2014

Software Test Specification (STS)

CMIS330

IAN NELSON, CHRISTY GILL, ED RUSSELL, HECTOR SANTIAGO, AND JESSE WASILEWSKI

DEVELOPED FOR **JENEO K. DURHAM**

Table of Contents

1. Introduction

1.1 Objectives

The objective of this document is to provide a testing framework for the three primary levels of the Bed and Breakfast Management System (BBMS). This framework provides a testing team the ability to perform black-box (component requirements) and white-box (algorithmic) tests against the software and validate both requirements and design.

1.2 Background

The BBMS is used to assist management of small bed and breakfast establishments with customer reservations and to ease the overhead of managing different sets of customers. By developing software to automate many of these repetitive tasks, higher efficiencies can be achieved and the organization of customers against a calendar of reservations can increase revenue. This document and subsequent testing framework are used to assist in the development and efficiency of development to provide a software product to customers that provides this service.

1.3 Scope

This testing documentation is scoped to the test cases specific for the three modules of the architectural context diagram:

1. User Interface – Section 2.1
2. Services – Section 2.2
3. Domain Objects – Section 2.3

Other services in the system, such as data persistence and database configuration, are out of scope for this document.

1.4 References

References for the software test specification include:

1. IEEE std. 829-1998 – IEEE Standard for Software Test Documentation
2. Software Requirements Specification (SRS) for BBMS
3. Software Design Document (SDD) for BBMS

1.5 Test Environment

The testing environment for the BBMS is relatively light in comparison to other systems. Since this software has limited requirements and interaction with other components, a simple environment for testers should be relatively easy to accommodate.

1.5.1 Software

The BBMS can run on either Linux or Windows operating systems. As a result, it is expected that the computer(s) provided for testing are running either of these operating systems in a fully usable fashion. In addition, the BBMS software should be installed on the operating system and verified for use.

1.5.2 Hardware

A modern computer (Core Duo or above) with usable RAM (1GB or more) and physical storage (at least 500MB post-operating system install) will accommodate the BBMS software. The computer must also include a mouse, keyboard, and display at 1280x1024 for optimal results viewing the software during testing.

1.5.3 Communications

As specified in section 3.1.4 of the SRS, external communications requirements are not applicable (N/A).

1.5.4 Tools

Testing of the BBMS should consist of a usable computer environment (defined in sections 1.5.1 and 1.5.2). Since the API and domain object layers of the software are not designed to be public, their access is limited to testers. As a result, the primary method of testing is the BBMS user interface itself. To test data entry into the database across the three modules, a MySQL tool, such as MySQL Admin or other equivalent RDBMS management tool can be used. The tester should have full administrative rights to the testing database in order to access both the database and fields in which the domain objects are stored.

1.5.5 Data

Testers of the BBMS should be prepared with the following data in order to complete the use cases in section 4 of the STS:

1. Date ranges for the calendar
    a. Non-overlapping ranges for successful reservations
    b. Overlapping ranges for unsuccessful reservations and checks.
2. Customer data
    a. Test names (first, last)
    b. Test addresses (street, state and zipcode)
    c. Dummy credit card information (valid and invalid formats)
    d. Dummy phone numbers (valid and invalid)
3. Reservation data
    a. Sample prices per day (valid and invalid)
    b. Guarantee dates for a guest
4. Establishment Data
    a. Set of room numbers to create for vacancies

The set of data listed above will accommodate the entirety of test cases across the user interface, service, and domain object cases listed in this document.

2. Architectural Context Diagram Mappings

2.1 User Interface

This module is for users of the system, such as managers of a bed and breakfast. They are able to use the system via this interface exclusively. Access to this user interface does not require any external interface, other than via the keyboard and mouse, and is a single-user set of views.

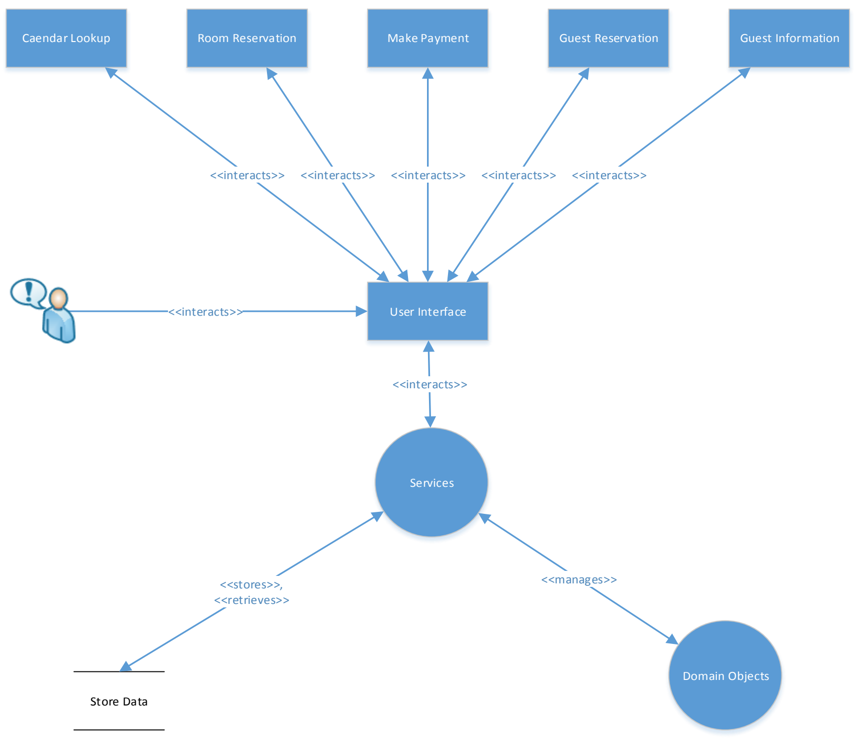

Figure 1. Architectural Context Diagram – User Interface

2.2 Services

This module is used to house the business logic for processes that the BBMS needs to conduct where domain objects are involved, such as looking up a calendar date or saving a payment. The service module provides output to the user interface, and input from the user interface is used to manage domain objects where appropriate. Persistence of the domain objects to the database is also managed by the service component.

