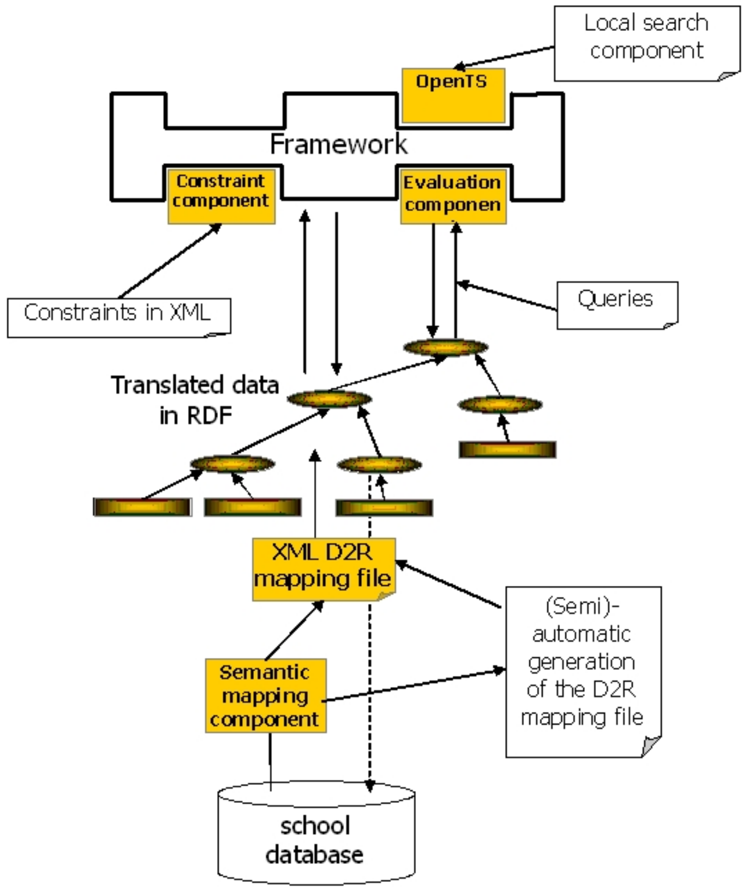

Fig. 2. Schematic overview of the tool

Two algorithms (heuristic and Bayesian approach) are developed to finalise the mappings.

The ontology mapping tool GLUE [11] uses of a multi-strategy learning approach with a set of learners of which the predictions are combined. Domain constraints and general heuristics improve the accuracy of the matching.

Anchor-PROMPT [19] and PROMPT [20] are tools for merging ontologies. The activity of mapping concepts of the ontologies is an important step in the merging process. Our tool has a similar aim as it tries to map the concepts in a database schema to an ontology. It differs in the sense that one ontology is considered fixed and will probably only cover a subset of the concepts in the database. PROMPT explores the ontologies to locate candidate terms for merging. It uses both syntactic and semantic information and feedback from the user. Anchor-PROMPT first searches pairs of related terms in different ontologies (‘anchors’), which are identified by the user or automatically generated by the system. Starting from these anchors, Anchor-PROMPT searches a new pair of terms on the path between anchors.

KAON Reverse [23] is the tool that approaches our needs for a semantic mapping tool best. It allows to export data from a database to an ontology. As a prototype, KAON Reverse’s functionality is too basic to fulfil the requirements of the timetabling mapping tool.

Missikof et al. [18] developed a software environment with the OntoLearn tool as a core. It can build and valuate domain ontologies. The software environment acquires new domain concepts by exploring available documents and related Web sites. WordNet (see Section 3) is one of the resources used for the semantic interpretation of the corpus. Missikof et al. state that capturing kindship relations is clearly important for an ontology based Web application.

3 Semantic Mapping Component

The purpose of the semantic mapping component is to ‘map’ the specific problem data of a relational database to our timetabling ontology. The structure and content of the database will always be problem dependent, but the target ontology is always the same. In Fig. 3, the timetabling ontology is represented in an ontology editor called OilEd [27]. In [26], the complete ontology can be found. This ontology is rather general since different problem specific databases have to be mapped onto it. Fig. 4 presents the user interface of the mapping component.

