

**Object Orientated  
Analysis & Design  
For the Addition of:  
  
 “Some System Fix/Addition”  
To the Existing Model**

**Feb-2015**

**Module:** CS4125 Systems Analysis & Design

**Authors:** Jeff McDarby, 09006076  
 Ross Tierney 12045284  
 Cormac Stone 1202724

PAGE INTENTIONALLY LEFT BLANK

**TABLE OF CONTENTS PAGE**

**SECTION 1: INTRODUCTION**

Marking Scheme o

Narrative Description o

OOAD - Project Description o

Project Plan & Allocation of Roles o

Software Lifecycle Model (SLC)  
 Waterfall Model o  
 Rapid Application Development (RAD) o  
 Agile Model o

**SECTION 2: REQUIREMENTS**

Functional: Existent Use Case Model o

Functional: Existent Decomposed Use Case Models o

Functional: Structure Use Case Descriptions o

Non-Functional: Add-ons to Use Case Descriptions o

Tactics to Support Specific Quality Attributes o

GUI Prototypes o

Layout/Formats for Electronic/Hardcopy Reports o

Discussion of System Architecture with Package Diagram o

**SECTION 3:** **OBJECT ORIENTATED ANALYSIS AND DESIGN (OOAD)**

Data Driven Design (DDD) List of Candidate Objects o

UML Class Diagram Sketches o

Sequence/Communication Diagram o

Entity Relationship Diagram with Cardinality o

**SECTION 4: DESIGN AMENDMENTS –SAMPLE IMPLEMENTATION**

Proof of Concept (PoC) Prototype o

Class Implementations o

Model-View-Controller (MVC) Architectural Pattern o

Automated Test Cases o

table of contents cont…

**SECTION 1: INTRODUCTION**

**MARKING SCHEME**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CS4125: Systems Analysis and Design. Semester II, 2014-2015 MARKING SCHEME for Team-Based Project:** Version 1 (19th February 2015 - Week 4) | | | | | |
|  | **Item** | **Detailed Description** | **Marks Allocated** | | **Marks Awarded** |
| Sub-Total | Total |
|  | Presentation | General presentation  Adherence to guidelines e.g. front cover sheet, blank marking scheme, table of contents |  | 2 |  |
| 3 | Narrative | Narrative description of business scenario |  | 1 |  |
| 4 | SLC | Discuss and justify SLC and risk management strategy |  | 1 |  |
| 5 | Project Plan | Plan specifying timeline, deliverables and roles |  | 1 |  |
| 6 | Requirements | Use case diagram(s) Structured use case descriptions(s) NFRs - quality attributes Tactics to support quality attributes Prototypes | 2 2 1 1 1 | 7 |  |
| 7 | System Architecture | System architecture diagram with interfaces |  | 2 |  |
| 8 | Analysis | Listing of candidate classes Sketch of a class diagram with generalisation, composition, multiplicity, dialog, control, entity, interfaces, pre and post conditions, etc. Sketch of an interaction diagram Entity relation diagram with cardinality | 1 3    2 1 | 7 |  |
| 9 | Code | Compiles and runs Object orientated, interfaces, etc. MVC Automated test cases Added value:  a) Architectural/Design Pattern(s)  b) Concurrency  c) GUI | 3 2 2 5 | P/F 12 |  |
| 10 | Added Value | Two page discussion on added value |  | 2 |  |
| 11 | Design Blueprints | Architectural diagram Class diagram Interaction diagram State chart Description of patterns and approach to concurrency support | 2 2 1 2 1 | 8 |  |
| 12 | Critique | Evaluate the analysis and design artefacts |  | 2 |  |
| 14 | References |  |  | P/F |  |
| 15 | Lab Attendance | Attendance at labs (weeks 5-11) |  | 5 |  |
|  | Interview (week 13) pass/fail basis | |  | P/F |  |
| **TOTAL MARKS** | | | | **50** |  |

**NARRATIVE DESCRIPTION**

**The Prince JJ Hotel – Limerick, Ireland**

Step into a world of ease and comfort and experience  
the finest of Irish hospitality at the luxurious  
Prince JJ Hotel, located in the heart of historic Limerick within walking distance of the most  
extraordinarily breath-taking churches, castles and monuments.

Be treated to unrivalled comfort and discerning service, in a proud example of lovingly restored  
18th century paradise that offers beautifully  
appointed guestrooms and deluxe suites with  
private balconies providing stunning views  
of Limerick’s bustling cityscape.

