Empirically Modelling a Retail Queuing System

0714116

Abstract

This paper investigates the use of Empirical Modelling to represent a Retail Queuing System Environment. The paper is devised as follows. Firstly, the problem of queuing systems is introduced, soon followed by an explanation of the model. We then proceed to discuss the topics of constructivism and dependencies, two important features of Empirical Modelling. With these fundamentals in place, tkeden is compared to two commercial modelling tools, SIMUL8 and SimPY. We conclude with an overview of the benefits and drawbacks of Empirical Modelling, which brings the paper to a close.

1 The Retail Queuing System

1.1 Problem Background

We have witnessed the fall of many high street retailers in the last few years. The recent recession, along with the intensely competitive nature of retail, have been significant reasons for large high street names such as Woolworths ceasing to trade.

For a high street retailer to be successful it needs to maintain both its customers and a strong brand image. Customers expect a service that is professional and efficient, with minimal queuing times.

It is queuing time that we intend to investigate by using an Empirical Model, but first we simply the problem. If a customer chooses to make a purchase at a retail store (such as Argos), they join the back of the queue. Here they wait until a cashier is available to serve them. When this occurs, the customer proceeds to the cashier desk and a transaction takes place. Upon completion of a transaction, the customer is removed from the queue.

Logically, if the rate of demand is greater than the rate of service, then a queue will form. If this continues, some customers may choose to leave the queue, perhaps expressing their frustration. On the other hand, if the rate of service largely exceeds the rate of demand, it is likely that some employees are underworked and are an expense to the business.

If we consider the above two statements, it suggests that the optimal solution is for the rate of service be equal to, or slightly higher than the current demand, so that customers can be served as quickly as possible.

1.2 Introducing the Model

Some retailers provide their customers with an estimated waiting time at particular points within the queue, perhaps with a sign that says something similar to "estimated waiting time at this point is x minutes". Although this may be a good estimate, it is calculated using statistics, rather than by observing what actually happens in a real world situation. If we use Empirical Modelling to represent the retail queuing system environment, it allows us to experiment with observable dependencies, rather than using hardcoded rules.

Although the model developed is relatively simple, it could easily be tailored for use by a specific retailer. *Figure 1* below provides an overview of the model, developed in tkeden.

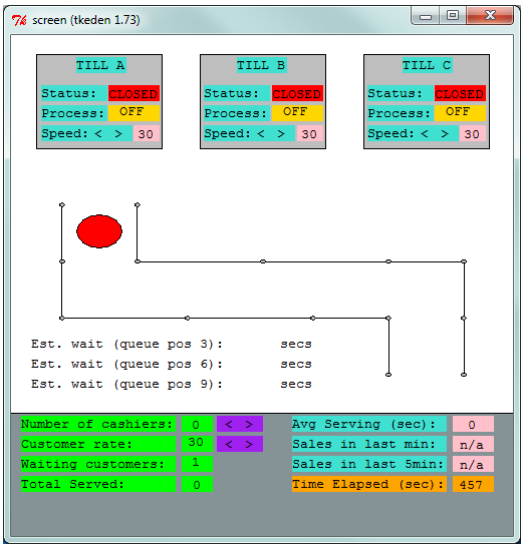

Figure 1: The Retail Queuing System Model Environment

A model such as this is of benefit to a retailer, because it enables them to experiment with observable dependencies, and make predictions from what they witness. Three possible reasons for using such a model are provided below.

1. The user may wish to calculate the **estimated waiting time** at a particular position in the queue. This information could be relayed to the customer as a means of reassurance or honesty.
2. To determine the **optimum number of cashiers** to satisfy a particular demand (through observing the dependencies of variables such as queue length, serving speed, and join rate).
3. To observe the **number of customers served** in *x* minutes (calculated by the actual number of customers served in a set timeframe, rather than a statistical calculation).

1.3 Dependencies

As we have already discussed, our model uses many observables. These are now discussed in more detail, although it is important to note that there are many more that could have been explored, resulting in a more complex model. Possible suggestions are provided in *section four*.

Number of cashiers: The number of cashiers/tills that are currently active in the model.
Till Status: Refers to the individual status of each cashier/till. There are three possible values. These are OFF (the till position does not exist), READY (the till position exists and is currently not serving) and BUSY (the till position exists and is currently serving a customer).
Transaction Speed: This is the time taken for a transaction to take place once a customer is being served.
Customer join rate: The time interval between new customers joining the queue.

When we make a change to one or more observables it has an influence on the models output. We want our Empirical Model to demonstrate similar behaviour to that of the real world. This is done through constant refinement of our model from observation of particular experiments. As a simple example, in the real world, an increase in the number of cashiers is likely to increase the number of customers served in a set period of time. We would want our model to display this same trend. If this does not occur, we would modify the models dependencies until this trend occurs.