Each relational database consists of relations (tables) with attributes. There are two types of special attributes: primary keys and foreign keys. One condition for the mapping component is that there are no composite primary keys in the relations. The second part of the semantic mapping component consists of the ontology for timetabling problems and is shown in the right part of the user interface (see Fig. 4). The ontology includes classes with properties. There are two special types of properties: identifier and reference properties. The first step in the process consists of mapping a relation from the database to a class in the ontology. Next, the attributes of the relation are mapped to the corresponding properties of the class. The correspondence between the database and the ontology terminology is illustrated in Fig. 5.

The semantic mapping component uses three brands of information source. First of all, meta data about the database offers useful knowledge. We mark the primary key of each relation and annotate each foreign key with the relation whose candidate key it matches. Domain knowledge about timetabling problems is a second source of information. Timetabling concerns resources that need to be scheduled in a time frame. Apart from these problem specific information sources, WordNet [28] adds external information to the mapping component. WordNet is the result of a research project at Princeton University that has attempted to model the lexical knowledge of a native English speaker. Information in WordNet is organised around logical groupings called synsets. Each synset consists of a

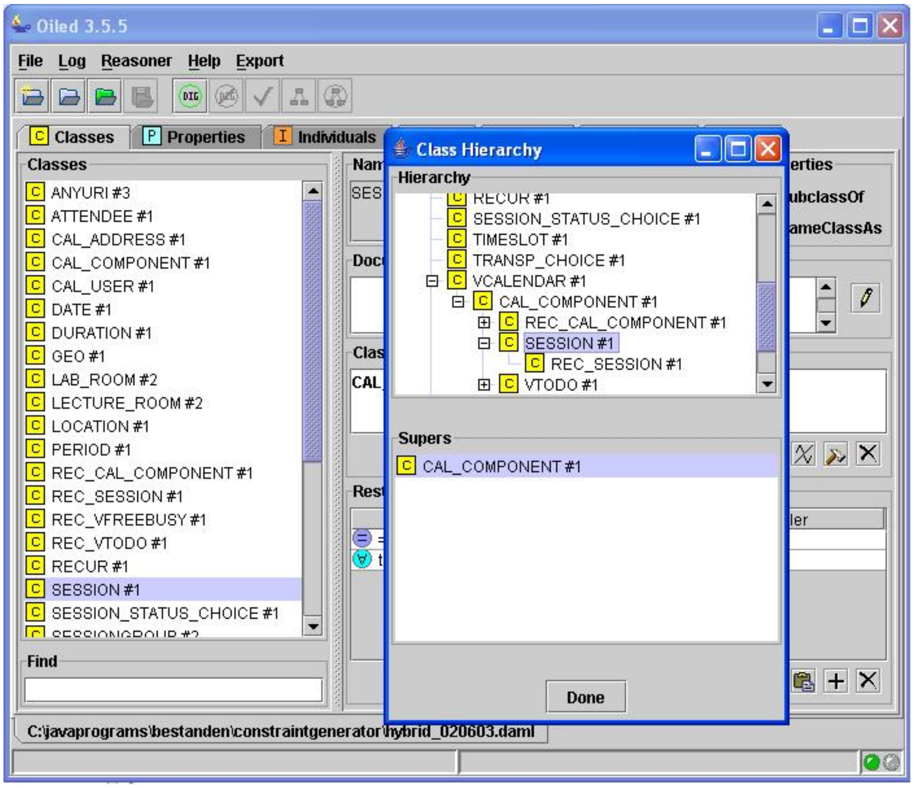

Fig. 3. The timetabling ontology presented in OilEd

list of synonymous word forms and semantic pointers that describe relationships between the current synset and other synsets. These semantic pointers can be of a number of different types including: 'Hyponym/Hypernym (Is-a, Has-a)' and 'Meronym/Holonym (Part-of/Has-part)'.

Fig. 6 describes the procedure for mapping the database to the ontology. In the first step, the central timetabling object is identified by the mapping process. This object will be liable to the calculation component and must be mapped to the class 'Session' in the ontology. A first possible way to recognise this central timetabling object is its number of foreign keys. The central timetabling object of the database will often be the relation with the largest number of foreign keys because it involves a lot of resources. A lecture, for example, is assigned to a teacher, students, a room, etc. The second rule for finding the central timetabling object indicates that it is a kind of activity or event. This characteristic can easily be checked using WordNet. If this still does not identify the central timetabling object user assistance is needed.

