|  |  |
| --- | --- |
| Assessment Title | Assignment (CarParkSA) |

## Competency Details

|  |  |
| --- | --- |
| Unit code/s and title/s | ICTPRG441 - Apply skills in object-oriented design |
| Qualification code/s and title/s | ICT40120 - Certificate IV in Information Technology |
| Business unit/Work group | BARTS/IT Studies |

## Instructions

|  |  |
| --- | --- |
| Method/s of assessment | Product (Create & Written) |
| Overview of assessment | This assessment will require you to complete 4 parts of the assignment in the CarParkSA scenario. The assignment requires you to document the system requirements and develop system specifications using the static and dynamic design techniques. The design deliverables involve various diagrams such as use case diagram, class diagram, sequence diagram activity diagram and stat chart diagram.  In this assessment you will cover the following topics:   * Document Requirements: use case diagrams, use case specifications and activity diagrams * Use of class diagrams * Use of sequence diagrams * Use of communication diagrams   The assessment is broken up and assessed in 4 parts and you will submit each part gradually as you develop your skills through the course. |
| Task/s to be assessed | This assessment requires you to complete the following tasks:   * 1. Assignment (Part 1) - Use Case Diagram, Use Case Specification & Activity Diagram   2. Assignment (Part 2) - Class Diagram   3. Assignment (Part 3) - Sequence Diagram   4. Assignment (Part 4) - Communication Diagram |
| Time allowed | Refer to your schedule for submission dates |
| Location of assessment | You can complete this assessment during your practical sessions and at home. |
| Decision making rules | To receive a satisfactory outcome for this assessment you must complete all parts correctly. |
| Assessment conditions | This assessment must be undertaken where the conditions replicate noise levels and interruptions that people typically experience working in the ICT industry.    This is unsupervised assessment and you may access any required resources.    This is not group work and must be completed as an individual. |
| Resources required | To complete this assessment, you will require the following:   * Access to Learn with Internet access * Learn resources * ICTPRG441 – ASDS – CarParkSA Assignment (Part 1) Student Files * ICTPRG441 – ASDS – CarParkSA Assignment (Part 2) Student Files * ICTPRG441 – ASDS – CarParkSA Assignment (Part 3) Student Files * ICTPRG441 – ASDS – CarParkSA Assignment (Part 4) Student Files * Word processing software such as Microsoft Word. * StarUML |
| Result notification and reassessment information | You will be provided feedback and the result for your assignment on TAFESA Learn. You will be and given the chance to resubmit with required corrections only once.  Refer to the TAFE SA assessment policy for more information <https://www.tafesa.edu.au/apply-enrol/before-starting/student-policies/assessment> |

**Assignment (Part 1 ~ Part 4) - CarParkSA**

CarParkSA is a private car park operates in Adelaide city centre. The car park is operating using the traditional parking ticket system.

CarParkSA is going to upgrade the system with features such as using license plate recognition technology, keep the record of registered customer and calculate the charge fee when exit. The new system does not require the paper ticket at entry or exit any more.

The paper ticket will be replaced by the license plate recognition (LPR) technology. See the following links

<http://www.youtube.com/watch?v=-owRUp1-Hf8>

IT Works Company has got the contract to produce a high-level design of the system:

* Use license plate recognition technology. Therefore, the new system does not require the paper ticket at entry or exit any more.
* Keep the record of registered customer
* Calculate the charge fee when exit

As you are a system designer in IT Works, you are required to produce a high-level design of the system. Jim Jones, the director, confirmed that the requirements not mentioned in the user stories below should not be included in the stage 2 of the development.

The user stories are collected in the following table:

|  |  |
| --- | --- |
| Unregistered customer | Register as customer online:   * Customer must register account online to become a registered customer. |
| Registered monthly customer | * Subscribe a reserved parking space by providing the customer information and the car details online. * Pay monthly payment online. |
| Registered casual customer | * To avoid paying cash at exit, customer may register by providing customer information and the car details online. * Deposit money initially and when account balance is low.   \*\*Note: Casual customer does not need any registration. |
| CarInCarPark  (Note: can either be registered monthly, registered casual or casual car) | At entry:   * When a Car goes through the car park entry, the camera will recognise the registration no. from the license plate. * The system will identify whether the incoming car is a registered car stored in the database. * The system identifies whether the incoming car is registered monthly, registered casual or casual car. * The system will get the current system date/time. * The system records the car entry details including the date /time of entry. * The system will add 1 to the total no. of cars in car park * The system will save the car in the car park to the database. |
| CarInCarPark | At exit:   * When the CarInCarPark arrives to the car park exit, the camera will recognise the registration no. from the license plate. * The system will retrieve the car in car park from the database. * If the payment method is “RM’ or “RC”, the system will also retrieve the registered car from the database using the registration no. * The system will calculate the parking charge.   For registered monthly cars:   * Park their cars at the reserved space. No need to pay parking charge at exit.   For registered casual cars:   * The system will retrieve the registered customer from the database. * Calculated parking charged is bigger than the account balance, need to pay the charge difference by cash (note: next stage of development, able to pay by credit card). * Registered monthly & Registered casual account customers may charge up account balance online. * View the balance online.   For casual cars:   * Pay for casual parking charge at exit. |
| Customer (register monthly customer or registered casual customer) | Register as customer online:   * Customer must register account online to become a registered customer.   For registered casual customer:   * To avoid paying cash at exit, customer may register by providing customer details and car details. They can deposit some money as initial account balance.   For registered monthly customer:   * Subscribe a reserved parking space with their car details online. * Pay monthly payment in advance online.   \*\*Note: Casual customer does not need any registration. |
| Staff | * Allocate parking space to registered monthly customer. * Maintain parking space information * Maintain the parking fees: monthly charge, casual hourly rate, daily charge and early bird charge. |
| System clock | * Print monthly statement to registered monthly customers. |
| Manager | * Print daily, weekly & monthly sales report. * Print car park utilization rate report. |

Note: the system uses a database system to record the information and handle the payment through the payment system.

## **Assignment Requirements:**

**Part 1. (Use Case Diagram, Use Case Specification and Activity Diagram)**

1. Watch the requirements video. Open the CarParkSA requirements worksheet file and determine which actors, functional requirements (i.e. use cases), non-functional requirements, and the activities of Record Entry are in the scope of the design.

*Note: the given activities list of* ***Record Exit (indicated in yellow)*** *business process are shown as examples for your references.*

In the worksheet, mark them in yellow if they are within the scope of design:

* + actors (9 required),
  + functional requirements i.e.use cases (12 required),
  + non-functional requirements (3 required),
  + and the activities of Record Entry (14 required).

Submit the completed worksheet to LEARN for marking.

1. Develop a use case diagram using StarUML. Draw a **use case diagram** in the **use case view**. Name the diagram as “**CarParkSA Use case Diagram**”. Include all actors and use cases that you have identified in the requirement worksheets.
2. Draw an activity diagram for the “**Record Entry**” business process. Draw an **activity diagram** under the **Record Entry use case** in **StarUML model.** The diagram must include all activities that you have identified in the requirement worksheet. The activities must be put into the swimlanes such as CarInCarPark and the system.
3. Use the use case specification template (provided in the student file), document the “**Record Entry use case specification**”. Fill each item appropriately according to the CarParkSA scenario. The steps in the basic flow must be described in **round trip description** format. You must also include at least two alternative flows, a pre-condition and a post-condition.

\*\* For submission, you need to submit a zipped file which includes:

* the requirement worksheet
* the use case diagram in StarUML
* the Record Entry use case specification word document and
* the activity diagram in StarUML

**Part 2. (Static Class Diagram)**

Read the above CarParkSA user stories and the business activities. Develop a class diagram using StarUML. To identify the classes, use the techniques of extracting nouns and noun phrases from the above user stories and business rules.

The following paragraph shows business rules and describes the CarParkSA operations.