The queuing system model returns the following output, based on the dependencies of observables previously discussed. Again, by increasing the model complexity, this would also enhance the model output.

Average Serving Time (in seconds): The average serving time of tills currently OPEN.
Number of queuing customers: The number of customers that are waiting in the queue at a given time.
Number of served customer: The number of customers that have been served since the model was launched.
Sales in last minute: The number of customers that have been served in the last minute.
Sales in the last five minutes: The number of customers that have been served in the last five minutes.
Estimated queuing time P3, P6, P9: The estimated queuing times for the customers in positions three, six and nine of the queue.

1.4 Constructivism

Part of my preparation for creating the retail queuing system model was researching queuing theory. The work of Kin Lai (2007), describes it as a "branch of probability theory". Essentially, his paper addresses various statistical methods for calculating queue times. These are such as exponential distribution (Markov) and erlangian distribution. More information on these methods is obtainable from his paper. What is interesting is that these contradict Empirical Modelling principles. This is because he considers queuing from a statistical perspective, rather than from observation.

Instead, we turn our attention to the concept of Constructivism as described by Papert (1991). He uses the term "learning-by-making". His paper explains that the best way to learn is through constructing a product, in a practical real world environment. In support of this, Beynon (2004), suggests that a "construal" is used for constructivism, rather than using a conventional program. A construal is our perception of the environment, which we can construct in the retail model through modification of observable values.

Furthermore, the constructivism approach provides us with a greater degree of flexibility. This is because we only learn the effect of a particular input through observation of its output. With this in mind, we now identify why modelling our scenario, rather than experimenting in the real world is beneficial.

1. If we choose to model the problem then we do not experience resource constraints. We can focus on the main modelling problem,

rather than having to worry about unrelated issues such as their being enough customers or enough products to sell.

2. We do not (or are unlikely to) experience real life repercussions through modelling. It allows us to experiment with observable relationships without disturbing real life customers.

3. Through simple modification to the models clock we can easily adjust the models timescale to observe a particular timeframe faster. This means experimentation can be vastly more efficient than performing real life experiments.

2 Model Construction

EDEN stands for "Engine for Definitive Notations". The retail queuing system model is written in the form of a several definitive scripts, executed by tkeden.

More specifically, the interpreter executes a *main.e* script, which triggers additional scripts. Each of these scripts contains a combination of *Eden*, *DoNaLD* and *SCOUT* notation. These scripts are such as *clock.s*, *retail.d* and *clock.s*. Multiple scripts have been used to group related code together. This was intended to make modifications easier, and also to give external users a better understanding of the codes operation.

Eden is a generalised notation, forming a basis for other specialised notations.

DONALD is a definitive notation for 2D line drawings. It has been used for the underlying queue, customer and till drawings.

SCOUT is a definitive notation for screen layout. In our retail model it has been used to define the position of windows for text and buttons.

3 Existing Modelling Software

This section of the paper introduces two alternative modelling tools, SIMUL8 and SimPY. Each of these tools is briefly described, followed by a comparison with the tkeden tool.

3.1 SIMUL8

The SIMUL8 website states that their software is intended to "take the risk out of decision making" (SIMUL8 Corp, 2011). They claim that it is now the industry standard in simulation software, with its users including British Airways, GM and Toyota.

Further investigation of this modelling tool showed that the differences between tkeden and SIMUL8 are extensive. The following numbered list provides three significant points of comparison between these software packages.

1. SIMUL8 claims to have "nearly two dozen statistical distributions" (Good Decision Partnership, 2011). I experienced that complex calculations in tkeden are difficult to perform, where as SIMUL8 is much more suited for such calculations. Furthermore, as discussed in *section 1.4*, relying on statistics contradicts with the idea of constructivism.

2. SIMUL8 enables the user to draw their model on the screen. In contrast, tkeden drawing is done within a definitive script. However, although writing a script may be more time consuming, it does give a greater degree of flexibility (in a similar way to a command line giving more control than a GUI).

3. SIMUL8 automatically collects performance data, intended for later calculations. Although it is possible to collect performance data using tkeden, we would need to include calculations within the script.

3.2 SimPY

Our second tool for comparison is SimPY (**Sim**ulation in **Py**thon). The SimPY webpage describes the tool as "an object orientated process-based discrete-event simulation language". Through reading the work of Matloff (2009), developers choose to use it due to its "clean manner" and their personal preference for the Python language. The software is currently used for traffic simulation and industrial engineering, along with numerous other uses.

In a similar fashion to *section 3.1*, we provide three points of contrast with tkeden.

1. SimPY features object orientation, where as Eden uses definitive notation. In some situations, object orientation would have been beneficial (such as the desire to create people objects that could join the retail model queue).

2. SimPY comes with "data collection capabilities, GUI and plotting packages" (SimPY Developer Team, 2010). These are features currently not provided by the Eden tools.