Once we managed to find the relation in the database that represents the central timetabling object, we can start mapping its attributes. A first rule for

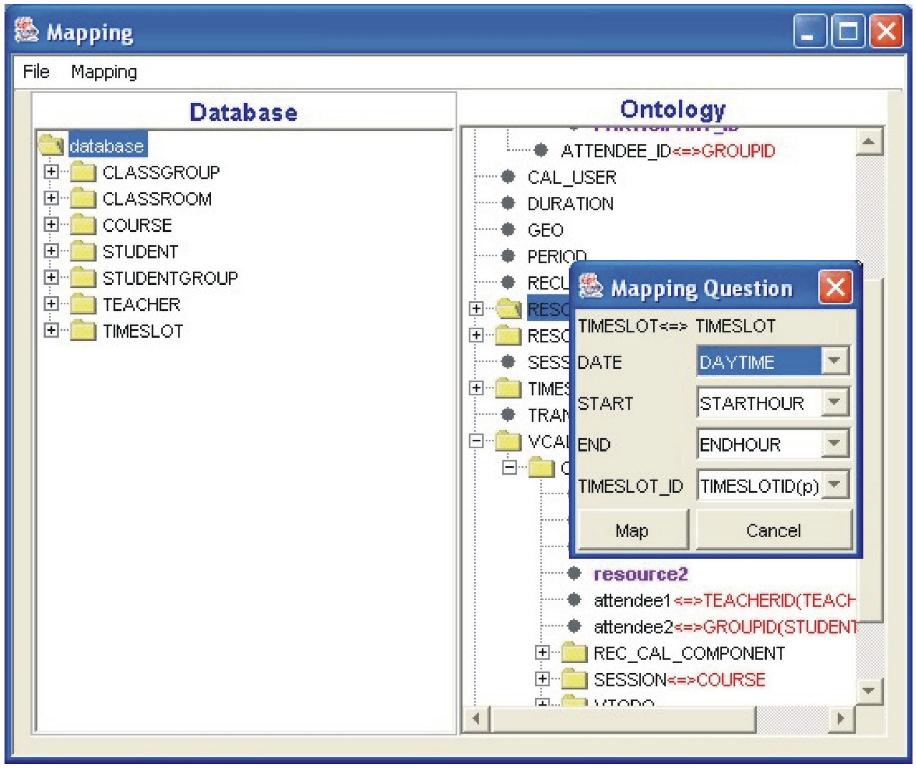

Fig. 4. GUI of the mapping component

mapping attributes is that the identifier of the class in the ontology will be mapped to the primary key of the corresponding relation. For example, we managed to recognise ‘shift’ (this is a task to be carried out within a specified time period) in nurse rostering as the central scheduling object. Therefore, ‘shift’ will be mapped to the ontology class ‘Session’. The ontology class ‘Session’ contains an identifier ‘SessionID’. We will map the primary key of the relation ‘shift’ to this identifier ‘SessionID’. Other attributes of the relation are mapped to the most similar property of the ontology. We apply a slightly altered version of the recursive algorithm for computing string similarity [13]. Attributes can also be mapped to properties that are not similar, by using knowledge about the English language. For example, ‘room’ can be mapped to ‘location’ because ‘room’ is a kind of ‘location’ according to WordNet. Similarly, ‘nurse’ is a possible dimension of the nurse rostering search space because it is a kind of ‘professional’ and it is an attribute of the central timetabling object.

Once a foreign key of the relation can be mapped to a reference property of the ontology, we can also map the corresponding relation to the referenced class. Subsequently, we can map attributes of the relation to properties of the class, etc.

