Engineering Design for Software Development 2 Projects

Project A: Dynamic Traffic Simulator

Consider a single lane one-way road. The capacity of this roadway is determined by a number of factors including the speed of the traffic using the road, the travelling distance between vehicles and the acceleration and deceleration properties where speed changes are required (such as for traffic lights or changes in the speed limit). All of these factors can vary by the vehicle type.

Providing additional running lanes can increase the capacity of this road. This allows faster traffic to pass slower traffic, but consideration must then be given to safe movement between lanes and the clearances (in terms of both time and distance) required.

A more complex solution still is the provision of a single lane in each direction, but the possibility to use the roadway in the opposite direction for passing, but this depends on traffic flows in the other direction too.

Design a piece of software that can be used to define and model road traffic that is able to predict the capacity of a section of roadway. You will need to define the road network to be simulated – including safe speeds and clearances - and the traffic being carried on that roadway – including speed and acceleration profiles.

Your design should show that you have considered how to approach the multi-lane and bi-directional roads, but need only implement the simplest case. Additional credit will be given for completing further areas, for providing a simple means to change the parameters of the model, and for providing the modelled results in easy-to-understand ways.

Notes

This could apply to any network, not just roads – perhaps if multiple groups are doing it, we should have each investigate the particular curiosities of their network type and design the classes and algorithms appropriately.

Project B: Database Engine

Design and implement a simple database system.

The Relational Database Management System (RDBMS) is a core part of many corporate systems and increasingly there is a requirement to store information within embedded systems for quick and easy retrieval for the control system. Such systems must allow the user to define a series of tables (relations) and then perform create, read, update and delete (CRUD) operations on the tables to manage the data. These operations use a standard grammar known as the Structured Query Language (SQL).

RDBMS systems can require many components, and you should investigate the key features and components of database systems. Having done so, you should design a simple single-user system in which you can define a structure and store arbitrary data (to disk). You should first consider the basic data structures or classes and I/O requirements that will need to be met in order to store the data. Following this, you will require a parser that can understand simple SQL commands typed by the user. A database engine will then be required to interface between the parser and the I/O module.

The project could be very large – you will not be expected to deliver the full range of features in commercial RDBMS, rather you are to identify key features required to meet the specification provided. A key part of this project will be defining the functionality that you intend to deliver and then delivering it.

Your design should consider how other components of an RDBMS that you read about in the design stage might be integrated later.

Notes

The project could be very large – you will not be expected to deliver the full range of features in commercial RDBMS, rather you are to identify key features required to meet the specification provided. A key part of this project will be defining the functionality that you intend to deliver and then delivering it.

Project C: Steganography Application

The aim of this project is to design and demonstrate a steganographic application that allows additional messages/information to be encoded within an image file. The application should be able to both encode and decode the messages in a variety of different images using a range of steganographic techniques. The user should be able to input the parameter values to set the level of hiding that is used. Stealing different numbers of bits or pixels to hide the user information will have varying degrees of impact on the quality of the viewed image and users should be provided with the means to grade or comment on picture quality.

Notes

There are a number of steganography methods available online. These are worth exploring but the group is expected to implement a number of these within their application or indeed propose their own.

The user interface should also allow users to compare image quality and robustness. You may wish to research and consider any standard quality measures as reference but the ability for users to rate the quality of the images (Quality of Experience) would be desired function in the final application.

Project D: Journey Planner

Design and implement a journey planner for an interconnected network – perhaps the Glasgow Bus System, the London Underground or an airline (real or fictional).

You should design your software to allow the user to input pertinent planning details – such as being able to minimise the time taken to traverse the network, the number of changes/hops or via/avoid information.

Consider the type of network you would like to model and the transit opportunities. As an example, walking a route or sending a message over a network can start any time, but for timetabled transport you may only have one or two opportunities per day to arrive at or leave a node. The type of network will also identify other potential requirements, such as allowing the user to enter an "arrive before" time, a "leave after" time and so on.

Additionally, you should have the software be able to at least read the network topology from a file, but may wish to provide additional functionality to allow the network to be altered (by adding and removing nodes and arcs) and saved for use again later.

