



## An Task and Time Scheduler System Based on Rich Internet Applications

Miss. Rupali R. Waghmare\*, Miss. Reshma R. Gatade, Miss. Rashmi R. Rajurkar  
and Prof. A. R. Manakshe

B.E-I.T (Final Year)<sup>1,2,3,4</sup>, Jawaharlal Darda Institute Of Engineering & Technology,  
Yavatmal (MS) INDIA

[rups1.waghmare@gmail.com](mailto:rups1.waghmare@gmail.com)<sup>\*1</sup>, [reshmagatade@gmail.com](mailto:reshmagatade@gmail.com)<sup>2</sup>, [rashmi01\\_rajurkar@rediffmail.com](mailto:rashmi01_rajurkar@rediffmail.com)<sup>3</sup>  
[manakshe.amit@gmail.com](mailto:manakshe.amit@gmail.com)<sup>4</sup>

**Abstract:** The manual system of preparing time table in colleges with large number of students is very time consuming and usually ends up with various classes clashing either at same room or with same teachers having more than one class at a time. These are just due to common human errors which are very difficult to prevent in processes such as these. To overcome these problems people usually taking the previous year's timetable and modifying it but still it is a tedious job to incorporate changes. To overcome all these problems we propose to make an automated system. So our aim is to develop a general purpose which can efficiently generate optimal solutions.

**Keywords:** RIA, Constraint, Xml, Interface, Web

### I. INTRODUCTION

Time table scheduling has been in human requirements since they thought of managing time effectively. It is widely used in schools, colleges and other fields of teaching and working like crash courses, coaching centers, training programs etc. In early days, time table scheduling was done manually with a single person or some group involved in task of scheduling it with their hands, which take lot of effort and time. While scheduling even the smallest constraints can take a lot of time and the case is even worse when the number of constraints or the amount of data to deal with increases. In such cases perfectly designed time table is reused for whole generation without any changes, proving to be dull in such situations.

Institutions/Schools/Collages/Universities are the regular users of such time tables. They need to schedule their course to meet the need of current duration and facilities that are available to them.

### II. INTRODUCTION AND HISTORY OF RICH INTERNET

Rich Internet applications (RIA) are web applications that have the features and functionalities of traditional desktop applications. RIAs typically transfer the processing necessary for the user interface to the web client but keep the bulk of the data (i.e., maintaining the state of the program, the data etc) back on the application server. RIAs typically:

- run in a web browser, or do not require software installation
- run locally in a secure environment called a sandbox

The term "Rich Internet Application" was introduced in a white paper of March 2002 by Macromedia, though the concept had existed for a number of years earlier under names such as:

- Remote Scripting, by Microsoft, circa 1998
- X Internet, by Forrester Research in October 2000
- Rich (web) clients
- Rich web application

Traditional web applications centered all activity around client-server architecture with a thin client. Under this system all processing is done on the server, and the client is only used to display static (in this case HTML) contents. The biggest drawback with this system is that all interaction with the application must pass through the server, which requires data to be sent to the server, the server to respond, and the page to be reloaded on the client with the response. By using a client side technology which can execute instructions on the client's computer, RIAs can circumvent this slow and synchronous loop for many user interactions. This difference is somewhat analogous to the difference between "terminal and mainframe" and Client-server/Fat client approaches.

Internet standards have evolved slowly and continually over time to accommodate these techniques, so it is hard to draw a strict line between what constitutes an RIA and what does not. But all RIAs share one characteristic that they introduce an intermediate layer of code, often called a client engine, between the user and the server. This client engine is usually downloaded at the beginning of the application, and may be supplemented by further code downloads as the application progresses. The client engine acts as an extension of the browser, and usually takes over responsibility for rendering the application's user interface and for server communication.