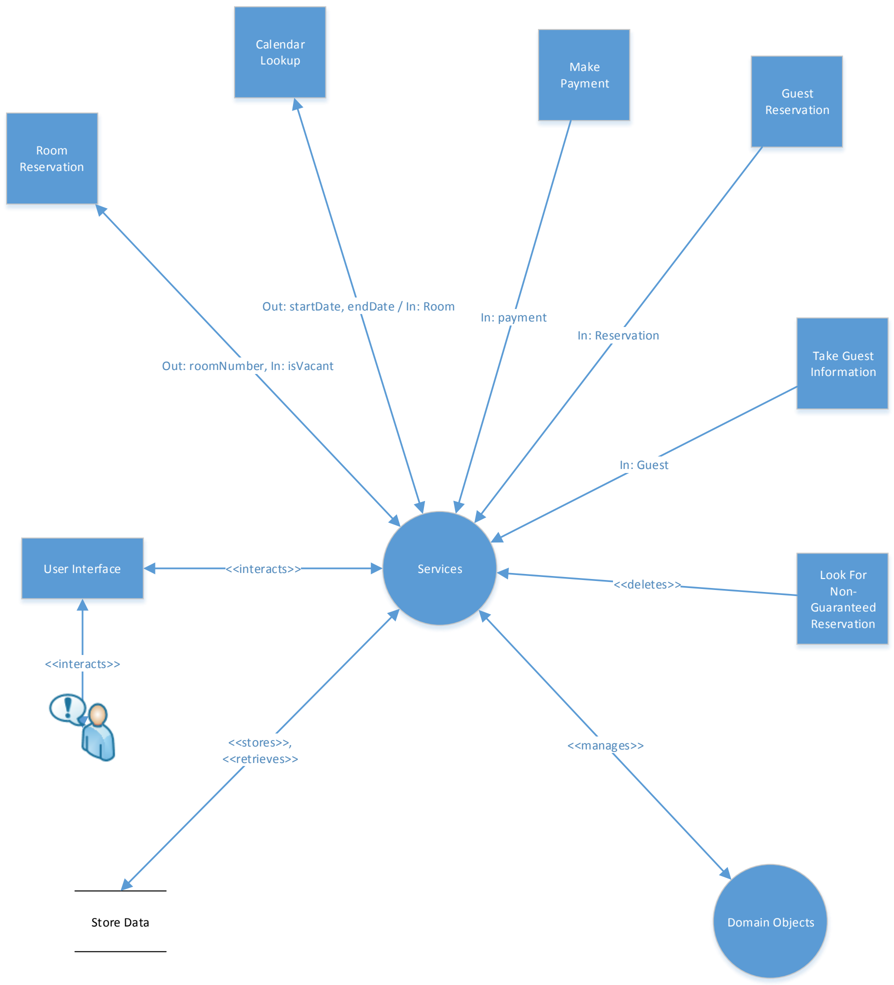

Figure 2. Architectural Context Diagram – Services

2.3 Domain Objects

This module covers all entities that hold data about a particular function in the BBMS system. These objects are referred to as domain objects.