Notes

For the core problem of assessing the shortest path in a network, it's expected some implementation of Dijkstra's shortest path algorithm be put together, but where the parameters can be tweaked (which means it'll take a bit more thought than just downloading an example from the web!).

Another alternative is the data-driven approach, where a timetable is stored and the potential for routes then varies, as the interchange time at each hop must also be considered.

Equally, other solutions might be nice… If you get far enough, the cost function could be dependent upon the parameter being minimised and it may be recognised that a "via" is actually two separate paths whereas an "avoid" can simply either remove that node from the network or add an infinite cost at that node. Please note that this is not just a route planner but a journey planner using scheduled transport services.

Project E: 3D Route Planner

Most route planners operate only in two dimensions (x & y) which makes sense for say walking, running, cycling and public transport options. However, not all journeys will be in two dimension; consider navigating through a shopping complex or even a University campus where, in moving from source to destination, stairs, lifts and escalators will be used to move (in the Z direction) between levels. The task in this project is to develop a route planner that will allow users to determine a path (between source/destination pairs that are not always on the same X, Y plane. In this task, the project is to develop 3-D route planers that can be used in one of the following applications:

a) Provide directions for shoppers within a retail environment to locate and travel to their desired destination. The application should take into account journey times, physical restrictions (disability issues, prams, trolleys etc) and also the nature of the search. Users may wish to locate the nearest specific store but also say the nearest car-park, food outlet, toilet, cash dispenser, first aid station, etc and thus the application should accommodate specific ad general request.

b) Provide directions for students to navigate through a University campus where buildings are joined and rooms located on multiple levels. Campus navigation is a challenge as often moving between classes is a time constricted issues (say between lectures) and also must take in to account disability issues

c) "Place and route" for Printed Circuit Board (PCB) design. In this example, the problem looks like a traditional routing issue but one where you need to support multiple simultaneous paths and minimise the number of level changes.

Notes

Like the journey planners, the core problem of assessing the shortest path in a network, will require the use of a formal algorithm such as Dijkstra's shortest path algorithm. However to move in 3 dimensions will need a bit of thought and tweaking of the algorithm; not just a cut-paste from the web.

The task is quite open-ended and a bit of a challenge; it is up to the group to determine which of the above applications they will tackle.

Project F: Student Workload (Life) Planner

Workload planning is an essential part of student life – balancing the demands of a busy lecture/lab/tutorial timetable with completing course work and extra-curricular activities. The aim of this project is to develop a software application that allows a student to plan and schedule their time to meet all these conflicting demands. Rather than being a rudimentary calendar that simply records times and dates, the user is allowed to enter activities and tasks, and the application will then allocate time (timetable) sufficient to meet the specified needs of the job. Another feature will be to allow the user to specify an activity of a particular type and total duration and to then schedule slots accordingly. That is, say a report required for lab must be written and 3 hours is allowed for this task, the application may for example schedule three hour long slots within the week to for this task. As tasks can often take longer than first thought, the software should also be able to adapt to changes to task requirements while the tasks are in progress – e.g. you get a request to work another shift in the Union Bar and thus need to reschedule meeting your folks and working on a University assignment.

Notes

This is not an academic- centred task. The software should able to take account of all aspects of student life and thus activities associated with socialising, part-time working, sports, entertainment should be included. The challenge in this project is probably within the data modelling and specification rather than with the C++ coding.

Standard templates for timetables and time usage can be specified and uploaded via input files rather than always prompt the user to enter in all data by hand.

The group should also think about a multi-user mode such that a later version of the software may allow multiple users to use this application to plan collaborative works such as EE273 project work!

Project G: Salon Management

Create a program that can be used to manage appointments and services with in hairdresser's salon. In many Salons, rather than being an employee, the stylist simply rents a chair (or chairs) from the Salon manager and provides the services (cuts, styling, colouring, extensions, drying etc) to customers either on an appointment or walk-in basis. The salon provides the facilities as well as handling bookings and payments in return of either a fixed fee or % return.

The proposed software must be able to cope with a multiple types of service providers (stylists, consultants), multiple numbers of providers who will utilise facilities (chairs) on a regular or irregular basis. There will be range of different services provided; each requiring different durations and incurring different charges; the booking system must ensure that clients are not kept waiting or given insufficient time while also endeavouring to maximise occupancy of the Salon as a whole.