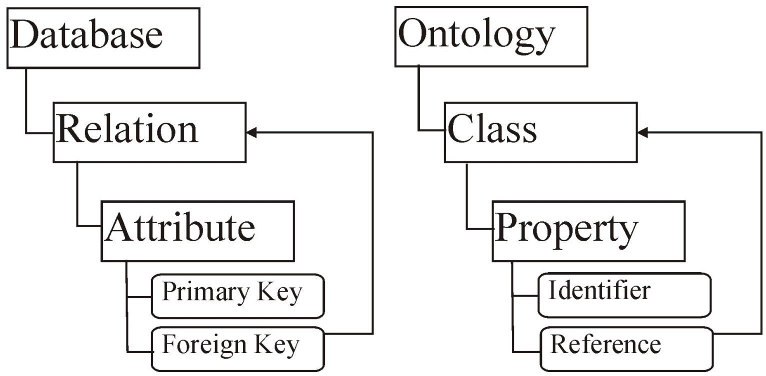

Fig. 5. Terminology

The Semantic Mapping Component also determines the dimensions of the solution space. In school timetabling, for example, sessions can be moved in time and space (rooms). In the sport timetabling problem that we studied [24], one single dimension (time) suffices (see also Fig. 1).

The result of the mapping is a semi-automatically generated XML file that is in the appropriate format to be used by the D2R mapping tool [2]. This last tool allows the translation of data from a relational database to an RDF file. Normally, the D2R XML file that describes how to map data from the database to the RDF file, is written manually, but our mapping tool is now able to semi-automatically construct this ‘translation’ file. The resulting RDF data file, containing the problem specific data in terms of the timetabling ontology, will be used in the computational part of the framework and in the graphical semantic constraint generator.

4 Timetabling characteristics

4.1 Constraints

Timetabling problems are not completely characterised by the RDF data file alone. Some additional semantic components are required.

We developed a semantic tool - the graphical semantic constraint generator (GSCG) - to assists in defining the timetabling problem with concepts from the user’s domain. The GSCG (Fig. 7) builds on the results of the mapping component and enables the user to specify the constraints. It stores the constraint description in separate XML files. We opt to save the constraints in pure XML, since we experienced that it was hard to express the constraints in DAML+OIL. These difficulties are also indicated in [16,17].

Constraints are expressed as follows:

[|MAX|MIN] number of concept1 PER concept2 is [LESS THAN|EQUAL|etc] concept3

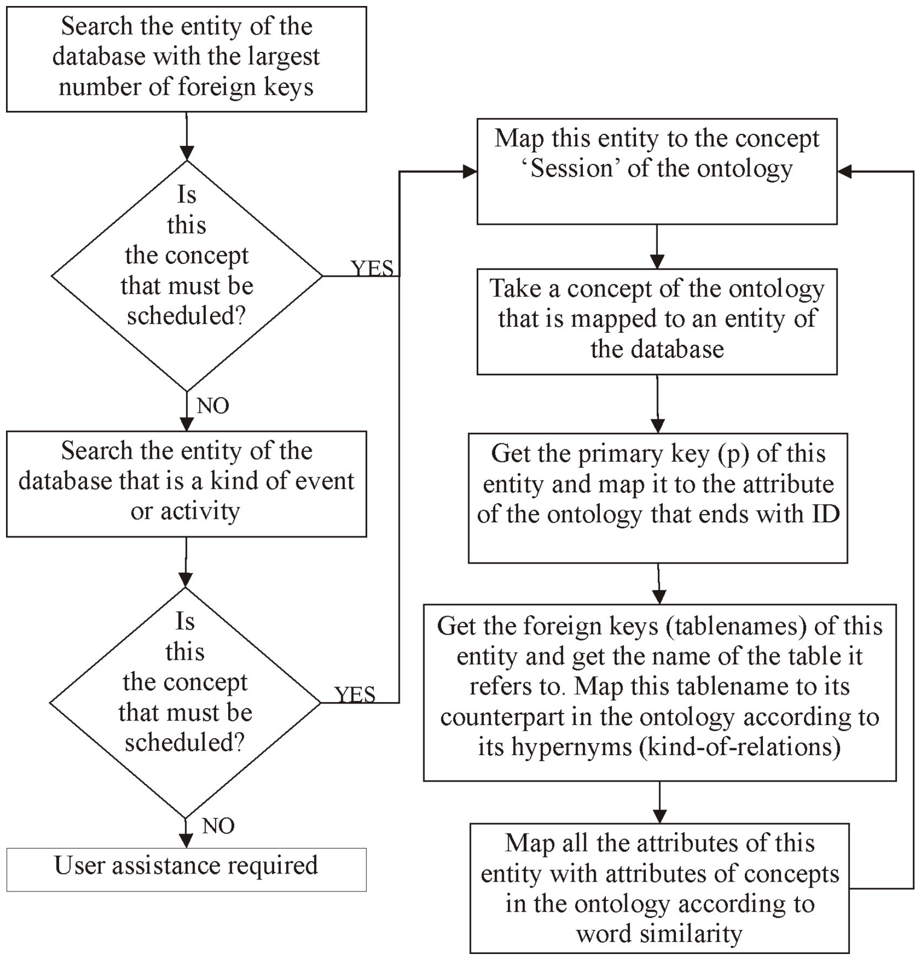

Fig. 6. Mapping rules

Remark that the user can select values of the first variable out of `empty`, `MAX`(imum) and `MIN`(imum).
For example,

$$\texttt{(number of sessions) per (timeslot and location)} \leq 1,$$

means that there cannot be more than one session per timeslot in a location (room). Both hard and soft constraints can be expressed in the same form.

In order to define comprehensible constraints, we sometimes need information that is not available in the ontology. 'Weekend' is an example of a concept that is commonly used in timetabling constraints but it is not available in the timetabling ontology nor in the database. The XML code in Fig. 8 clearly demon-

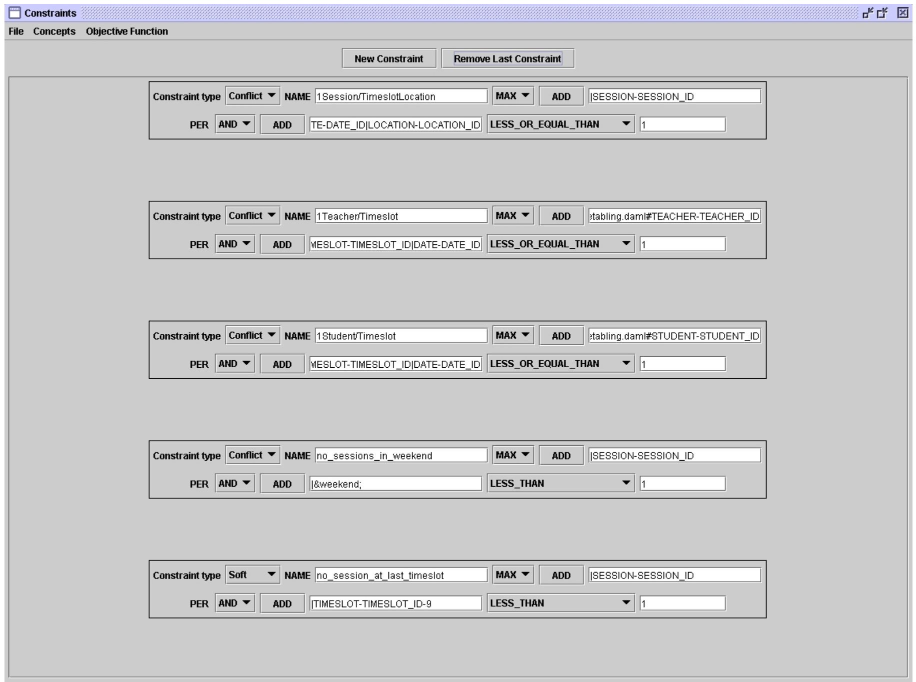

Fig. 7. GUI for constructing constraints

strates that the existing (primitive) concept 'Date' is used to define the concept 'Weekend'. We use the translated primary key of the date concept (`DATE_ID`) to define the 'Weekend' concept. In the example, we consider a period of 3 weeks (21 days) consisting of 3 weekends. Every weekend is defined as consisting of a Saturday and a Sunday. Of course, it is also possible to define a 'Weekend' as starting on Friday and ending on Monday.

4.2 Objective Function

Once the mapping has been completed and the constraints are known, the timetabling problem is essentially defined. The calculation component, however, still requires an objective function that deals with constraint violations. A final component (Fig. 9) supports the user in the definition of this function. It can be seen as a tuning instrument for the application, allowing the user to express the relative importance of the constraints.