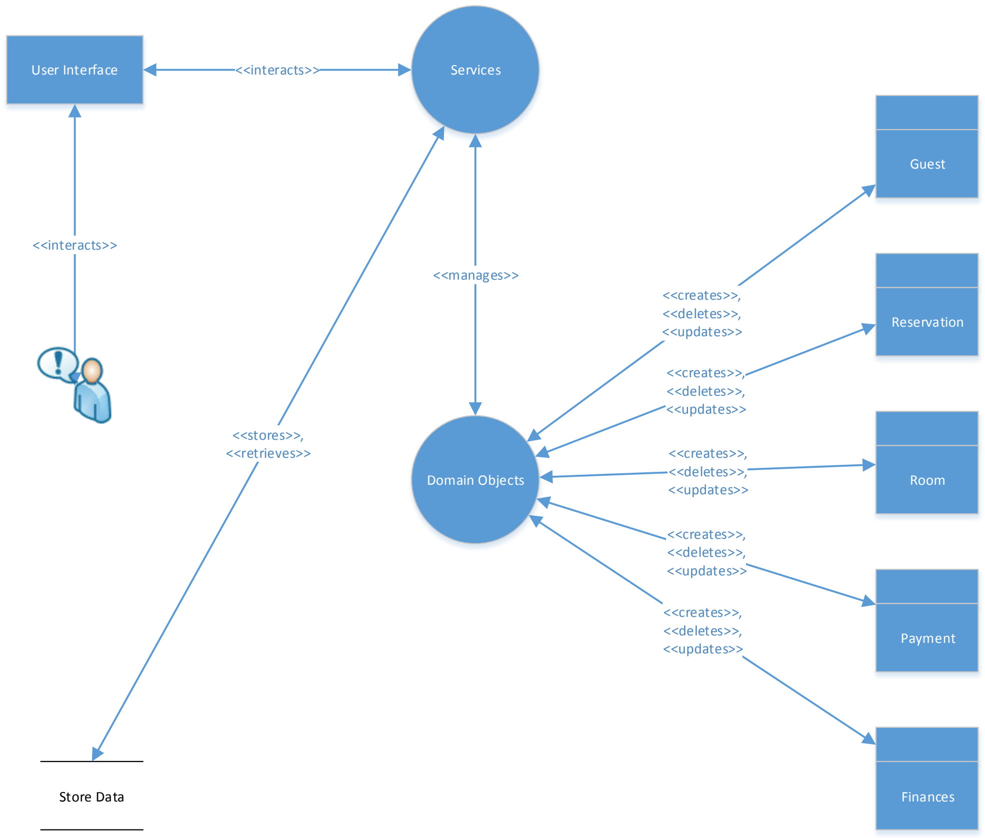

Figure 3. Architectural Context Diagram – Domain Objects

3. Traceability Matrix

The traceability matrix provides a linkage between the initial software requirements (SRS) and software design (SDD) to the testing (STS) use cases defined below. Additionally, the matrix provides accountability of the testing framework for pass/fail completion of each test.

Category	Description	System Req.	Use Case	Software Req.	Test Case	Pass/Fail
User Interface	Calendar	3.1.1.2	App. F, Case 1, S#1	5.1	4.1.1	
User Interface	Room Reservation	3.1.1.3	App. F, Case 2, S#1	5.2	4.1.2	
User Interface	Payment	N/A	App. F, Case 2, S#2	5.3	4.1.3	
User Interface	Guest Reservation	3.1.1.3	App. F, Case 2, S#1	5.4	4.1.4	
User Interface	Guest Information	3.1.1.3	App. F, Case 2, S#1	5.4	4.1.5	
User Interface	Reservation (Create)	3.1.1.3	App. F, Case 2, S#1	5.2	4.1.6	
User Interface	Reservation (Update)	3.1.1.3	N/A	5.2	4.1.7	
User Interface	Reservation (Delete)	3.1.1.3	N/A	5.2	4.1.8	
Services	assignReservation	3.2.2	N/A	6.1.1	4.2.1	
Services	takeGuestInfo	3.2.2	N/A	6.1.5	4.2.2	
Services	takeGuestInfo	3.2.2	N/A	6.1.5	4.2.3	
Services	makePayment	3.2.4	N/A	6.1.3	4.2.4	
Services	makePayment	3.2.4	N/A	6.1.3	4.2.5	
Services	createReservation	3.2.2	N/A	6.1.4	4.2.6	
Services	deleteReservation	3.2.2	N/A	6.1.4	4.2.7	
Services	createRoom	3.2.3	N/A	6.1.1	4.2.8	
Services	deleteRoom	3.2.3	N/A	6.1.1	4.2.9	
Domain Objects	Validate Guest (name)	3.2.1	App. F, Case 2, S#1	6.2.1	4.3.1	
Domain Objects	Validate Guest (Credit Card)	3.2.1	App. F, Case 2, S#1	6.2.1	4.3.2	
Domain Objects	Invalidate Guest (name)	3.2.1	App. F, Case 2, S#2	6.2.1	4.3.3	
Domain Objects	Invalidate Guest (Credit Card)	3.2.1	App. F, Case 2, S#2	6.2.1	4.3.4	
Domain Objects	Validate Guarantee	3.2.3	App. F, Case 2, S#1	6.2.3	4.3.5	

4. Test Case Specifications

Test cases are provided for the three core modules of the BBMS, covered in section 2 of this document. These tests constitute the completion of steps to validate both inventory and algorithmic design (white-box), as well as requirements-bound (black-box) processes. Overall, the test case specifications are a derivative of both the SRS and SDD documents for the provided for the BBMS engineering process.

4.1 User Interface Test Cases

These cases are designed for interactive testing of the BBMS through the user interface (UI), and require a testing team to verify.

4.1.1 Specification ID: BBMS-UI-01

Objective: The objective of this test case is to provide a black-box test for the Calendar portion of the user interface as required by section 3.1.1 of the SRS.

Test Items: This test will emulate a manager utilizing the Calendar, Room Vacancy Status and Make Reservation capabilities in the CalendarLookup Entity.

Input Specifications

1. Click on a day within the calendar view
2. Click on the arrow on the upper left-hand side of the calendar view
3. Click on the arrow on the upper right-hand side of the calendar view
4. Click on the button on the lower left-hand side of the calendar view
5. Enter 1 July 2015
6. Click on the button on the lower right-hand side of the calendar view
7. Enter 10 July 2015
8. Click on 4 July 2015
9. Click on each Room option available in the upper right-hand panel
10. Click Make Reservation button

Output Specifications

1. Calendar view highlights day selected
2. Calendar displays previous month view
3. Calendar displays the next month view
4. Starting date option menu appears
5. Calendar view begins with 1 July 2015
6. Ending date option menu appears
7. Calendar view ends with 10 July 2015
8. Calendar view highlights 4 July 2015
9. Panel highlights each Room as it is chosen
10. Panel button reflects graphical response to action and initiates Room Reservation UI

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse

4. Keyboard
5. Monitor
6. Operating system with BBMS installed.

Special Procedural Requirements

BBMS services will require manager's permissions starting in the CalendarLookup Entity.

Inter-case Dependencies

None

4.1.2 Specification ID: BBMS-UI-02

Objective: The objective of this test case is to provide a black-box test for the RoomReservation portion of the user interface as required by section 3.1.1 of the SRS.

Test Items: This test will emulate a manager utilizing the RoomReservation Entity to create and update vacancy statuses.