3. SimPY provides monitor variables to assist with generating statistics. Although it is possible to collect performance data using tkeden, we would need to include calculations within a script.

4 Extensions to the model

Given that this model is relatively small, there is certainly potential for extension. The following list describes some interesting modifications that could be made to the existing model.

1. At present, the number of potential cashiers is restricted to three. The model could allow the user to specify the number of cashiers, rather than it being limited.

2. Currently, only the speed observable controls the transaction rate. However, in reality numerous factors would determine the speed of a transaction. These may be such as the cashiers serving ability/experience, the number of items that a customer is purchasing and the type of transaction. Therefore by introducing more specific observables, the user can experiment to learn the significance of each one through observation.

3. The current model graphically shows people joining the queue, however it is difficult to uniquely identify each customer due to them all being the same colour. This problem could be resolved by either labelling, or by using a revolving colour pattern for each customer. A possible solution to this problem is demonstrated in *figure 2*.

4 Conclusions

We conclude by addressing the benefits and drawbacks of using Empirical Modelling. Some are general points of discussion, whilst others are specially related to our retail queuing system model.

4.1 Benefits of EM

Firstly, the concept of dependencies proved to be very useful during modelling. To recap, this is the idea that a variable gets updated automatically if the variables it is dependent on change. We already witnessed dependencies in some common software, such as in Spreadsheet cells, but there is still potential for further application. Take for example the installation of computer software. Some software packages require a computer reboot before becoming useable. However, with dependencies, we would see changes made automatically, meaning a restart would not be necessary.

Secondly, tkeden uses definitive notation. There are numerous benefits for using this type of notation.

- All variables containing the most up-to-date value.
- Easier interaction can occur since the response to an individual's actions will be immediate.
- It enables modification of variables at runtime, meaning the user can interact with the model and make runtime changes.

Lastly, Empirical Modelling uses the concept of iterative improvement. This is the idea that software can be developed slowly, giving opportunity for experimentation. The user is able to witness how changing variables influences the model output, and can make appropriate changes based on this.

Figure 2: A queuing model that clearly shows people moving up the queue by using colours.

4.2 Drawbacks of EM

There were some problems experienced whilst exploring Empirical Modelling in tkeden. In particular, the lack of object orientation made the concept of people joining a queue rather difficult. Ideally, I wanted customers (represented by a circle) to join the queue. However, tkeden does not allow the creation of objects whilst the model is running. I had to work around this, by adjusting the customers colour and position when they were required in the model.

Secondly, there are known problems with using definitive notation. To date, there is no commercially software available for it, meaning most that have been written are experimental. When using tkeden I did experience minor bugs and error messages that occasionally caused difficulties when working. Although I appreciate that tkeden is well developed and tested, at times solving these problems was time consuming.

It would be interesting to use Empirical Modelling to observe the behaviour of agents in more complex scenarios, (mostly to do with social interaction). However, these seemed very difficult to model in such an environment. Some of these are scenarios are given below.

- In a real world environment, customers may join the queue as groups of friends. How would this be represented in an Empirical Model? Perhaps the group only gets considered as a single entity, provided only one group member is completing a transaction.

- The range of possible interactions that can be carried out by a customer is extensive. They may choose to stop in the queue for a discussion with another customer, perhaps forget an item and have to temporarily leave the queue to collect, or stop to browse at items in a sales bin. Many of these cases are unpredictable and difficult to account for.

- There is also the problem of queue jumpers. Customers may join the queue at different positions, which would make the queue model significantly more complicated to implement.

References

Beynon. M, Roe. C. (2006). Computer Support for Constructionism in Context. *University of Warwick, Computer Science Department*. p2.

Good Decision Partnership. (2011). *Benefits.* Available: http://www.gooddecision.co.uk/benefits2.php. Last accessed 22nd Jan 2011.

Kin Lai, M. (2007). *Queueing Theory [PowerPoint Slides]*. Retrieved 24 January 2011, from University of California, Computer Science Department.

Kleinrock, L. (1975). Queuing Systems. Vol. I: Theory.

Matloff, N. (2008). Introduction to Discrete-Event Simulation and the SimPy Language, p1-33.

Papert, S. (1991). Situating Constructionism. *Constructionism*. Ablex Publishing Corporation.

SimPY Developer Team. (2010). *SimPY Overview.* Available: http://simpy.sourceforge.net/simpy_overview.htm. Last accessed 28th Jan 2010.

SIMUL8 Corporation. (2011). *Why use SIMUL8?.* Available: http://www.simul8.com/products/evalu8/. Last accessed 22nd Jan 2011.

Vignaux, G. (2007). *The Bank: an example of SimPy Simulation.* Available: http://simpy.sourceforge.net/SimPyDocs/Tutorials/TheBankOO.html. Last accessed 24th Jan 2011.