What can be done in an RIA may be limited by the capabilities of the system used on the client. But in general, the client engine is programmed to perform application functions that its designer believes will enhance some aspect of the user interface, or improve its responsiveness when handling certain user interactions, compared to a standard Web browser implementation. Also, while simply adding a client engine does not force an application to depart from the

normal synchronous pattern of interactions between browser and server, in most RIAs the client engine performs additional asynchronous communications with servers.

### III. GENERAL PROBLEM FORMULATION

The considered college is a four-year college which has the following characteristics on its course administration. The college offers courses for daytime students. The classes for daytime students are scheduled in the weekday's daytime and Saturday morning. The types of lectures are theory lecture, tutorial, or practical. The class size of theory lectures is from 40 to 70. A minimum time slot is a 30 minutes interval. For theory and tutorial classes, 2 slot time corresponds to 1 credit, and 1 credit for practical takes 4 slots time. Once an instructor decides to offer a course for a specific year-session of a department, an offered course  $X_i$  takes place in the timetabling problem, which is expressed as a tuple of attributes (course, credits, department, instructor, year, section, class-group, course-type, lecture-type, Timeslots, Rooms). Except TimeSlots and Rooms, all attributes of  $X_i$  are determined at the time the course is decided to be offered. Both TimeSlots and Rooms are list fields to contain assigned time slots and classrooms for the course  $X_i$ . To indicate an attribute attr of an offered course  $X_i$ , we use the notation  $X_i.attr$ . The time slots are generally assigned from 11 AM to 5.30 PM for weekdays and from 7.30 AM to 11 AM on Saturday. The time slots are labeled as  $T_i$  ( $i = 1 \dots 55$ ). There are various constraints to be satisfied at the time to instantiate variables about time slots and classrooms. The constraints can be categorized into strong and weak constraints as follows:

#### A. Strong Constraints:

C1: A classroom is not assigned to more than one lecture at the same time.

C2: An instructor cannot teach more than one class at the same time.

C3: Courses for the same year-session students of a department cannot take place at the same time.

C4: The classroom for a course should have enough capacity to take students registered in the course.

C5: The classroom should be well equipped with required facilities for the classes.

#### B. Weak Constraints:

C6: The lectures are not assigned to time slots which are in the instructor's forbidden time zones.

C7: Instructors' daily lecture hours should be restricted to be within the allowed maximum hours.

C8: As far as possible, classes are scheduled in the instructor's preferred time zones.

C9: A lunch/dinner break must be scheduled.

C10: If possible, the lectures should not be scheduled on too late night time slots.

C11: The theory courses are scheduled on Monday and Tuesday, and the practical courses are scheduled on Wednesday, Thursday, and Friday.

C12: For daytime students, the cultural subjects courses are scheduled in the morning time slots (1st to 4th time slots

on weekdays), and major courses are scheduled in the afternoon time slots (5th to 8th time slots).

C13: For nighttime students, the cultural-subjects courses are scheduled on the 11th to 12<sup>th</sup> slots, and the major courses are scheduled on the 13th to 16th time slots on weekdays.

C14: If possible, the lecture hours for a course should be scheduled consecutively.

C15: As far as possible, classes should be scheduled in their corresponding department's exclusive-use classrooms.

C16: The classrooms should be allocated in a manner to minimize the distances between adjacent classes' classrooms.

It is desirable for timetables to satisfy all strong and weak constraints. However, it is usually not easy to meet all these constraints. The strong constraints must be satisfied all the times, but weak constraints can be somewhat sacrificed to find feasible timetables. Among the weak constraints, constraints from C6 to C14 are imposed on the allocation of time slots.

Constraints C15 and C16 are imposed on the allocation of classrooms. The constraints are arranged in the order of importance in the scheduling. For example, if it is impossible to find schedules satisfying both C6 and C7 simultaneously, it is preferred to choose a schedule that satisfies C6 but C7 rather than a schedule satisfying C7 but C6.

### IV. REQUIREMENTS AND ANALYSIS

In this section we formulate the various requirements of our project.