1. CarParkSA is a multi-levels car park. Each level is identified by a level number. It has information of level name.
2. Each level has one or many parking spaces. Each parking space must be located at a level.
3. Each parking space is identified by a parking space id. It has information length and width, parking space type, subscribed type (casual or monthly subscribed), subscribed start date, subscribed end date.
4. Each car in the car park (CarInCarPark) is identified by a registration number. It has an entry date/time and an exit date/time, charging method and $ parking charge. At entry some attributes are null at entry.
5. Each registered car (either registered monthly car or registered casual car) is identified by the registration no. It has information of manufacturer, model, year of manufacture, body colour, car type (sedan, van or minibus) and account current balance.
6. Each registered car has one registered customer (either registered monthly or registered casual). Each registered customer may own one and only one car. (e.g. if a customer owns two cars, the system will regard it as two customers).
7. Each customer, if registered, is identified by a customer id. It has information like first name, last name and contact phone no.
8. Each customer has an address which is identified by an address id. The address has information of street, suburb, postcode and state. Each address must belong to one and only one customer.
9. Registered monthly customer is a kind of customer. He/she must register online with the customer information, the start date of the month, the end date of the month, allocated parking space (allocated by administrator later) & monthly charge.
10. Registered casual customer is a kind of customer. He/she must register online with the customer information, and the preferred parking level. Instead of paying cash on exit, parking charge will be deducted from the current balance and the related car details. New balance will be calculated. Registered casual customer may deposit money in advance. Paying cash on exit is required when the balance is less than the charge.
11. Each CarInCarPark may have 0 or 1 entry event. Each entry event must have a CarInCarPark.
12. Each CarInCarPark may have 0 or 1 exit event. Each exit event must have a CarInCarPark.

## Each CarInCarPark may be charged to one or many fees. Each fee may be charged to 0, one or many car(s) in the car park.

Each standard fee is identified by a fee id. It has a fee name, fee amount, fee description, start time (if any), exit time (if any).

e.g. Early bird has id = 5, feeName = ‘Early bird’, feeDescription = ‘Early bird has condition must enter before the start time and no exit earlier than the end time’, startTime = ‘09:00:00’ and exitTime = ‘14:00:00’.

e.g. Hourly rate has id = 6, feeName = ‘Hourly rate’, chargeDescription = ‘Hourly rate will be used if other fee no applied’, startTime = null, endTime=’null’

**The following describes the activities when car enters into the car park:**

When a car arrives to the entry point, the system will recognise the licence plate. A new CarInCarPark record will be created. The system will check with the availability of park spaces i.e. how many CarInCarPark in the database. If the car park is not full, the system will display a “Available - Please drive in”, otherwise the system will display “Full, please wait”. The system will keep checking the availability until the car park is not full. The system will record the entry date time of the incoming car. The system will check whether it belongs to one of the registered car stored in the database. If it is a registered car, the system will further determine whether it is a registered monthly car or causal registered car. Otherwise, it is a causal car. The system will set the CarInCarPark car type accordingly. i.e. “RM” for registered monthly, “RC” for registered casual and “C for casual. For causal car, the system only needs information such as registration no. and entry date/time. The system will add 1 to the total no. of cars in the car park and then save the incoming car information to the database.

**The following describes the activities when the cars exit the car park:**

When a car is at the exit, the system reads the registration number. The system will retrieve the parked car record from the database. The exit date/time will be recorded. The system will calculate the parking charge to pay. If it is a registered monthly car, the system will set the parking charge to pay to zero and may exit without payment. If the car is a registered casual car, the system will check whether the account balance is bigger than the parking charge. If yes, the system will calculate the new balance by deducting the parking charge from the current balance. The system will display the parking charge to pay for the registered casual car and the casual car. The system will save the new account balance and the CarInCarPark record into the database.

e.g. The system will calculate the casual cars (both registered or not) parking charge based on the following rules:

* Early bird fee $15 per day max. applies if casual car enters before 9:00am and exits by 7:00pm.
* Hourly charge @5 per hour, if entered after 9:00am.
* If the total hourly charge exceeds $30, then charged at $30 max.

**Part 2 Requirements: - Class Diagram**

1. Open the Assignment (Part 2) student file. In the **Logical View**, create a class diagram under the appropriate package and rename the file as ***yourname\_Part2a\_(Class Diagram).uml****.* The diagram should include:

* The identified classes which as part of the object-oriented design methodology.
* Association between the classes must have name and multiplicities
* Association should consider be aggregation, composition or generalization whereas appropriate
* For each class identified, add attributes and appropriate data types
* Include visibility of the attributes (i.e. state data)
* Include navigability of the associations (if any)

**b)**

1. Copy the StarUML model that you developed in Part a and name it ***yourname\_Part2b\_(Class Diagram refined).uml***. You are required to refine the class diagram to ensure the extra requirements are implemented, if it is not included in the design in Part 2a.