As well as dealing with bookings, the code must also deal with costings and keep a track of projected income for chairs and stylists as well as actual income. Remember, clients may often cancel an appointment while it also possible to have walk-in clients arrive unannounced. Note also that services may not just be rented for a full day but perhaps for part day – morning, afternoon or evenings.

The code produced should be extensible and apply to a range of establishments; large or small. For the purposes of demonstration and testing, the demo-ed system can be limited to manageable numbers but these limits MUST not be hard coded.

Notes

Scissors rather than circuits! Will require a wee bit of investigating. However, the business model that is used is quite universal and can be applied to a range of applications and businesses.

Project H: Inventory Management System – The Internet and the Fridge!

Inventory management systems are often thought of being the sole preserve of warehouses, retail suppliers and larger organisations. However, with proliferation of the Internet things are changing and inventor management systems may have a role to player closer to home. Consider the home environment where intelligent devices can monitor the stocks or health of items in a fridge or storage space and potentially flag up when stocks are low or use-by dates are exceeded. Or, the small business (restaurant, plumber, flowershop, designer, etc) where it is crucial to keep track of what is available or shortly to be required to enable the organisation to perform effectively. Consider the case of starting to cook lasagne and then finding out you are short of pasta sheets!

This project involves the design and implementation of an Inventory Management System used is used to track stock levels, product popularity and help the organisation/individual make informed decisions about restocking, cash flow, product range, etc and also communicating with suppliers

Items may have short lifetime (expiry date or technical obsolescence) and may have variable lead times before the items can reach the shelves. The program should have an interface to allow operators to input stock levels and consumption while it produces report table on item popularity, residual stock and generation of order forms. Additionally, it should trend those metrics over time. The reports maybe requested on the basis of a single item (i.e. milk) or an item category (i.e. diary).

The IMS can be extended further to manage multiple sites and should show that the design has considered scalability issues. Various backends maybe used such as file based storage and Relational Database Management Systems.

Notes

The choice of application domain or sector is left up to the group but the final version and examples used should reflect the domain. The Internet and the Fridge refers to the case when the IMS system links

Project I: Project/Topic Allocation System

In this project, the objective is to match, as well as possible, students' preferences for projects given in a list of options, offered by different supervisors, with an even distribution of supervision responsibilities.

Each supervisor should offer between 5 and 10 project titles. Each title will have a unique identification number, of the form *x*.*y* where *x* is the supervisor's ID and *y* is the ID of the project for that supervisor. Students can see these and can select up to *p* projects, in order of preference with the restriction that they cannot select more than *n* projects by the same supervisor where *p* and *n* are given the program's user (and may default to 4 and 2 respectively).

The purpose of the project allocation software is to

a) read in a list of project titles and their unique identification numbers;

b) read, for each student, the set of *p* chosen projects with their order of preference;

c) for each student, allocate just one of their chosen projects to them such that no one supervisor has to supervise more than *s* projects (*s* is an integer specified by the user), and giving students their preferred titles wherever possible.

(The value of *s* might default to (*nstudents*/*nsupervisors*)+1 where *nstudents* is the total number of students that should have projects allocated and *nsupervisors* is the total number of supervisors).

The actual allocation might be done using a heuristic method, i.e. one based on trial and error based on rules that you develop, or some optimisation algorithm, perhaps complemented by some heuristics. An example of the latter is 'constraint satisfaction', for which publicly available library code can be found on the internet.

As well as research into and development of the main allocation algorithm, the project will involve definition of appropriate classes to represent students, supervisors and projects. Students should be identified by name and registration number, supervisors by name, ID number and the name of the research group of which they are member. As already noted, projects will have a title and an ID number that identifies the supervisor and a specific ID for the project.

To test the program, it may be necessary to write another program to generate some typical data specifying the set of supervisors, some projects, the set of students and their choices. These might be written to suitable files that can then be read by the main project allocation program. Sample excel or csr files can be provided to test the application.

It will generally be possible that some supervisors will have had too few of their projects selected by students to take up enough of the supervision load so that other supervisors are not overloaded. That, in turn, would lead to some students being unallocated. In that instance, a supervisor and student may be paired even without specifying a project, provided the supervisor is in the same research group as at least one project within the student's set of *p* preferences.