Visit the highly acclaimed award winning, Scrumptious Ristorante, inspired by Limerick’s vibrant culture and history, offering sophisticated selections of delicious traditional and contemporary cuisine - perfect epicure for every occasion.

Overlooking the magnificent River Shannon, the hotel's luxurious 200 square meter swimming pool is an al-fresco masterpiece, nested in verdant gardens, and is maintained using Advanced Oxidation (AO) technologies, helping reduce the amount of required chlorine by 70%.

Additional amenities include high-speed Wi-Fi internet, air-conditioning, soundproofed rooms, TVs with international entertainment and news, in-room safe boxes, business desks with international power sockets, free hair dryers and ironing boards (on request), complimentary newspapers, and 24-hour concierge services, and much, much more.

Simply put, our goal is to bring you the very finest in hospitality and standards.

The Prince JJ Hotel, Limerick The Apex of Opulence

**OOAD - PROJECT DESCRIPTION**

The purpose of this Object Orientated Analysis and Design (AAOD) document is to detail enhancements to an existent System Model, as currently employed by the Prince JJ Hotel.

First, a detailed analysis of the existent model will be provided. Next, our enhancements will be detailed, including sample code implementations.

**PROJECT PLAN AND ALLOCATION OF ROLES**

For the duration of the completion of this OOAD document, version control was implemented via <http://github.com/cs4125> and GROUP activities were co-ordinated via a team scrum board located at <http://scrumblr.ca/cs4125>.

The software framework used to offer basic support for requirements traceability, automated documentation generation and round-trip engineering of structural code was Visual Paradigm <http://www.visual-paradigm.com/>

A text based activity log, maintained to record progress, and was updated after every team meeting.

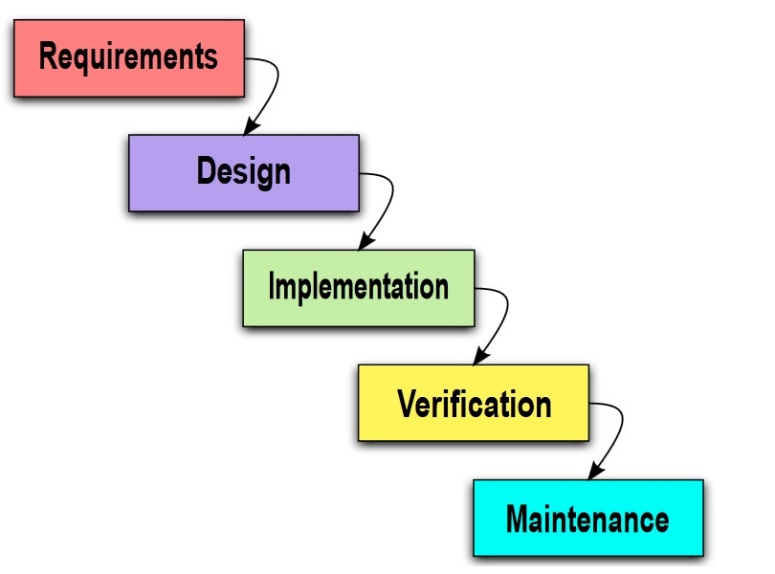
|  |  |  |
| --- | --- | --- |
| **ROLE** | **DESCRIPTION** | **MEMBER** |
| GROUP | All team members | All team members |
| PROJECT LEADER | Project Facilitator | Jeff McDarby |
| SOFTWARE TESTER | Facilitation of text case generation | Ross Tierney |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**SOFTWARE LIFECYCLE MODEL (SLC) WATERFALL MODEL**

**BACKGROUND**

As the Prince JJ Hotel has evolved its information systems over recent decades, various software development methodologies have been employed, each with its own strengths and weaknesses.  
A chronological summary of three of the major life cycle methodologies employed is detailed below.

**THE WATERFALL MODEL**

The Waterfall approach was first  
SDLC (Software Development Life Cycle) model to be employed by the Prince JJ Hotel to manage their Software Engineering requirements. It was employed soon after a formal description introduced the methodology in a 1970 article by Winston W. Royce, although Royce did not use the term “Waterfall” in that article.

Also referred to as a linear-sequential life cycle model, it proved simple to understand and use. The software development process was divided into distinct sequential step-by-step phases, with progress flowing from the top to the bottom, like a cascading waterfall.