4.3 Calculation component

The generic calculation component is built on the OpenTS [14] framework. This is a framework for tabu search that is implemented in Java. OpenTS has all the

```xml
<concept ID="1">
   <name>weekend</name>
   <def logical_operator="AND">
      <primitive>
         <DATE>
            <DATE_ID>6</DATE_ID>
         </DATE>
      </primitive>
      <primitive>
         <DATE>
            <DATE_ID>7</DATE_ID>
         </DATE>
      </primitive>
      <primitive>
         <DATE>
            <DATE_ID>13</DATE_ID>
         </DATE>
      </primitive>
      <primitive>
         <DATE>
            <DATE_ID>14</DATE_ID>
         </DATE>
      </primitive>
      <primitive>
         <DATE>
            <DATE_ID>20</DATE_ID>
         </DATE>
      </primitive>
      <primitive>
         <DATE>
            <DATE_ID>21</DATE_ID>
         </DATE>
      </primitive>
   </def>
</concept>
```

Fig. 8. Example of XML file for expressing the 'Weekend' concept

ingredients for tabu search already built in. All that is required from the user is to implement the problem specific interfaces. In these problem specific classes, the user describes a structure for the solution, a tabu list, an objective function, the allowed moves and a move manager executing these moves. To assure that a wide range of timetabling problems can be solved, our implementation classes are as generic as possible. As a consequence, our algorithms are less efficient in finding optimal solutions for specific problems. As a pay-off, we obtained a generic tool for decision support that can be used in a multitude of application domains.

5 Conclusion

It is the aim of this research to apply specific timetabling knowledge to solve any kind of timetabling problem within a generic framework. We have achieved to support experienced planners in integrating their problem (represented in a database) into the timetabling framework.

Given the following information:

- the ontology,

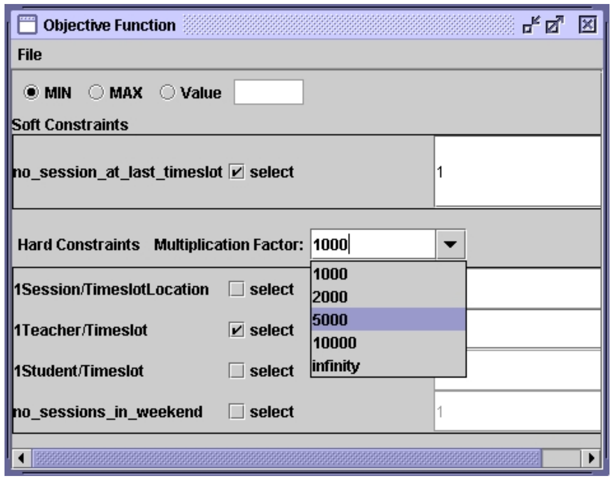

Fig. 9. GUI for constructing the objective function

- the mapped data of the problem (Section 3),
- the dimensions of the solution space (Section 3),
- the objective function (Section 4.2),
- and a list of hard and soft constraints (Section 4.1),

the computation component is generally applicable to timetabling problems.

We have tested the semantic components on a number of timetabling problems. The problems that we tackled with the semantic and calculation components are nurse rostering, sport timetabling and school timetabling. The database used for the school timetabling problem is the existing school database. As it turns out, it contains lots of irrelevant data for the generic framework. Although the mapping component can easily be extended with extra rules, we managed to map about 80% of the data automatically. The constraint and objective function components that we presented in this paper turn out to be sufficiently complex to express most the characteristics that we came across in the test problems. Future work will include further investigation of large, real world timetabling problems in order to fine tune all the semantic components.

References

1. T. Berners-Lee, J. Hendler and O. Lassila: The Semantic Web, Scientific American, May 2001
2. C. Bizer: D2R MAP - A Database to RDF Mapping Language, Proceedings of WWW, Budapest, 2003
3. T. Bray, J. Paoli, C. M. Sperberg-McQueen and E. Maler: Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation, 2000 (http://www.w3.org/TR/REC-xml)