Input Specifications

1. Click Vacant checkbox to reserve the room for the 4 July 2015 reservation
2. Click Vacant checkbox again to remove reservation
3. Click Vacant checkbox again to reserve the room for the 4 July 2015 reservation
4. Click Save Room

Output Specifications

1. Checkmark appears on the Vacant checkbox
2. Checkmark disappears from the Vacant checkbox
3. Checkmark reappears on the Vacant checkbox
4. Panel button reflects graphical response to action and initiates Calendar Lookup UI

Environmental Needs

The hardware requirements are for a BBMS interface client with keyboard and mouse and software requirements are BBMS services operating.

Special Procedural Requirements

BBMS services will require manager's permissions starting in the RoomReservation Entity.

Inter-case Dependencies

BBMS-UI-01 must be completed first in order for the BBMS to initiate the reservation workflow; this test case will reflect the Room information from the BBMS-UI-01 test case in the form of Room Number text box being populated with the chosen Room to confirm vacancy assessment operated correctly.

4.1.3 Specification ID: BBMS-UI-03

Objective: The objective of this test case is to provide a black-box test for the MakePayment portion of the user interface as required by section 3.1.1 of the SRS.

Test Items: This test will emulate a manager utilizing the MakePayment Entity for receiving a payment and adjusting a Finances balance for a chosen guest.

Input Specifications

1. Click on Guest text box
2. Click through each guest name
3. Chose the last name that appears on the guest list
4. Click on Total data entry box
5. Enter the value $1776.00
6. Click on Payment data entry box
7. Enter the value $1776.00
8. Click Make Payment button

Output Specifications

1. Text box should populate with guest names
2. Text box should highlight each name as it is chosen
3. Guest name should remain highlighted throughout the test
4. Cursor should appear within the data entry box
5. The data entry box should reflect $1776.00 value entered and remain throughout the test
6. Cursor should appear within the data entry box
7. The data entry box should reflect $1776.00 value entered and remain throughout the test
8. Panel button reflects graphical response to action and initiates Calendar Lookup UI

Environmental Needs

7. Sufficient hard disk space to store increasing amounts of data.
8. Working computer with a processor capable of executing multiple simultaneous commands.
9. Mouse
10. Keyboard
11. Monitor
12. Operating system with BBMS installed.

Special Procedural Requirements

BBMS services will require manager's permissions starting in the MakePayment Entity.

Inter-case Dependencies

BBMS-UI-01 and BBMS-UI-02 must be completed first in order for the manager to substantiate a payment. BBMS-UI-04 must be completed to initiate a payment workflow.

4.1.4 Specification ID: BBMS-UI-04

Objective: The objective of this test case is to provide a black-box test for the GuestReservation portion of the user interface as required by section 3.1.1 of the SRS.

Test Items: This test will emulate a manager utilizing the GuestReservation Entity for inputting specific information for a guest's stay.

Input Specifications

1. Click on the calendar button to the right of Reservation Start data entry box
2. Choose 4 July 2015 from pop-up calendar
3. Click on the calendar button to the right of Reservation End data entry box
4. Choose 7 July 2015 from pop-up calendar
5. Click on Price Per Day data entry box
6. Enter $444.00
7. Click Guaranteed checkbox
8. Click Save Reservation

Output Specifications

1. Reservation start date option menu appears
2. Reservation Start reflects 4 July 2015
3. Reservation end date option menu appears
4. Reservation End reflects 7 July 2015
5. Cursor should appear within the *Price Per Da*y data entry box
6. $444.00 should appear in the *Price Per Day* data entry box and remain throughout this test
7. Checkmark should appear and remain in the *Guaranteed* checkbox
8. Save Reservation button reflects graphical response to action

Environmental Needs

13. Sufficient hard disk space to store increasing amounts of data.
14. Working computer with a processor capable of executing multiple simultaneous commands.
15. Mouse
16. Keyboard
17. Monitor
18. Operating system with BBMS installed.

Special Procedural Requirements

BBMS services will require manager's permissions starting in the MakePayment Entity.

Inter-case Dependencies

BBMS-UI-01 must be completed first in order for the manager to initiate a workflow.

4.1.5 Specification ID: BBMS-UI-05

Objective: The objective of this test case is to provide a black-box test for the GuestInformation portion of the user interface as required by section 3.1.1 of the SRS.

Test Items: This test will emulate a manager utilizing the GuestInformation Entity for inputting specific information for a guest.

Input Specifications

1. Click the Guest Information button on the Guest Reservation panel
2. Click on First Name data entry box
3. Enter “John”
4. Click on Last Name data entry box
5. Enter “Smith”
6. Click on Address data entry box
7. Enter “520 Chestnut Street, Philadelphia, PA”
8. Click on Phone Number data entry box
9. Enter “877-444-6777”
10. Click on Credit Card Number data entry box
11. Enter “1111222233334444”
12. Click Save Guest button

Output Specifications

1. Guest Information button reflects graphical response to action and Guest Information UI appears
2. Cursor appears in First Name data entry box
3. “John” appears in First Name data entry box and remains throughout the test
4. Cursor appears in Last Name data entry box
5. “Smith” appears in Last Name data entry box and remains throughout the test
6. Cursor appears in Address data entry box
7. “520 Chestnut Street, Philadelphia, PA” appears in Address data entry box and remains
8. Cursor appears in Phone Number data entry box
9. “877-444-6777” appears in Phone Number data entry box and remains throughout the test
10. Cursor appears in Credit Card Number data entry box
11. “1111222233334444” appears in Credit Card Number data entry box and remains
12. Save Guest button reflects graphical response to action and Guest Reservation UI reappears

Environmental Needs

19. Sufficient hard disk space to store increasing amounts of data.
20. Working computer with a processor capable of executing multiple simultaneous commands.
21. Mouse
22. Keyboard
23. Monitor
24. Operating system with BBMS installed.

