Bucks Centre for Performing Arts:  
Online Ticketing System

By Thomas Golbourn, Student ID: 21614843.  
Harvey Rush, Student ID: 21518237

# Requirement Gathering

## Introduction

The Bucks Centre for Performing Arts would like to enable their customers to order tickets through the internet. They have presented us with a document outlining the system they want. This document contains all the known requirements of the system from a functional perspective. Some of these functions lend themselves to non-functions as well. Here we will be outlining the requirements presented in the document given, both functional and non-functional, so that we can begin analysing them.

## Capturing Requirements

In this case, most of the requirement capture has been done for us, as we have been given a very specific outline of how the software should operate in a document. However, where there are gaps, or ambiguity in the document, requirement capture techniques can be used. These techniques can be broken down into 5 categories; interview, observation, document sampling, background reading, and questionnaire.

### Background Reading

In this project background reading is not entirely needed if the project is built to be scalable to suit a large theatre organisation with multiple venues, shows, and thousands of users. However, if building it for a smaller scale, then background reading would need to be undertaken to gauge how small a scale might be acceptable. We can also use background reading as a technique to learn about existing online ticket systems for theatres. This can help us avoid obstacles others have faced in the past. A major concern about background reading is that sometimes relevant or up to date information can be difficult to find, and can even have red herrings within them. In any case due to the unfamiliarity with the organisation background reading should take place to begin to understand the fact-finding questions we’ll need to ask, and to understand the business we’ll be working with.

#### Results of Background Reading

The first part of the background reading was discovering any online presence, and how they currently book tickets through that. Their website is <http://bucksperformingarts.blogspot.co.uk/> and is quite simplistic. From this site we can see their most recent show used the Courtyard Theatre in London, but they have used Wycombe Town Hall, and multiple other venues in the past. This means we will absolutely have to accommodate multiple venues in the software. Their most recent method of ticket booking was via the Courtyard Theatre’s use of TicketWeb, with an alternative to phone and book tickets directly with the theatre. However, they have used other theatre’s own payment methods, such as Norden Farm theatre’s website, in the past. This could be the reason why they need their own system, as the inconsistent booking methods are not working for them.

### Interviewing

In this project interviewing is a readily available tool, as there are regular meetings with the client, who is also well-versed in computer science and programming. With just one point of contact there may be bias, but at least there shouldn’t be any conflicting accounts. Under normal circumstances it would be costly and time-consuming to conduct the interviews and transcribe the audio or compile the notes in a meaningful and useful way. However due to the regularity of the meetings, the knowledge of the client, and the university funding them, these disadvantages do not apply quite as much. Due to this interviewing is one of the most powerful tools at our disposal in this project.

#### Interview Questions & Answers

Interview questions are shown in bold italics as headings with the paraphrased and reformatted answer indented afterwards. These were paraphrased and written down on-the-fly and agreed upon with the client during the interview as acceptable answers to the questions.

##### The document isn’t clear on which ‘customer information’ an agent must provide to purchase a ticket on a customer’s behalf, could you clarify which information should be given and how consent should be handled?

Consent from the user will be asked for when signing up, prior to entering credit card or personal information. This consent will explicitly state that agents may book on their behalf at their request whenever the requested tickets become available. Furthermore, it should state that the payment will be automated and can be subject to a processing time leading up to and including within 24 hours of the event, starting from the payment date. An agent needs to be able to find the customer in the system via any identifier or search term, such as the customer’s name or address. A unique identifier such as an email address, customer id number, or username, should also be available to use. Once an agent has identified the customer they should be able to make the purchase for them without knowing any further details about the customer. The customer should then be notified of this via email.

##### The document has no mention of target operating systems or hardware, do you have a preference? Are there hardware or software considerations on the server end we need to account for, or systems in place we need to integrate with?