4. E. K. Burke and P. Ross (editors): The Practice and Theory of Automated Timetabling Volume I: Selected Papers from the 1st International Conference on Practice and Theory of Automated Timetabling, (PATAT 1995) Edinburgh, Springer LNCS 1153, 1996
5. E. K. Burke and M. Carter (editors): The Practice and Theory of Automated Timetabling II: Selected Papers from the 2nd International Conference on Practice and Theory of Automated Timetabling, (PATAT 1997), Toronto, Springer LNCS 1408, 1998
6. E. Burke and W. Erben (editors): The Practice and Theory of Automated Timetabling III: Selected Papers of the 3rd International Conference on Practice and Theory of Automated Timetabling, (PATAT 2000), Konstanz, Springer LNCS 2079, 2001
7. E. Burke and P. De Causmaecker (editors): The Practice and Theory of Automated Timetabling IV: Selected and Revised Papers of the 4th International Conference on Practice and Theory of Automated Timetabling, (PATAT 2002), Gent, Springer LNCS 2740, 2003
8. E. K. Burke and S. Petrovic (editors): Timetabling and Rostering, European Journal of Operational Research (EJOR), Volume 153, Issue 1, 2004
9. D. Connolly, F. van Harmelen, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider and L.A. Stein: DAML+OIL Reference Description, W3C Note 18 December 2001 (http://www.w3.org/TR/daml+oil-reference)
10. P. De Causmaecker, P. Demeester, Y. Lu, G. Vanden Berghe: Using Web Standards for timetabling, In proceedings of 4th International Conference on the Practice and Theory of Timetabling (Patat), 2002, 238-257
11. A. Doan, J. Madhavan, P. Domingos and A. Halevy: Learning to map between ontologies on the semantic web, The Eleventh International WWW Conference, Hawaii, 2002
12. K. Dowsland: Off-the-Peg or Made-to-Measure? Timetabling and scheduling with SA and TS, In proceedings of 2nd International Conference on the Practice and Theory of Timetabling (Patat), 37-52, 1997
13. D. Gusfield: Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology, Press Syndicate of the University of Cambridge, ISBN 0521585198, June 1997
14. R. Harder: OpenTS - Java Tabu Search http://www-124.ibm.com/developerworks/oss/coin/OpenTS/index.html
15. O. Lassila and R. R. Swick: Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation, 1999 (http://www.w3.org/TR/REC-rdf-syntax)
16. Lockheed Martin: DAML Language Lessons Learned, http://ubot.lockheedmartin.com/ubot/lessons/language.html
17. L. Miller, G. FitzPatrick and D. Brickley: SkiCal and iCalendar in DAML+OIL: a case study, http://www.ilrt.bris.ac.uk/discovery/2002/03/skical-daml/
18. M. Misskof, R. Navigli and P. Velardi: Integrated Approach to Web Ontology Learning and Engineering, IEEE Computer 35 (11), 60-63, 2002
19. N. F. Noy and M. A. Musen: Anchor-PROMPT: Using Non-Local Context for Semantic Matching, Workshop on Ontologies and Information Sharing at the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001), Seattle, 2001
20. N. F. Noy and M. A. Musen: PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment, Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-2000) Austin, 450-455, 2000

21. S. Prasad, Y. Peng and T. Finin: Using Explicit Information To Map Between Two Ontologies, Proceedings of the Workshop on Ontologies in Agent Systems, 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems, Bologna, 2002
22. S. F. Smith and M. A. Becker: An Ontology for Constructing Scheduling Systems. In Working Notes from 1997 AAAI Spring Symposium on Ontological Engineering, Stanford, 120-129, March 1997
23. N. Stojanovic, L. Stojanovic and R. Volz: A reverse engineering approach for migrating data-intensive web sites to the Semantic Web, Intelligent Information Processing, IFIP 17th World Computer Congress - TC12 Stream on Intelligent Information Processing, Montreal, 141-154, 2002
24. M.A. Trick: Challenge Traveling Tournament Instances, http://mat.gsia.cmu.edu/TOURN/
25. A. Wren: Scheduling, Timetabling and Rostering - A Special Relationship?, Selected Papers of the 3rd International Conference on Practice and Theory of Automated Timetabling, Springer LNCS 1153, 46-75, 1996
26. http://ingenieur.kahosl.be/projecten/cofftea/2003/03/hybrid.daml
27. http://oiled.man.ac.uk/
28. http://www.cogsci.princeton.edu/~wn/obtain.shtml