Each phase must be completed before the next phase can begin, without overlapping of the phases. At the end of each phase, a review took place to determine if the project was on the right path, and whether or not to continue, restart the phase, or discard the entire project. Testing started only after the final development was complete.

*ADVANTAGES IDENTIFIED INCLUDED:*  
  
1) It was simple to implement and understand, and to manage as it provided a rigid framework, minimising the amount of resources required.

2) Output was generated after each stage, which had high visibility, giving a strong sense of progress. It was noted that psychological factors play an important role in the development process.

3) Project management was facilitated as deadlines could be set for the completion of each phase supported by scheduled evaluations.

4) The methodology proved significantly better than previous haphazard approaches, providing a template into which methods for requirement specification, design, implementation, verification (testing) and maintenance could be placed.

*DISADVANTAGES IDENTIFIED INCLUDED:*

1) In reality, projects rarely followed a predefined sequential flow. This frequently forced changes which caused confusion and diluted the model, especially when separate projects proceeded concurrently.

2) It proved difficult to identify customer requirements explicitly, and even when identified, specifications frequently changed. The non-reversible nature of the waterfall model often proved inflexible.

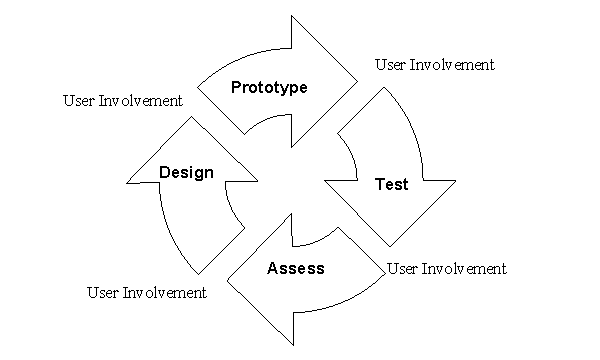
3) Reverting back a phase or two proved extremely costly as all previous work then became redundant. This proved a costly affair.

4) While the Waterfall model has proven suitable for small projects, it was felt that model did not best suit the continuous, on-going and large scale nature of operations at the Prince JJ Hotel.

**SOFTWARE LIFECYCLE MODEL (SLC) RAPID APPILCATION DEVELOPMENT (RAD)**

As the software requirements of the Prince JJ Hotel increased with stronger demands for faster software development, as the 1980’s progressed, the inadequacies of the Waterfall model became apparent and Rapid Application Development (RAD) methodologies, developed by Barry Boehm and James Martin at IBM, amongst others, were adopted in its place.

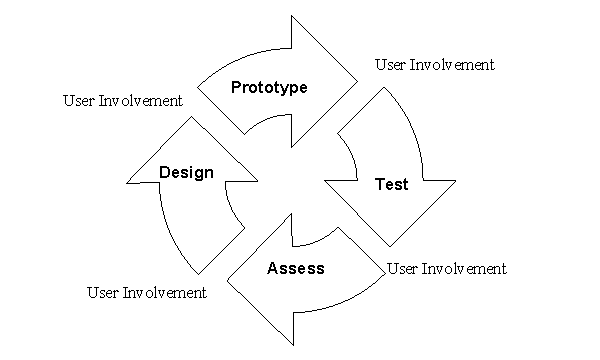
After a quick requirements gathering phase, prototype application were built and presented to hotel management. Various components or functions were developed in parallel, as if each was a separate mini-project. Feedback from hotel management then provided a loop to improve or add functionality. Each component was time boxed, delivered and then assembled into a working prototype. The concurrent and looping nature of this model can be conceptualised as a spiral.



**Component 1**

**Team 1**

**Month 1**



**Component 2**

**Team 2**

**Month 1**

*ADVANTAGES IDENTIFIED INCLUDED:*

1. A reduction in development time was noted.
2. Components proved reusable.
3. Concurrent, early component feedback occurred.
4. Early prototyping minimised integration and maintenance issues.

*DIS-ADVANTAGES IDENTIFIED INCLUDED:*

1. Some of the hotels requirements could not be clearly modularized to suit RAD.
2. A high dependency on modeling skills proved a resource challenge.
3. The RAD process proved inappropriate for small scale projects as the cost of modeling  
   was relatively high.
4. The pre-requisite of continious cohesive, highly motivated team dynamics proved a barrier.