The target operating system for the customer, venue manager, and agent is Windows 7, Windows 8, and Windows 10. The server should be hosted on a Windows Server 2012 machine. Payment systems will be set-up via a phoneline connected specialist machine. This machine has a local address on the network with the server and takes inputs at a rate of up to 10,000 per second in the following format:   
 [Unique ID (Int), CardNumber (Int), CVC Code (Int), Expiry (Int (MMYY), Charge (Float (####.##))]  
These inputs simply need to be parsed to a plain text file, line by line, on the server machine, in a windows network accessible location. The specialist card machine will read the inputs from there once it has been configured by the BCPA staff. Once it is running it will generate a file named ‘output.txt’ and a file named ‘log.txt’ in the same directory as the input file. Within the output file the results of operations parsed to input will be shown in the following format: [Unique ID (Int), DateTime (DateTime), Status (String)]  
The log file will show these inputs and outputs in this format: [[InputData], [OutputData]]  
The email system should use a text file in this same location called ‘email.txt’. This should be fed data line by line, after each line is read it will be deleted, the format specified is:  
[Email Address (Str), First Name (Str), Last Name (Str), Ticket Info (Any Multi-Line String)), MessageTemplateName (Str (Available Types: ‘PurchaseConfirmed’, ‘PurchaseProcessed’))]

##### Roughly how many employees does the BCPA have, and expect to have during the software’s expected lifecycle?

Less than 50 on both accounts.

##### Roughly what peak number of concurrent users do you think the software see during it’s expected lifecycle?

Between 100 and 600 concurrent users.

##### Who will be the Data Protection Officer should we need one, and will they ensure and finance the projects adherence to all current and future data privacy and e-commerce laws?

Yes, we will cover the cost of adherence to all data privacy and e-commerce law during the software’s development and lifecycle, and I am already the appointed Data Protection Officer for this software.

#### Interview Conclusions

The key take-aways are: Mail and credit card charging systems are easy to use and access. Data privacy and e-commerce law adherence will be overseen by the client. The software should handle over 600 users at once.

## Non-Functional Requirements

Non-functional requirements are requirements that are concerned with system design rather than use. For example, the use of the ticket system is to enable customer purchases, however, the design of the system also determines several key parts of the program. Here I have created a list of non-functional requirements that should be considered, with the reasons why.