#### A. Automatic Generation of Timetable:

The prime requirement was to build the automatic weekly Lecture-Room timetable generator application for our college which will solve problem involved manual construction of timetable. There are always weekly or monthly changes in structure of timetable, so all the time reconstructing the timetable is very tedious job. So we wanted to build the automatic timetable generator algorithm that is adaptable to these changes. There are various timetabling algorithm developed for solving our college timetabling problem. But in most of the cases either the actual implementation is not available on the Internet or the projects which are available freely are not suitable to our meet requirement of our college's timetabling problem. Also due to fixed time period, we have decided to develop our own timetabling algorithm tailored to our college's timetabling problem rather than wasting time for searching various timetabling algorithms or ready to use software. Although during literature survey, we found only one open source software name UniTime available from [www.unitime.org](http://www.unitime.org) to be very impressive or effective to implement. But again this software was developed on very large scale, it has hundreds of packages and it has used all leading Java Web based frameworks including Strut & Hibernate. It uses MySQL database for storing its data.

So finally we decided that we will not borrow our algorithm from any other source rather we will build it our self. Next question came to mind was, which language we should use. Whether we should use LISP or Prolog? But since we were not familiar with these languages, we decided that we will go with JAVA only. Since JAVA is cross platform and

there are various IDE available to assist us building algorithm. Also GWT uses JAVA for writing web application. We decided that we will use JAVA Only.

Java is a programming language originally developed by Sun Microsystems and Details about Java released in 1995 as a core component of Sun's Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to bytecode which can run on any Java virtual machine (JVM) regardless of computer architecture. On 13 November 2006, Sun released much of Java as free software under the terms of the GNU General Public License (GPL). On 8 May 2007 Sun finished the process, making all of Java's core code open source, aside from a small portion of code to which Sun did not hold the copyright.

#### **a. Platform independence:**

One characteristic, platform independence, means that programs written in the Java language must run similarly on any supported hardware/operating-system platform. One should be able to write a program once, compile it once, and run it anywhere.

#### **b. Different criteria and conditions to consider while generating timetable:**

We wanted that timetable generator should be able to consider following different resources available

- a) Teachers
- b) Courses
- c) Semesters
- d) Rooms

It should provide mechanism for specifying various constraints such as

- a) Teacher-Room Constraints
- b) Year-Room Constraints
- c) Teacher-Timeslot Constraints
- d) Teacher-Day Constraints
- e) Teacher-Year Constraints

#### **B. Standardized Knowledgebase representation:**

Not just we wanted to implement automation process but we also wanted to keep out data in standard form, so that any external application can access our data. By doing to so we were thinking ahead of time. We were thinking about Future implementation of automatic timetable generator by some other groups or by our self in other language and on other platform. Keeping knowledgebase portable, we can write cross platform application and transfer our knowledgebase seamlessly between two platforms. We decided we will go with XML. Details about choosing XML for Knowledgebase are as follows.

##### **a. XML:**

The Extensible Markup Language (XML) is a general-purpose specification for creating custom markup languages. It is classified as an extensible language because it allows its users to define their own elements. Its primary purpose is to facilitate the sharing of structured data across different information systems, particularly via the Internet, and it is used both to encode documents and to serialize data. In the

latter context, it is comparable with other text based serialization languages such as JSON and YAML. It started as a simplified subset of the Standard Generalized Markup Language (SGML), and is designed to be relatively human-legible. By adding semantic constraints, application languages can be implemented in XML. These include XHTML, RSS, MathML, GraphML, Scalable Vector Graphics, MusicXML, and thousands of others. Moreover, XML is sometimes used as the specification language for such application languages. XML is recommended by the World Wide Web Consortium. It is a fee-free open standard. The W3C recommendation specifies both the lexical grammar and the requirements for parsing.