Some degree of data checking would be required when first reading in the sets of students, choices, supervisors and projects offered. For example:

1. each student should have used all their *p* choices.
2. no student should choose more than *n* projects by the same supervisor;
3. each supervisor should have offered at least 5 projects.

(It is generally assumed that the total number of students will be less than 5 times the number of supervisors, but this, too, should be checked). Each problem should be logged and written to a file with the user asked to correct the errors and re-run the program. (In practice, students and supervisors would be asked to correct the errors and submit a new set of information).

Notes

There are maybe a number of heuristics that you may wish to consider when doing the allocation. You should include these in your specification and in your solution as you feel appropriate. You should however recognise that time is limited and will you are unlikely to be able to implement all options; thus you should consider in your implementation, the possibility for additional algorithms being added at a later date.

Project J: Cinema or Theatre Booking Application

Most people are familiar with the process of buying cinema or theatre tickets – either in advance or on the day. Develop a program that allows users (customer or staff) to book seats for individual showings of films or theatre events (you determine the specific application). Films/shows are usually scheduled on a week by week basis with showings at specific times in specific venues (screens or stages) with the possibility of different prices for different seats or showings. Your program should allow users to search or find available showings associated with a particular event along with a list of available seating. It should be possible to then reserve a specific seat (or seats) for the customer.

Notes

You will need to think about how to represent the venues and seats as well as particular shows with associated variable pricing and timings. Furthermore you will need to consider individual shows or films within your application. The user interface will be important and it will be ideal to have some form of graphical interface to support the user. You need not worry directly about payment – simply divert the user to a "dummy" payment method to represent that part of the process.

Project K: The "Game of Life"

The Game of Life (or Life) was originally proposed by John Conway in the 1970s. An example of cellular automation, it evolved from an academic curiosity to have applications and uses in a wide range of areas ranging from economics, philosophy to network planning and video games. The zero player game operates in a very simplistic manner, based around two dimensional grid of square cells, and has four very simple rules. Starting from some initial value, the grid states change over instances of times to create repeating, decaying patterns of interest. Students fill in the rest with background research! Interesting!

In this project, the team will produce a program that is able to demonstrate the "Game of Life" showing a range of still, oscillating and dynamic patterns, including gliders and spaceships. Secondly, the program will then support a 1 or 2 player game such that a present or random pattern is applied to the grid and each player takes turns in adding one cell of their own and removing one of their opponent and then the "Life" rules are applied. The game ends when one player has all their cells eliminated.

The group is encouraged to be creative and imaginative in the application of the "The Game of Life" algorithm and also in the user interface. The application should be informative and fun.

Notes

A well-established and simple algorithm so the challenge is to actually bring something new to the problem. Simply getting stuff from the internet will not be acceptable!

Project L: An Object Oriented Wind Farm Simulator

This project involves designing and implementing software which simulates the output of a wind farm or a group of wind farms based on wind speed data for different sites.

Power output of a wind turbine is generally given by:

$Power = \frac{1}{2} rAv^3$ (Where r=air density kg/m3, A=rotor area in m2 and v=wind speed in m/s)

Importantly, turbines cannot generate at all wind speeds; a minimum wind speed is required before the turbine can generate (the 'cut in' speed, typically between 3 and 5 m/s). Similarly, in strong winds turbines can reach cut out at high speeds (typically around 25m/s), where they stop generating. In between, there is generally a 'rated speed' (typically around 12m/s) after which the blade pitch is adjusted to utilise the wind energy efficiently but not exceed the rating of the machine. The simulator should take these features into account.

Another important measure is the capacity factor of a generator, which is defined as:

$$CapacityFactor = \frac{ActualPowerGenerated}{PowerGeneratedAtFullCapacity}$$

For example, if a generator rated at 2 MW produces 1MWh of power over the period of one hour, it has a capacity factor of 50%.

