University Timetable Generator

A PROPOSAL TO:

Kosie Eloff

02-20-2012

Group name:
STOPWATCH

Members:
Hendrik Prinsloo s10022806

Contents

P.1) The Problem
- Motivation
- How it is done now
- Why would students be interested?
- Our solution to the problem

P.2) Technicalities
- The testers
- Reason

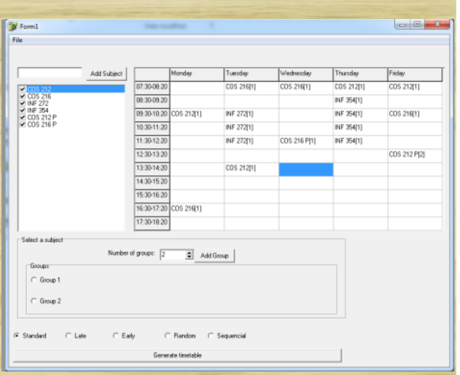
P.3) Initial Requirements
- Functional
- Non-functional
- Data Requirements
- Environmental

Introduction

We have realized a problem for which there are no computer solution for yet. Most students are faced with an unnecessary task at the start of each semester of working out their timetable by only referencing the standard Timetable book the University publishes each year. This is a messy and time-consuming exercise which can be avoided. We will develop a simple but effective application to solve this.

Motivation

Each year students are supplied with a timetable book which consists of 150+ pages, +-1800 different modules. Each module has on average 3 different groups. This means there are over 5000 different groups. The average student has about 6-10 modules per semester. Statistically 24 groups to choose from with an improbable 6000+ permutations. Clashes between different groups brings the possible combinations down dramatically. The time it takes you to generate the possible combinations is the primary objective of our project.



☑ COS 212
☑ COS 216
☑ INF 272
☑ INF 354
☑ COS 212 P
☑ COS 216 P

	Monday	Tuesday	Wednesday	Thursday	Friday
07:30-08:20		COS 216[1]	COS 216[1]	COS 212[1]	COS 212[1]
08:30-09:20				INF 354[1]	
09:30-10:20	COS 212[1]	INF 272[1]		INF 354[1]	COS 216[1]
10:30-11:20		INF 272[1]		INF 354[1]	
11:30-12:20		INF 272[1]	COS 216 P[1]	INF 354[1]	
12:30-13:20					COS 212 P[2]
13:30-14:20		COS 212[1]			
14:30-15:20					
15:30-16:20					
16:30-17:20	COS 216[1]				
17:30-18:20					

Select a subject
Number of groups: 2
Groups
Group 1
Group 2

[✓ Standard] Late Early Random Sequencial

First rough prototype of the proposed application.

The Problem

How it is done now...

Except for the few faculties which give standard timetables to their students, most students create there timetable from scratch, by hand. This can be a time consuming and frustrating exercise especially if the given student is dragging a few of his/her previous year's modules. This causes clashes and can take the individual hours to find the right permutation. Students also may want to schedule their timetable to fit daily activities, introducing another factor to increase the difficulty. On the other hand the University hands out these books to each student on registration, money and trees can be saved by eliminating this.

Why would students be interested?

Almost all software are developed with the idea of increasing end-user efficiency and effectiveness. Some students might feel that they have no problem paging through the timetable book in order to find their subject and generate their personal timetable. But there are always the chance of overlooking something as human error is ever present. This application will allow students to specify their subjects, fetch the subject's groups from an external resource, and automatically generate all possible permutations. The student will be able to scan through the possible generated timetables and choose one that best suites him/her. This will save time and ensure the user that no mistakes have been made.

Our solution to the problem

First of all and probably the most drastic. There are always changes to the timetable's venues/times/groups. Having these printed on paper provides a probable timetable layout but it cannot be guaranteed to be 100% correct/up-to-date. Having a centralized data-source will assure students that their timetable is generated with the most recent version. We propose to supply all students with a digital copy of the application, downloadable via the university's website. This application will need a data-file (Most probably XML) to work with. This file will also be published online and updated as needed.

Technicalities

The lab-rats

From left to right:

1.	Fransoic Strydom	- BSc Actuarial and Financial Mathematics (2nd)
		- franni@vodamail.co.za
		- 0828726050
2.	Pieter Cronje	- Bcom ekonometrie (3rd)
		- pieter@cronierwines.com
		- 0828206904
3.	Marnus Heunis	- Bsc IT (1st)
		- spoefter@gmail.com
		- 0724120272
4.	Henri van Staden	- Financial Accounting (3rd)
		- krokivstaden@gmail.com
5.	Andrew Barnes	- Electrical Engineering (4th)
		- 0724486744
6.	Ricus Papp	- Tourism (Final)
		- ricuspapp@yahoo.com
		- 0785834432

Reason for group

This group represent the most popular faculties which makes up the most students of the university. They also vary in computer literacy which makes for a great control group. As our application will be all about end-user involvement, we will focus mainly on user-centered design. This will give us the best guidance in developing the most user-friendly application.

Initial Requirements

Functional Requirements

The application should be able to collect data on the internet from a single data-file and use it to generate an timetable according to the modules the end-user has specified. It should be quick, efficient, accurate and user-friendly. There are two ends to the system. The end-user and the data-file upkeep. The data-file upkeep should be made as easy and quick as possible. The user should have the ability to save, load and print his/her timetable.

Non-functional Requirements

Our aim is not to develop the application for multiple platforms. Windows is the most commonly used platform among students. The application will mainly be used twice a year, which means that we need to have structures in place to ensure access to the data-file, as it may be in high demand. A default data-file format should be chosen, we will probably make use of XML. Furthermore the application should be as small as possible to ensure quick and cheap distribution. No third party installations should be required as we only want it to be in the form of an single executable.

Data Requirements

There will be made use of only a single data-file which contains all the information needed by the application. It will in XML format. The file should not be larger than 2MB, as this will delay the download time. This file will be updated only if changes are made.

Environmental Requirements

Physical Environment

The single instance of the application will run on each user's computer. This application will require a data-file which will be available to download online. This file may be in high demand at the start of each semester, therefore quick distribution is needed.

Social Environment

The single instance of the application will run on each user's computer. This application will require a data-file which will be available to download online. This file may be in high demand at the start of each semester, therefore quick distribution is needed.

Organizational Environment

The only form of organizational environment will be the communication between the users and the data-file. And the fact that the data-file must be updated as regularly as changes occur. This will be a simple exercise.

Technical Environment

The program must be compatible with entry-level computers. No installation will be required. An internet connection will be beneficial to the end-user as it will guarantee the most recent data. Otherwise the student can also get the data file from any computer on campus through the intranet.