**SOFTWARE LIFECYCLE MODEL (SLC) AGILE MODEL**

In approximately 2005, the Prince JJ Hotel switched to the Agile Model as defined via the “Manifesto for Agile Software Development” and it’s “Twelve Principles of Agile Software”, set in February 2001, when 17 software developers met at the Snowbird resort in Utah to discuss lightweight development methods.

**The manifesto dictates (direct extract):**

“We are uncovering better ways of developing software by doing it and helping others do it.  
Through this work we have come to value:

* Individuals and interactions over processes and tools.
* Working software over comprehensive documentation.
* Customer collaboration over contract negotiation.
* Responding to change over following a plan.

That is, while there is value in the items on the right, we value the items on the left more.”

**The 12 Principles of Agile as defined in 2001 are:**

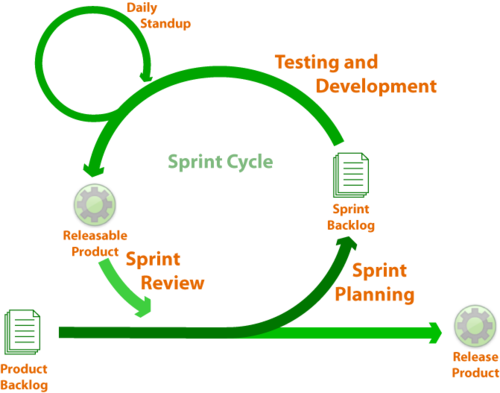
1. Our highest priority is to satisfy the customer through early  
   and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development.  
   Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months,  
   with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

**SCRUM**

The process employed by the Prince JJ Hotel to achieve its agile software development objectives centres on daily team-based scrums designed to provide the flexibility to achieve objectives, set as user story points, within a specified time period, aka the “burndown” of a “sprint”.

Scrum predates Agile, having being developed by Hirotaka Takeuchi and Ikujiro Nonaka in the "New New Product Development Game", published in 1986. However their model proved highly suitable for use in Agile Software Development.

**The current model employed by the hotel is illustrated below:**



**SCRUM ROLES: PRODUCT OWNER SCRUM MASTER DEVELOPMENT TEAM**

*SUMMARY OF THE SCRUM PROCESS EMPLOYED BY THE PRINCE JJ HOTEL*

The various, distinct software development programmes required by the Prince JJ Hotel are each assigned a “Product Owner” who is responsible for achieving the individual tasks that constitute the programme.

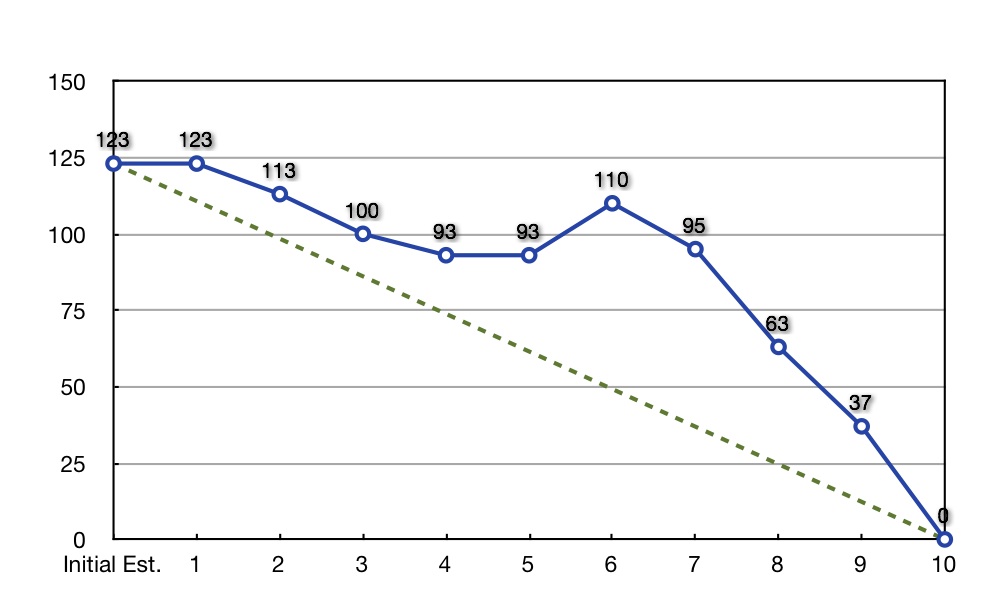
The “Product Owner” breaks the programme into smaller component parts called “Epics”. Each of these is then further divided into logical “User Stories” which are prioritised through the allocation of numeric “Story Points” for each “User Story” for each “Epic”. This is the generation of what is called the “Product Backlog”, which the “Product Owner” manages.