##### **b. Advantages of XML:**

- a) It is text-based.
- b) It supports Unicode, allowing almost any information in any written human language to be communicated.
- c) It can represent common computer science data structures: records, lists and trees.
- d) Its self-documenting format describes structure and field names as well as specific values.
- e) The strict syntax and parsing requirements make the necessary parsing algorithms extremely simple, efficient, and consistent.
- f) XML is heavily used as a format for document storage and processing, both online and offline.
- g) It is based on international standards.
- h) It can be updated incrementally.
- i) It allows validation using schema languages such as XSD and Schematron, which makes effective unit-testing, firewalls, acceptance testing, contractual specification and software construction easier.
- j) The hierarchical structure is suitable for most (but not all) types of documents.
- k) It is platform-independent, thus relatively immune to changes in technology.
- l) Forward and backward compatibility are relatively easy to maintain despite changes in DTD or Schema.
- m) Its predecessor, SGML, has been in use since 1986, so there is extensive experience and software available.
- n) An element fragment of a well-formed XML document is also a well-formed XML document.

##### **C. Responsive Interface:**

We knew that the success of any application lies not only in powerful algorithms but also in the successful graphical user interface designs. We wanted to build the Rich Internet Application which will exactly behave as our normal desktop application. So that user will find it friendly and easy to work on. We also wanted the output of our algorithm to be flexible enough so that we can zoom it, print it or able to modify it manually. Question was how to build the Rich Internet Application? So we did lot of survey on various platforms available for RIA. Finally we decided we will go with AJAX based Google Web Toolkit.

## V. IMPLEMENTATION

This application has been developed using Java as front end tool and MySQL Server as its back end tool. The application has been coded to be platform independent running on Java Virtual Machine. Netbeans IDE has been chosen as its development environment because of the following features

Designing interface for the application has been simplified by its drag and drop GUI pallet.

- a. Debugging can be easily done using the Logger class.
- b. Easy database access with NetBeans database plug-in.
- c. Simplified automated editor error detection.
- d. Automatic code generation.
- e. Automatic documentation.
- f. Simplified class factory method lookups.
- g. Easy to create jar \_les using build option.
- h. Profiling option.
- i. Project can be run on debugging mode which provides current state of the variables with the help of break points.
- j. Net Beans IDE has wide help and support on the web. While coding TimeGene application several constraints related to its computation has been taken into account. Timetable generating problem provides us with various alternatives in the design of the algorithm, interface and the database. Among the various designs what we have implemented is detailed below

### A. Interface Implementation:

There are ten classes each contains a JFrame which is associated with an interface.

The association are as follows

- a) \_ Main.class for login interface
- b) \_ TimeGene.class for basic information interface
- c) \_ Subjects.class for subject interface
- d) \_ Teachers.class for teachers interface
- e) \_ Batches.class for batches interface
- f) \_ SubTea.class for Subject Teacher interface
- g) \_ BatTea.class for Batch Teacher interface
- h) \_ SelBat.class for Batches Selection and Priority interface
- i) \_ ShowTable.class for Timetable Output interface
- j) \_ SelectTable.class for open and save interface

#### a. Login Interface:

Enter the MySQL login name and password as a primary information

#### b. Basic Information Interface:

Enter basic information related to collage name, academic year, select working hours from drop down list and select check boxes for working days. To open saved table click on Open Table. To know more on Time Gene click About.

#### c. Subject Interface:

Enter the Subject ID and Subject Name, enter continuous working hours per week, enter the number of teachers needed in subject and click Save. To edit select the Subject ID and

click Save. To remove select the Subject ID and click Remove.

#### d. Teacher Interface:

Enter the Teacher ID and Teacher Name and click Save. To edit select the Teacher ID and click Save. To remove select the Teacher ID and click Remove.

#### e. Batch Interface:

Enter the Batch ID and Batch Name and click Save. To edit select the Batch ID and click Save. To remove select the Batch ID and click Remove.

#### f. Subject-Teacher Interface:

Select the Subject ID and their respective Teacher ID. Click the double headed symbol to add or remove from Selected Teachers list.

#### g. Batch-Subject Interface:

Select the Batch ID and their respective Subject ID. Click the double headed symbol to add or remove from the Selected Subjects list.