Special Procedural Requirements

BBMS service will require guest's limited access starting in the GuestReservation Entity (interface section through service section and management).

Inter-case Dependencies

BBMS-UI-01 and BBMS-UI-04 must be completed first in order for the manager to initiate the workflow and create a reservation.

4.1.6 Specification ID: BBMS-UI-06

Objective: The objective of this test case is to provide a black-box test for the Calendar Reservations performed with the user interface as required by section 3.1.1 of the SRS.

Test Items: This test will emulate a guest utilizing the Reservation Entity for choosing Reservation Calendar scheduling.

Input Specifications

1. Click Guest Reservation button on window. This test will emulate a guest utilizing the Reservation Entity for choosing Reservation Calendar scheduling.
2. Click Guest Reservation create option.
3. Click Reservation Calendar date(s).
4. Click Reservation Start Date.
5. Enter "July 04, 2015"
6. Click Reservation End Date.
7. Enter "July 07, 2015"
8. Click Rooms Vacancies.
9. Click Rooms.
10. Click Room IDs.
11. Enter "Room ID 1A10."
12. Click Estimated Room Price.
13. Click Save Reservation.
14. Click Payment.
15. Click on Credit Card Number data entry box
16. Enter “1111222233334444”
17. Click Save Guest button.

Output Specifications

1. Guest Reservation widow appears with Guest Reservation option button
2. Cursor appears over Reservation Calendar.
3. Cursor appears in Reservation Start Date data entry box.
4. "July 04, 2015" appears in Start Date data entry box and remains throughout the test.
5. Cursor appears in Reservation End Date data entry box
6. “July 07, 2015” appears in End Date data entry box and remains throughout the test
7. Cursor appears over Rooms Vacancies availability dates box
8. “YES” appears in Rooms Vacancies availability box and remains
9. Cursor appears over Rooms display
10. "4" appears in Room data box and remains throughout the test.
11. Cursor appears over Room IDs button
12. “1A10, 1A14, 1A15, and 1A19” room IDs appears in Rooms data entry box and remains throughout the test
13. Cursor appears over Room ID.
14. "1A10" Room ID assigned.
15. Cursor appears in Room Price.
16. "$444.00" Estimated Price.
17. Cursor appears in Credit Card Number data entry box

18. “1111222233334444” appears in Credit Card Number data entry box and remains
19. Save Guest button reflects graphical response to action and Guest Reservation UI reappears.

Environmental Needs

25. Sufficient hard disk space to store increasing amounts of data.
26. Working computer with a processor capable of executing multiple simultaneous commands.
27. Mouse
28. Keyboard
29. Monitor
30. Operating system with BBMS installed.

Special Procedural Requirements

BBMS service will require guest's limited access starting in the GuestReservation Entity (interface section through service section and management).

Inter-case Dependencies

BBMS-UI-01 and BBMS-UI-04 similar to manager must be completed first in order for the manager to initiate the workflow and create a reservation in the interaction section of the ***Overall Entity Relationship Design Diagram*** (figure 1, SDD).

4.1.7 Specification ID: BBMS-UI-07

Objective: The objective of this test case is to provide a black-box test for the Calendar Reservation updates performed with the user interface as required by section 3.1.1 of the SRS.

Test Items: This test will emulate a guest utilizing the Reservation Entity for update (updating) Reservation scheduling.

Input Specifications

1. Click Guest Reservation button on window.
2. Click Guest Reservation select: update option.
3. Click Reservation Calendar date(s).
4. Click Reservation Calendar update option. (Change in schedule, situation or weather condition).
5. Click Reservation update. (For changes to Reservation End Date)
6. Enter "July 09, 2015".
7. Click Room Vacancies. (Steps after this is to verify vacancies, room assignments and payment)
8. Click Rooms.
9. Click Room IDs.
10. Click Estimated Room Price.
11. Click Save Reservation.
12. Click Payment.
13. Click on Credit Card Number data entry box
14. Enter “1111222233334444”
15. Click Save Guest button

Output Specifications

1. Guest Reservation widow appears with Guest Reservation option button
2. Cursor appears over Reservation Calendar update option button.
3. Cursor appears in Reservation Update button.
4. "July 09, 2015" appears in End Date data entry box and remains throughout the test.
5. Cursor appears over Rooms Vacancies availability dates box
6. “YES” appears in Rooms Vacancies availability box and remains
7. Cursor appears over Rooms display
8. "4" appears in Room data box and remains throughout the test.
9. Cursor appears over Room IDs button
10. “1A10, 1A14, 1A15, and 1A19” room IDs appears in Rooms data entry box and remains throughout the test
11. Cursor appears over Room ID.
12. "1A10" Room ID assigned.
13. Cursor appears in Room Price.
14. "$666.00" Estimated Price.
15. Cursor appears in Credit Card Number data entry box
16. “1111222233334444” appears in Credit Card Number data entry box and remains
17. Save Guest button reflects graphical response to action and Guest Reservation UI reappears

Environmental Needs

31. Sufficient hard disk space to store increasing amounts of data.
32. Working computer with a processor capable of executing multiple simultaneous commands.
33. Mouse
34. Keyboard
35. Monitor
36. Operating system with BBMS installed.

Special Procedural Requirements

BBMS service will require guest's limited access starting in the GuestReservation Entity (interface section through service section and management).