A separate and similar prioritisation of the larger “Epics” by the “Product Owner” allows for clearer prioritisation of software development goals, which must be achieved within a set time period, known as a “Sprint”. This process is known as “Sprint Planning”. A series of “Sprints” then forms a “Release” for the software programme.

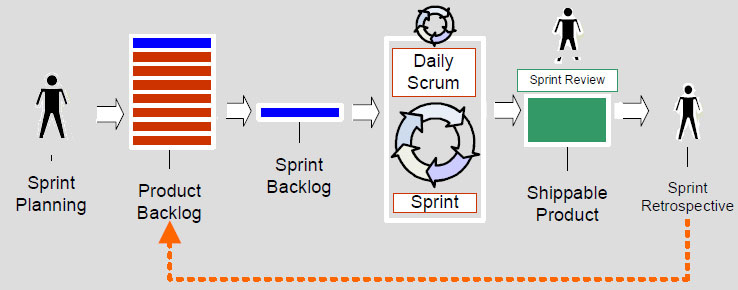
“User Stories” for a “Sprint” are then assigned to development teams, complete with their “Story Point” numeric values. However, the “Product Owner” does not become involved in “Sprint” activities or people-management.

The “Product Owner” passes the “User Stories” to be achieved within the time frame of a “Sprint” to the “Scrum Master”, who can be viewed as a facilitator for his software development team, as opposed to a feudalistic, top-down manager.

At this point, the “Development Team” take full responsibility for achieving the “Burndown” of their “Story Points” within the set “Sprint” period. They alone manage the allocation of the required tasks within their team in the form of “User Stories” normally via a “Scrum Board” (illustrated below), which can be further sub-divided into “Sub-tasks”. Progress can then be monitored via the use of “Burndown Charts” which represent work remaining over time for a set “Sprint”.



*Sample burndown chart showing story points on the y axis and sprint time in days on the x axis*