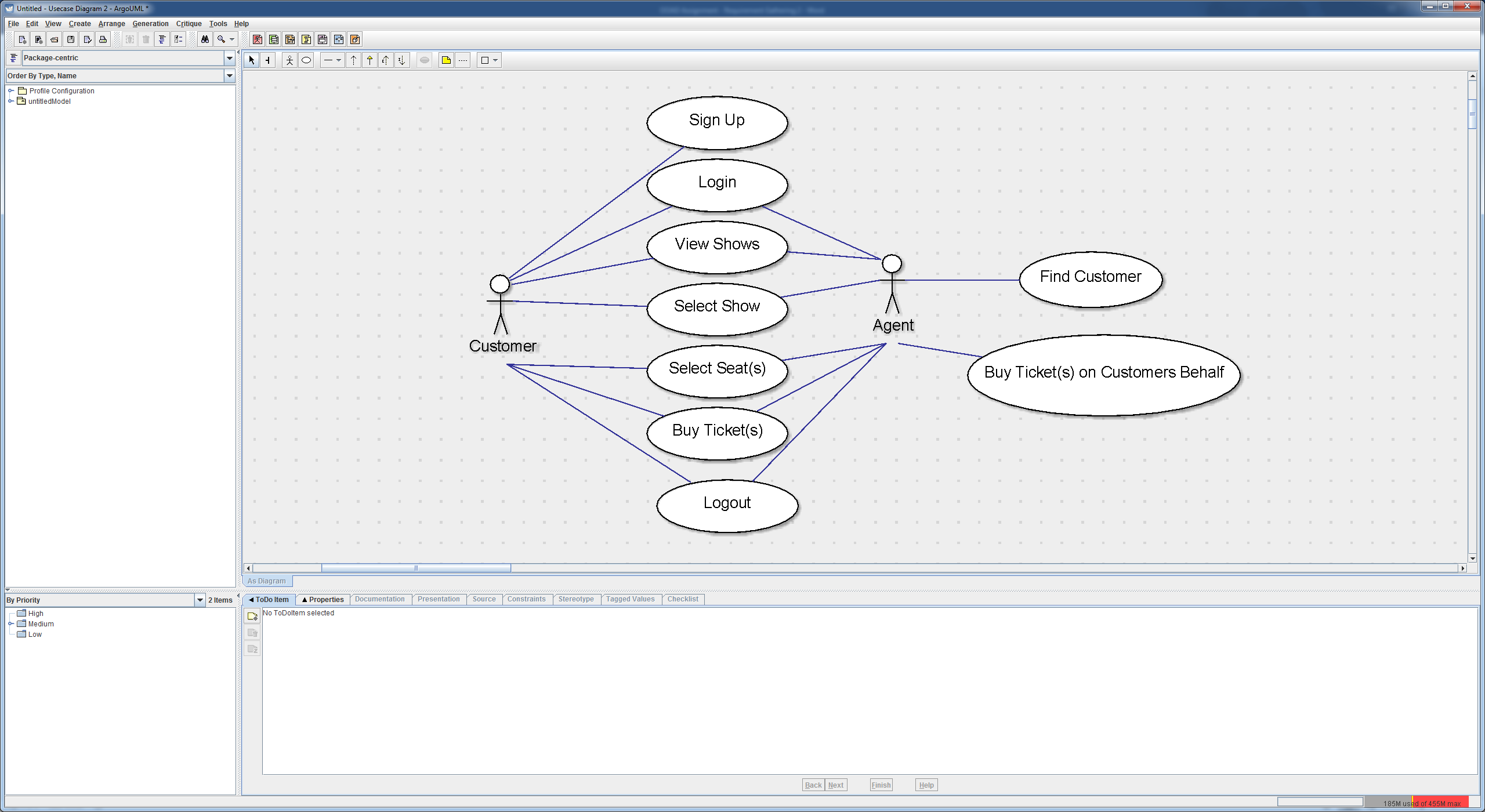
1. **Security**: Since the program requires users to create accounts with personal and financial details, and since agents and venue managers will be able to access these details, security measures are needed. If an unauthorised user managed to access one customer, or several customer accounts via an agent or venue manager account, the result would be a disaster. Customers would no longer trust the software with their financial or personal details.
2. **Quality**: If the program is of low quality, then users may lament having to use the program and call for its replacement. Customers especially may not trust the program with their information if it appears the program is of inadequate quality.
3. **Usability:** If the software is difficult to use for any user, they will be dissatisfied with it. In the venue manager’s case, where there are a lot of ways to configure a show, usability should be considered strongly. If it is not considered, then the program could easily end up feeling ‘bloated’, inundated with buttons and text, and hard to understand.
4. **Privacy:** Since Britain is still currently operating under the European Union law, the European Union’s laws must be considered. Some of these laws pertain to data privacy, control, and processing, this concerns how data can be collected, stored, used, and transported. Compliance with such laws, directives, and regulations, should be ensured during the design, build, and use of the software. Britain also uses its own laws, or clauses within EU laws, pertaining to data privacy, storage, transportation, and processing, that must be considered alongside EU law. In May of 2018 new EU legislation is coming into practice, this will also need to be considered and adhered to, along with any laws introduced by Britain following Brexit.
5. **Accountability:** Since the requirements dictate that the agent is allowed to make purchases on a customer’s behalf, actions taken within the software should trigger mechanisms that record or report such actions. This is to make the software safe to use in an online environment. If a security breach, error, or user mistake occurred and there was no accountability then discovering where, when, and how it occurred could be difficult.
6. **Configurability:** Since the software outline document does not include any seating layouts or price structures to work with, and theatres can vary these between shows, we need to ensure that the software is configurable to meet the theatre’s unique properties. These configurations will not be set-up by a software engineer, but instead by the venue manager. As such the software needs to be easily configurable on the user-level.
7. **Reliability:** If the software is unreliable, a multitude of problems can occur. Tickets may not go on sale at the right time, or the right price, seats may be oversold, users may not be able to access or use parts of or the entire system. These will hurt the business and the consumers trust in the business. This will have a ripple effect, and our software solutions may no longer be as trusted by other businesses, which could lose us clients. As such reliability is incredibly important. If a lift breaks half the times it is used, the stairs will be taken instead, regardless of how good the lift is.

## Requirements Model

### Basic Use Cases

|  |  |
| --- | --- |
| Actors | Descriptions |
| Venue Manager | The venue manager is responsible for the creation and editing of shows, including their prices, seating layouts, promotions, and discounts. They are also responsible for processing any refunds or amending tickets, as well as management of agent accounts. |
| Agent | The agent is responsible for booking tickets on behalf of customers. |
| Customer | The customer is able to access shows, select seats, and purchase tickets. |

#### Customer and Agent



#### Venue Manager