Inter-case Dependencies

BBMS-UI-01 and BBMS-UI-04 similar to manager must be completed first in order for the manager to initiate the workflow and create a reservation in the interaction section of the ***Overall Entity Relationship Design Diagram*** (figure 1, SDD).

4.1.8 Specification ID: BBMS-UI-08

Objective: The objective of this test case is to provide a black-box test for the Calendar Reservation deletions performed with the user interface as required by section 3.1.1 of the SRS.

Test Items: This test will emulate a guest utilizing the Reservation Entity for delete (deleting) Reservation scheduling.

Input Specifications

1. Click Guest Reservation button on window.
2. Click Guest Reservation select: delete option.
3. Click Reservation Calendar date(s).
4. Click Reservation Calendar delete option. (Change in schedule, situation or weather condition).
5. Click reservation delete. (Ends the reservation)
6. Enter "date when deleted". (Register at BBMS)
7. Click Room Vacancies.
8. Click Rooms.
9. Click Room IDs.
10. Click Estimated Room Price.
11. Click Save Reservation.
12. Click Payment.
13. Click on Credit Card Number data entry box
14. Enter “0”
15. Click Save Guest button

Output Specifications

1. Guest Reservation widow appears with Guest Reservation option button
2. Cursor appears over Reservation Calendar delete option button.
3. Cursor appears in Reservation Delete button.
4. "............" appears in Reservation Date data box and remains throughout the test.
5. Cursor appears over Rooms Vacancies availability dates box.
6. “YES” appears in Rooms Vacancies availability box and remains.
7. Cursor appears over Rooms display
8. "4" appears in Room data box and remains throughout the test.
9. Cursor appears over Room IDs button.
10. “1A10, 1A14, 1A15, and 1A19” room IDs appears in Rooms data entry box and remains throughout the test.
11. Cursor appears over Room ID.
12. "........." Room ID assigned.
13. Cursor appears in Room Price.
14. "0" Estimated Price.
15. Cursor appears in Credit Card Number data entry box.
16. “0” appears in Credit Card Number data entry box and remains.
17. Save Guest button reflects graphical response to action and Guest Reservation UI reappears.

Environmental Needs

37. Sufficient hard disk space to store increasing amounts of data.
38. Working computer with a processor capable of executing multiple simultaneous commands.
39. Mouse
40. Keyboard
41. Monitor
42. Operating system with BBMS installed.

Special Procedural Requirements

BBMS service will require guest's limited access starting in the GuestReservation Entity (interface section through service section and management).

Inter-case Dependencies

BBMS-UI-01 and BBMS-UI-04 similar to manager must be completed first in order for the manager to initiate the workflow and create a reservation in the interaction section of the ***Overall Entity Relationship Design Diagram*** (figure 1, SDD).

4.2 Service Test Cases

These cases are designed to test the application programming interface (API) of the BBMS, and can be performed either by a user testing team or through automated testing.

4.2.1 Specification ID: BBMS-SV-01

Objective: The objective of this test case is to provide a white-box (process) test for the reservations algorithm to assign a new reservation through the service.

Test Items: Reservation (Object), Reservation[] checkDates, assignReservation, “Make Reservation” button (See SDD: sec 5.1 pg. 15, sec 6.1.2 pg. 21)

Input Specifications

1. Click on a date on Calendar (UI) or enter a date range in text boxes.
2. Click on an available room.
3. Click on “Make Reservation” button.

Output Specifications

1. Guest Reservation form will be displayed and the user will be able to enter the reservation information.
2. assignReservation service will show a new domain object being created.

Environmental Needs

43. Sufficient hard disk space to store increasing amounts of data.
44. Working computer with a processor capable of executing multiple simultaneous commands.
45. Mouse
46. Keyboard
47. Monitor
48. Operating system with BBMS installed.

Special Procedural Requirements

More than one room may be available and would require the user to request additional preferences from the customer to advance to the reservation form.

Inter-case Dependencies

None

4.2.2 Specification ID: BBMS-SV-02

Objective: The objective of this test case is to provide a white-box (process) test for the guest information algorithm to assign a new guest through the service.

Test Items: takeGuestInfo(date, date, double, bool, date), Guest[] guests, Guest Information form, “Save Guest button” (Guest Information Form) (See SDD: sec pg. 19, sec 6.1.5 pg. 23-24)

Input Specifications

1. Verify the Guest Reservation form is shown.
2. Click on “Guest Information” to access the Guest Information form.
3. Enter customer’s first name.
4. Enter customer’s last name.
5. Enter customer’s address.
6. Enter customer’s phone number.
7. Enter customer’s credit card number.
8. Click on “Save Guest” button.
9. If successful, the Guest Information form will automatically close and the system will return to the Guest Reservation form.

Output Specifications

1. Guest Information form will automatically close and the system will return to the Guest Reservation form.
2. takeGuestInfo service will show a new domain object being created.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.

Special Procedural Requirements

BBMS must be at a point to accept guest information. At this point, system expects valid input.

Inter-case Dependencies

The CalendarLookup test case (BBMS-SV-01) must be completed prior to the TakeGuestInformation test because the Guest Information form can only be accessed after the user selects “Make Reservation” on the Calendar Lookup form.

4.2.3 Specification ID: BBMS-SV-03

Objective: The objective of this test case is to provide a white-box (process) test for the guest information algorithm to cancel assigning a new guest through the service.