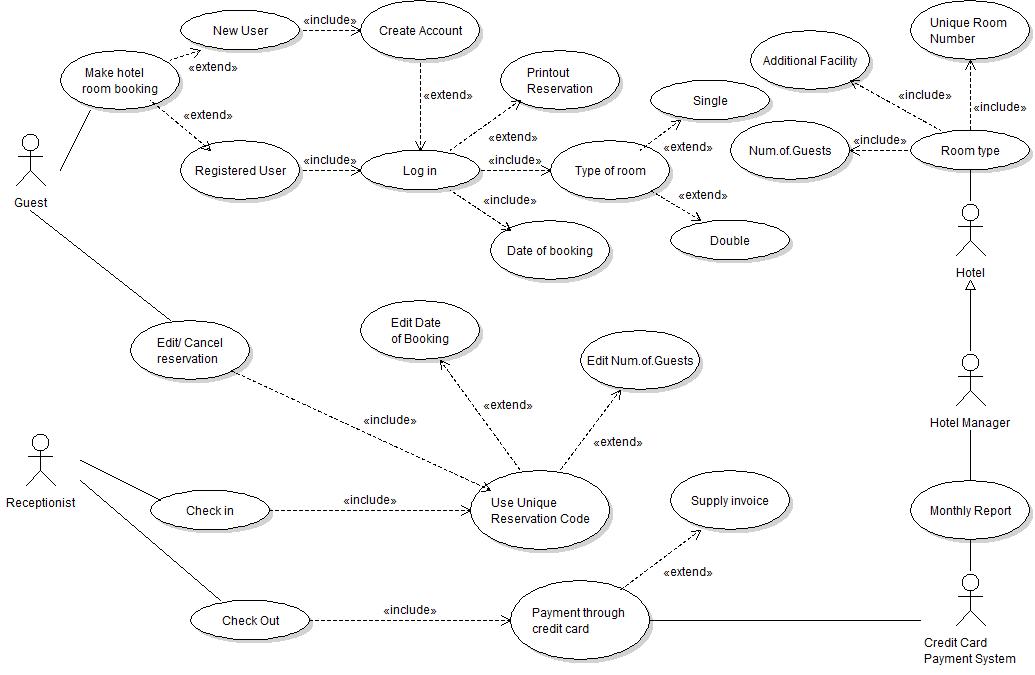


*Illustration of the overall Agile process*



*Illustration of a development teams scrum board*

**SECTION 2: REQUIREMENTS**

**Functional: Existent Use Case Model**

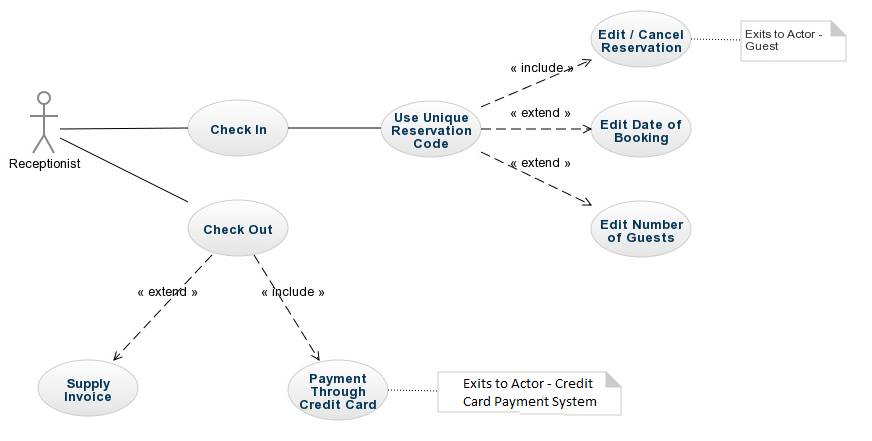
*Diagram 1 – Existent Model Use Case*

The Prince JJ Hotel currently employs the above System Model, which offers broad scope for improvement.

Five unique Actors currently service hotel requirements:

* **GUEST ACTOR**  
  A paying and staying visitor at the Prince JJ Hotel who can make or cancel a reservation.  
  This may be a new user, unique to our System, or an existing pre-registered user. If they are a new user, the system must create a new account, otherwise an account will pre- exist. The GUEST can then be logged in to the System, and a booking date can be assigned for either a single or a double room. A printable copy of the reservation becomes available at this point.
* **RECEPTIONIST**A member of staff who checks in or checks out a logged in, visiting guest who has a unique reservation code. Amendments to the date of the booking and to the number of guests can be made at this point, if the GUEST wishes to edit their reservation. They may also cancel their reservation entirely. However, the system does not currently include a facility for changing room types. A check out occurs through payment by credit card for which a printable invoice can be supplied.
* **CREDIT CARD PAYMENT SYSTEM**The only functionality modelled for this facility is the generation of a monthly report. Payment processing is not comprehensively modelled in the existent System design.
* **HOTEL MANAGER ACTOR**Hotel Manager is not comprehensively modelled in the existent System design.
* **HOTEL ACTOR**This function can be viewed in terms of facilities management. Under the guidance of the HOTEL MANAGER, the HOTEL consists of rooms of a defined type, and each has a unique room number. The number of GUESTS per room and details regarding any additional facilities are included at this point, on a room by room basis. One could guess additional facilities includes kitchens, cleaning rooms, store rooms, hallways, etc. Additional facilities are not comprehensively modelled in the existent System design.

**Functional: Existent Decomposed Use Case Models**



*Diagram 2 – Receptionist Actor*

**Functional: Structure Use Case Descriptions**

**Non-Functional: Add-ons to Use Case Descriptions**

**Tactics to Support Specific Quality Attributes**

**GUI Prototypes**

**Layout/Formats for Electronic/Hardcopy Reports**

**Discussion of System Architecture with Package Diagram**

**SECTION 3: OBJECT ORIENTATED  
ANALYSIS AND DESIGN (OOAD)**

**Data Driven Design (DDD) List of Candidate Objects**

**UML Class Diagram Sketches**

**Sequence/Communication Diagram**

**Entity Relationship Diagram with Cardinality**

**SECTION 4: DESIGN AMENDMENTS –  
SAMPLE IMPLEMENTATION**

Proof of Concept (PoC) prototype

Class Implementations

Model-View-Controller (MVC) Architectural Pattern

Automated Test Cases

**SOURCES & REFERENCES**

|  |  |  |
| --- | --- | --- |
| **NUMBER** | **PAGE** | **SOURCE** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

PAGE INTENTIONALLY LEFT BLANK