The wind farm simulator should allow the user to input the details of all the wind turbines which constitute the wind farm and store those details in an object oriented fashion. Based on a time-series wind speed profile data (comma separated values captured at a constant time step), read in from a file or set of files, the simulator should:

- Give the user the option of adding randomly generated variance to the wind profiles, i.e. adding a randomly generated number between a user-defined range to each point in the time series data. The new profile should be written to a file viewable in Excel.
- Calculate the total power generated by the wind farm at each time step;
- Output the total power output time series data to a commas separated data file, which can then be displayed using Excel;

- Generate a report in the form of a text file which gives the following details:
    - The total power generated across the entire profile at each wind farm and the total across a group of wind farms;
    - The income generated for the wind farm for a user defined price for each kWh generated by the wind farm; and
    - The total income generated by the group of wind farms;
    - The capacity factor of the wind farm and a group of wind farms.
    - The periods of a given profile where turbines were unable to generate due to high winds and insufficient wind on a turbine by turbine basis.

There are several different types of wind turbines based on different types of electrical machine, e.g. synchronous machines and induction machines. An advance feature of the simulator will be able to model different types of wind turbine and assess their reactive power import/export of the wind farm in addition to the real power export. Advanced OO design techniques should be used to allow additional turbine models to be added to the simulator in the future.

Project M: Electricity Security of Supply

Consumers, businesses, the energy regulator OFGEN and the government are all concerned that demand for electricity can be met. This is heavily dependent on both the magnitude of demand and the availability of electricity generation capacity. Demand for electricity is at its highest in the winter, and is especially high in cold winters. Normally, there are enough power stations and individual generating units available at them to meet the highest demand, but if there have been many breakdowns and the weather is especially cold, there might not be. What is the probability of there will not be enough generation available to meet the peak demand for power on a particular day?

The software should be capable of meeting the following objectives.

1. To read in a user-specified file containing specification of the power stations to be modelled. This should include: the name of the power station; the type of power station (e.g. coal, nuclear, wind, …); the total capacity of the station; the number of generating units at the station (each unit may be assumed to be of the same size); for thermal power stations, the winter and summer availabilities of each unit; for wind or hydro, the winter and summer load factors of the station.
2. To ask the user to specify the winter peak demand and the summer minimum demand
3. To ask the user to specify the standard deviation of demand variation.
4. To ask the user to specify a number of trials to be carried out in a Monte Carlo simulation.

5. To carry out a Monte Carlo simulation of 365 days of a year, with the peak demand and availability of generation on each day sampled and tallied, and to perform that simulation the number of times previously specified by the user.
6. To show, on the screen, the following summary statistics: average generation available in winter; average generation available in summer; the number of days on which the available generation was insufficient to meet demand.
7. To write daily total available generation for each generation type and the sampled peak demand out to a file for subsequent reading into Excel and plotting of histograms, calculation of other statistics, etc.

Project N: Battleground Game

In this project, the group is required to devise and implement a computerised version of game based upon the old fashion battleship game. In this game, the environment must be extended to be multi-terrain - land, sea and air - and that mobility is enabled such that hazards/targets/obstacles are allowed to move (following defined probabilistic mobility models) and hide in between user turns. The general specification of the game must be set the group itself who determine the overall functionality and operation of the game. For those who prefer not to think in terms of such defence/militaristic concepts it is possible to rephrase or change the context of the game to reflect an alternative "search and find" context – providing the difficulty or complexity is not unduly compromised. As a game application, the group must consider the user interface and also the option for single or multi-user operation along with the ability to determine degree of user ability. Think of options to allow users to specify game/player standards such as novice, easy, advanced etc. Also consider the narrative associated with the game as it the goal is to produce an innovative and interesting game – not simply a copy of a standard game.

Notes

Not as straightforward as you may think it is. Requires the group to bring something new to the problem. A user interface would be good.

Project O: Prime Number Football game

The project is based around a game suggested by Marcus Du Sautoy in his "The Number Mysteries" – see page 5. Here you are required to implement a software version of general game based upon the principles described in the book. Clearly, the first task is to obtain the original specification of the game and then define a specification for a software version of the game and any associated features to enhance or extend the game. The application should provide the option for single or two player operation and for a variable number of team members – Marcus Du Savoy original specification stated 3 players per team. The additional challenge of the group is how to display the game on the screen and show player and ball position and movement. This project is NOT FIFA2016 ;-)

Notes

A good introduction to number theory and also basics of cryptography. Look at the interface issue too.