Test Items: takeGuestInfo(date, date, double, bool, date), Guest[] guests, “Cancel” button (See SDD: sec 5.5 pg.19, sec 6.1.5 pg. 23-24)

Input Specifications

1. Verify the Guest Reservation form is shown.
2. Click on “Guest Information” to access the Guest Information form.
3. Enter customer’s first name.
4. Enter customer’s last name.
5. Enter customer’s address.
6. Enter customer’s phone number.
7. Enter customer’s credit card number.
8. Click on “Cancel” button.

Output Specifications

1. Guest Information form will automatically close and the system will return to the Calendar form.
2. takeGuestInfo service will not show a new domain object being created.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.

Special Procedural Requirements

None

Inter-case Dependencies

The CalendarLookup test case (BBMS-SV-01) must be completed prior to the TakeGuestInformation test because the Guest Information form can only be accessed after the user selects “Make Reservation” on the Calendar Lookup form.

4.2.4 Specification ID: BBMS-SV-04

Objective: The objective of this test case is to provide a white-box (process) test for the payment algorithm to make a payment and assign it to a guest through the service.

Test Items: makePayment(Payment), assignGuest(Guest), “Make Payment” button (See SDD: sec 5.3 pg. 17, sec 6.1.3 pg. 22)

Input Specifications

1. Verify the Guest Reservation form is shown.
2. Click on “Make a Payment” to access the Make Payment form.
3. Enter a total.
4. Enter the payment amount.
5. Select the applicable guest.
6. Click on “Make Payment”.

Output Specifications

1. If successful, the Make a Payment form will automatically close and the system will return to the Guest Reservation form.
2. makePayment service will show a new payment made.
3. takeGuestInfo service will a relationship between the payment and guest.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.

Special Procedural Requirements

None

Inter-case Dependencies

The CalendarLookup test case (BBMS-SV-01) must be completed prior to the MakePayment test case because the Make Payment form can only be accessed after the user selects “Make Reservation” on the Calendar Lookup form. The TakeGuestInformation test case must also be completed prior to this test because a payment must be assigned to a guest’s profile.

4.2.5 Specification ID: BBMS-SV-05

Objective: The objective of this test case is to provide a white-box (process) test for the payment algorithm to cancel a payment through the service.

Test Items: MakePayment form, “Cancel” button (See SDD: sec 5.3 pg. 17, sec 6.1.3 pg. 22)

Input Specifications

1. Verify the Guest Reservation form is shown.
2. Click on “Make a Payment” to access the Make Payment form.
3. Click on “Cancel”.

Output Specifications

1. If successful, the Make a Payment form will automatically close and the system will return to the Guest Reservation form.
2. makePayment service will not show a payment made.
3. takeGuestInfo service will not a relationship between the payment and guest.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.

Special Procedural Requirements

None

Inter-case Dependencies

The CalendarLookup test case (BBMS-SV-01) must be completed prior to the MakePayment test case because the Make Payment form can only be accessed after the user selects “Make Reservation” on the Calendar Lookup form. The TakeGuestInformation test case must also be completed prior to this test because a payment must be assigned to a guest’s profile.

4.2.6 Specification ID: BBMS-SV-06

Objective: The objective of this test case is to provide a white-box (process) test for the reservation algorithm to create a reservation and assign to a guest through the service.

Test Items: Reservation (Object), Reservation[] reservation, createReservation(Reservation), takeRoomReservation(void), RoomReservation roomReservation, "Save Reservation" button (See SDD: sec 5.4 pg. 18, sec 6.1.4 pg. 22-23)

Input Specifications

1. Verify the Guest Reservation form is shown and values are present for the reservation start and end dates.
2. Enter the appropriate price per day in text box.
3. Check (or uncheck if not paid) the guaranteed checkbox.
4. Enter the guarantee date the customer must pay by.
5. Click on "Save Reservation".

Output Specifications

1. Guest Reservation form will close and the RoomReservation form will be displayed.
2. createReservation will show a new reservation entity created.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.

Special Procedural Requirements

None

Inter-case Dependencies

The CalendarLookup test case (BBMS-SV-01) must be completed prior to the GuestReservation test case because the GuestReservation form can only be accessed after the user selects "Make Reservation" on the Calendar Lookup form. Test cases BBMS-SV-02 through BBMS-SV-02 must also be completed first so all of the information needed to save the reservation is entered.

4.2.7 Specification ID: BBMS-SV-07

Objective: The objective of this test case is to provide a white-box (process) test for the reservation algorithm to delete a reservation and remove a guest through the service.

Test Items: Reservation (Object), Reservation[] reservation, deleteReservation(Reservation), takeRoomReservation(void), RoomReservation roomReservation, “Delete Reservation” button (See SDD: sec 5.4 pg. 18, sec 6.1.4 pg. 22-23)

Input Specifications

1. Verify the Guest Reservation form is shown and values are present for the reservation start and end dates.
2. Enter the appropriate price per day in text box.
3. Check (or uncheck if not paid) the guaranteed checkbox.
4. Enter the guarantee date the customer must pay by.
5. Click on “Save Reservation”.

Output Specifications

1. Guest Reservation form will close and the system will return to the Calendar form.
2. deleteReservation will return Boolean status of reservation deletion.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.

Special Procedural Requirements

None

Inter-case Dependencies

The CalendarLookup test case (BBMS-SV-01) must be completed prior to the GuestReservation test case because the GuestReservation form can only be accessed after the user selects “Make Reservation” on the Calendar Lookup form. Test cases BBMS-SV-02 through BBMS-SV-02 must also be completed first so all of the information needed to save the reservation is entered.

4.2.8 Specification ID: BBMS-SV-08

Objective: The objective of this test case is to provide a white-box (process) test for the room algorithm to create a room through the service.