#### h. Batch Selection Interface:

Select the Batches to be scheduled. Click the double headed symbol to add or remove from the Selected Batches list and if necessary set or remove subjects priority for the selected batch. The priority is set by selecting subject, day, hour from the drop down list.

#### i. Timetable Output Interface:

Select Batch tab to view associated batch time table. Select teachers view or subjects view to display time table in teacher or subject respectively. To edit generated list select the period and edit using keyboard. To save generated time table click Save.

#### j. Save Table Interface:

Enter filename to be saved to the database and select save. To replace an existing file or delete a \_le, use delete and then save the new timetable to the database. To avoid changes click cancel.

#### k. Open Table Interface:

Select the filename and click the Open Table option. To delete an existing \_le, select the \_le and click delete. To go back press cancel.

## VI. GENERATION OF TIMETABLE

The system will take various inputs like details of students, subjects and class rooms and teachers available, depending upon these inputs it will generate a possible time table.



Day/Time	11.00 To 11.30	11.30 To 12.00	12.00 To 12.30	12.30 To 13.00	13.00 To 13.30	13.30 To 14.00	14.00 To 14.30	14.30 To 15.00	15.00 To 15.30	15.30 To 16.00	16.00 To 16.30	16.30 To 17.00	17.00 To 17.30
Monday	8th DSP R2 m.s.a	SE R2 s.m.g	Recess	DSP Pract C-Lab	m.d	Project And Seminar C-Lab							
Tuesday	8th DSP R2 m.s.a	SE R2 s.m.g	Recess	MMT Pract C-Lab	s.t	Project And Seminar C-Lab							
Wednesday	8th DSP R2 m.s.a	MMT R2 g.r.b	Recess	MMT Tut R2	g.r.b	IT R2 m.a.p							
Thursday	8th DSP R2 m.s.a	MMT R2 g.r.b	Recess	IT R2 m.a.p	IT Pract C-Lab	r.p.s							
Friday	8th SE R2 s.m.g	MMT R2 g.r.b	Recess	IT R2 m.a.p	IT Tut R2	m.a.p							
Saturday	8th SE R2 s.m.g	MMT R2 g.r.b	Recess	IT R2 m.a.p	Project And Seminar C-Lab								

Figure: 1

## VII. ADVANTAGES

- This project has educational benefits and administrative efficiencies.
- Made in Java, hence it is extensible, platform independent and robust.

## VIII. CONCLUSION

In this project we have designed and implemented our Java based automatic college timetable Generator. Also we have designed the standard XML bases knowledge base for storing data, rules and facts. Finally we can say that our implementation is successful and practically application because to generate timetable for one department requires a very small time varying from 2 sec to 5 sec only. So our algorithm is definitely faster.

## IX. REFFERNCES

### Journals

- Practical Timetabling Algorithm for College Lecture-Timetable Scheduling Springer-Verlag LNCS
- Practice and Theory of Automated Timetabling Volumn IV. (1997).
- M. A. Saleh Elmohamed, P. Coddington, G. Fax. A comparison of annealing techniques for academic course scheduling. In Practice and Theory of Automated Timetabling II (E. Burke and M. Carter(eds.)). Springer-Verlag LNCS 1408. (1998)

### Web References

- Google Web Toolkit
- [http://en.wikipedia.org/wiki/Google\\_Web\\_Toolkit](http://en.wikipedia.org/wiki/Google_Web_Toolkit)
- <http://code.google.com/webtoolkit/resources.html>
- <http://code.google.com/webtoolkit/gettingstarted.html>
- <http://googlewebtoolkit.blogspot.com/2007/12/developing-gwt-applicationswith.html>

### html

- XML Tutorial
- <http://www.w3schools.com/xml/>
- <http://en.wikipedia.org/wiki/XML>
- dom4j: the flexible XML framework for Java <http://www.dom4j.org/guide.html>