Project P: Elevator Simulator

We have all spent time waiting for a lift/elevator to arrive in order to convey us up or down a building. The complexity of lift/elevator control is deceptive – it is more complex than it looks and is often complicated by individual user behaviour; impatient or malicious users who press the "wrong" buttons! In this project, the group is required to develop a programme that simulates the operation of arrays (columns) of elevators running over a multi-storey building. The program (application) should be able to cope with user defined number of elevators and number of storeys along with an ability for a user to simulate or represent elevator passengers making request for elevator service to particular floors. The program should have a user interface that displays the elevators' states allows user input and ideally shows (simulates) the vertical movement of each elevator.

Notes

A real engineering problem – the issue of how elevators respond to user requests is an interesting issue – how to ensure that customers are served as quickly as possible AND with good energy utilisation – try to avoid needless traversing of floors. A form of travelling salesman problem.

Project Q: Traffic Light Network Emulator

Red, Amber, Green…. The operation of traffic lights and pedestrian crossings are, for all us, a familiar and an ever present in our daily lives. The operation and sequence of traffic lights is well understood but the timing and placement of such services is not a trivial task. The aim of a traffic light system is to facilitate the smooth (efficient) flow of traffic via our road system. In this project, the group is required to develop an application that allows the user to configure a network of traffic lights (with or without a pedestrian crossing feature) across an urban environment and then show/demonstrate its operation in a time ordered fashion. The user should be able to set up a network of roads with sets of lights at each junction and then step through the network operation while a "car" or user journeys through the system. The user may be a pedestrian or road user (car, motorcycle or bike). The programme should have the ability to define operation of light sequences (at each junction) to be globally synchronised or indeed to work independently – subject to the requirement that no junction would permit contradicting lights to be "Green" simultaneously. No deliberate crashes! As a first consideration, the group may want to consider grid based road systems such as Glasgow and New York as exemplar networks. It is left to the users to consider if filter lanes are to be considered.

Notes

Need to look as some of the other projects as there is a link with routing and traffic management. Note ascii characters – can change colours of display without the need to for GUI code – although use of GUI would be good.

Project R: Rock Band Family Tree

Genealogy – (from Greek: γενεά genea, "generation"; and λόγος logos, "knowledge") represents the study of family history and is a widely followed activity supported by a range software products. In the 1980/90s Pete Frame (a music journalist) published via the NME and in book form a series of articles illustrating the genealogy (before the word was fashionable) of a range of important (and not so important) rock bands that indeed even spawned a BBC documentary series. In this project, the group is asked to develop a genealogy application that supports the recording/display of rock-band information, making links between group members, instruments, producers, genres, songs etc. that shows how bands, artists have developed over the years. One appreciates that Facebook, Twitter and other social media applications (along with the internet) may already provide such information and indeed will be the source of your application. One appreciates that the use of the word "rock" is misleading and perhaps dated so the project application is not constrained to a single genre. Groups are free to consider alternative domains such as classical music and artists, provided the application is sufficiently rich in detail.

Notes

Look at applications such as Family Tree maker and various Genealogy sites to gain an insight as to what is required but applied to musicians and bands.

Project S: Command Line Graphical Calculator

The aim of this project will be to implement a system along the lines of Matlab/Maple/Mathematica, where you can input equations and plot simple representations of data, such as boxplots and bar charts, on the command line. This will involve some form of string parsing/tokenisation to parse an equation, involving decisions on operator precedence, as well as having some form of recursion/looping to use every token in the equation. Classes/objects involved:

- Equation solver
- Equation element
- Plot area
- Plot axes
- Plot artefacts/elements

The key challenge is to be able to parse equations and the need to define their own library of 'available' functions, as well as dealing with operator precedence to decide in which order to evaluate elements; represents a useful introduction to algorithmic data structures and string parsing.

An alternative approach may be a calculator that deals with Polish Notation / Reverse Polish Notation which is perhaps an easier variant to implement, as values and operators are simply "popped" from a stack and thus may not have as many objects/classes

involved. Or, another option could be to have a more restrictive option of just plotting data via command line; the application calls the command line executable with flags to dictate plot type and the columns to use, and then plot this data via ASCII, providing a simple method of visualizing some data.

Surname	**Forename**	**Group**	**Project**