Test Items: Room getRoom(int), createRoom(Room), Room[] rooms, “Save Room” button (See SDD: sec 5.2 pg. 16, sec 6.1.1 pg. 21)

Input Specifications

1. Verify the RoomReservation form is shown.
2. Enter a room number.
3. Check (or uncheck accordingly) the Vacant checkbox.
4. Click the “Save Room” button.

Output Specifications

1. RoomReservation form will close and the system will return to the Calendar form.
2. createRoom service will create a new Room entity.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.

Special Procedural Requirements

None

Inter-case Dependencies

Test cases BBMS-SV-01 through BBMS-SV-07 test cases must be completed prior to the RoomReservation test case because the RoomReservation form can only be accessed after the user selects “Make Reservation” on the Calendar Lookup form, and all of the information has been added.

4.2.9 Specification ID: BBMS-SV-09

Objective: The objective of this test case is to provide a white-box (process) test for the room algorithm to delete a room through the service.

Test Items: Room getRoom(int), deleteRoom(Room), Room[] rooms, “Delete Room” button (See SDD: sec 5.2 pg. 16, sec 6.1.1 pg. 21)

Input Specifications

1. Verify the RoomReservation form is shown.
2. Enter a room number.
3. Click the “Delete Room” button.

Output Specifications

1. RoomReservation form will close and the system will return to the Calendar form.
2. deleteRoom service will return a Boolean indicator (true) showing the room was deleted.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.

Special Procedural Requirements

None

Inter-case Dependencies

BBMS-SV-01 through BBMS-SV-07 test cases must be completed prior to the RoomReservation test case because the RoomReservation form can only be accessed after the user selects “Make Reservation” on the Calendar Lookup form, and all of the information has been added.

4.3 Domain Object Test Cases

These cases are designed to test the data encapsulation and storage portion of the BBMS, and can be performed either by a user testing team or through automated testing.

4.3.1 Specification ID: BBMS-DO-01

Objective: The objective of this test case is to provide a white-box (process) test for the Guest domain object have the guest name changed.

Test Items: The items to be tested in this test case are the data validation function of the “get” and “set” member functions of the guest object. This case tests valid data entry.

Input Specifications

1. The user of the BBMS enters a valid customer name (e.g. John Smith) of string data type.

Output Specifications

1. The successful creation of a guest object
2. Entry into an event log indicating successful update.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.
7. The BBMS services need to be running and event and error logs capturing data.

Special Procedural Requirements

None

Inter-case Dependencies

BBMS-SV-02

4.3.2 Specification ID: BBMS-DO-02

Objective: The objective of this test case is to provide a white-box (process) test for the Guest domain object have the guest credit card information changed.

Test Items: The items to be tested in this test case are the data validation function of the “get” and “set” member functions of the guest object. This case tests valid data entry.

Input Specifications

1. The user of the BBMS enters a valid 16 digit credit card number without spaces of type integer.

Output Specifications

1. The successful creation of a guest object
2. Entry into an event log indicating success.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.
7. The BBMS services need to be running and event and error logs capturing data.

Special Procedural Requirements

None

Inter-case Dependencies

BBMS-SV-02

4.3.3 Specification ID: BBMS-DO-03

Objective: The objective of this test case is to provide a white-box (process) test for the Guest domain object have the guest name changed and validate improper data.

Test Items: The items to be tested in this test case are the data validation function of the "get" and "set" member functions of the guest object. This case tests invalid data entry.

Input Specifications

1. The user of the BBMS enters a user name that includes a value of 0-9 or special character.

Output Specifications

1. No update to the Guest entity.
2. Validation shows error of data entry.
3. Event log indicating failure.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.
7. The BBMS services need to be running and event and error logs capturing data.

Special Procedural Requirements

None

Inter-case Dependencies

BBMS-SV-02

4.3.4 Specification ID: BBMS-DO-04

Objective: The objective of this test case is to provide a white-box (process) test for the Guest domain object have the guest name changed and validate improper data.

Test Items: The items to be tested in this test case are the data validation function of the "get" and "set" member functions of the guest object. This case tests invalid data entry.

Input Specifications

1. The user of the BBMS enters a credit card number that includes spaces or any character other than 0-9.

Output Specifications

1. No update to the Guest entity.
2. Validation shows error of data entry.
3. Event log indicating failure.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.
7. The BBMS services need to be running and event and error logs capturing data.

Special Procedural Requirements

None

Inter-case Dependencies

BBMS-SV-02

4.3.5 Specification ID: BBMS-DO-05

Objective: The objective of this test case is to provide a white-box (process) test for the Reservation domain object to have the guarantee status changed and validated.

Test Items: The items to be tested are the isGuaranteed(void) Boolean and the setGuaranteeDate(date) member functions of the Reservation object.

Input Specifications

1. The user of the BBMS attempts to create a reservation for a date that has been guaranteed by other customers.
2. The reservation date information is entered in the appropriate fields of the guest reservation element.
3. The user clicks "Save Reservation."

Output Specifications

1. The user will receive an error indicating that all rooms have been reserved by other customers and payment received.
2. The isGuaranteed Boolean would have been set to ***true*** for all reservations on that date.
3. The error is logged.

Environmental Needs

1. Sufficient hard disk space to store increasing amounts of data.
2. Working computer with a processor capable of executing multiple simultaneous commands.
3. Mouse
4. Keyboard
5. Monitor
6. Operating system with BBMS installed.
7. The BBMS services need to be running and event and error logs capturing data.

Special Procedural Requirements

None

Inter-case Dependencies

BBMS-SV-06