* All customers will have to provide the email address information.
* The registered monthly customer will have to specify their monthly billing cycle date.
* For all registered customers (either registered monthly customers or registered casual customers) will need to indicate their minimum balance to be notified if their current balance falls below that amount. Implement this design preference in your model.
* There is no need to instantiate the Registered Customer class. i.e. the Registered Customer class is abstract.
* The senior analyst would prefer the system to have implemented when the "RegisteredCar" object is destroyed, the corresponding registered customer will also be destroyed from the current memory. Implement this design preference in your model.
* Choose **3 classes** that they are **associated with each other**. Suggest at least **3 operations** **for each of these classes (e.g CarInCarPark, RegisteredCar** and **ParkingSpace)**. You **must** include:
* their operation signatures (if any) and return types.

Make sure that you do not pass more parameters than required.

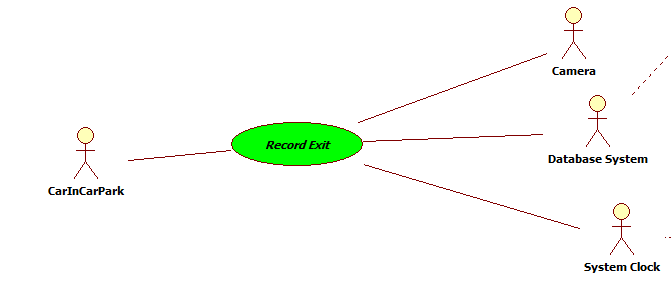
* appropriate visibility of the operations (i.e. class services)

1. In ***yourname\_Part2b\_(Class Diagram refined).uml*** model file, in the CarInCarPark class, design an operation so that it validates the current object has the charge method must be within the values of “RM”, “RC”, “C” or “W”. If it is not one of these values, return true otherwise false. Assign visibility so that these operations can only be called internally within the CarInCarPark class.

**\*\* Note:** use separate StarUML model files for **Part 2a** & **Part 2b**. Do not use one single StarUML model to cover both parts. For submission, compress both **Part 2a** and **Part 2b** models into a zip file and upload to LEARN for marking.

**Part 3. (Sequence Diagram)**

The ITWorks Company continues working on the contract to conduct the Use Case Analysis phase – i.e. use case realization of the system. This stage is the design at the designer’s perspective. The following documents are the **Record Exit Use Case Specification** &the **Record Exit – Basic Flow** Pseudocode,



Record Exit use case specification:

## Brief Description

When a car stops at the exit gate, the license plate number is recognized. The car is identified and retrieved for what kind of charging method to apply. The system will calculate the parking charge to be paid.

## Flow of Events

### Basic Flow

This use case starts when a car stops at the exit point.

#### The system will recognise the registration number through the camera.

#### The system will retrieve the database to locate this parked car.

#### The system will get the current system date/time and will assign as the exit date time.

#### The system will need to calculate the total amount the car needs to be paid:

##### If the charging method is “**RM**” as Registered monthly, the parking charge will be set as zero.

##### Otherwise, *the system will decide whether “****early bird****” price applied or not by retrieving the entry record from the database. The system checks the entry date time. If it is qualified for the early bird, the system will retrieve the item price from the database to get the appropriate price for early bird. Then the parking charge is set accordingly. \*\*Note: the car may have parked for multiple days. Add record to ParkingChargeApplied.*

##### *The system will also check for hours parked (if hourly rate applies) by comparing the entry date & time and exit date & time. The system will retrieve the price from the database. e.g. appropriate hourly price. The parking charge is calculated. Add record to ParkingChargeApplied.*

##### If it is a “**RC**” as RC registered casual car, the system will retrieve the Registered Car from the database including the latest balance. Parking charge will be compared with the balance to see whether sufficient money to cover the parking charge. If sufficient balance, deduct the parking charge to give a new balance. Total charge is set to zero otherwise the total charge will be calculated as the amount to pay.

##### Deduct one from the total number of cars in car park.

##### All related car exit information will be saved to the database.

### Alternative Flows

#### Database not connected

The use case will end if database is not available.

## Special Requirements

None.

## Pre-Conditions

The parked car arrives at the exit point and after the system has recognized the registration no.

## Post-Conditions

The car will be charged at exit. The car in car park record will be saved to the database.

## Extension Points

None.

**Record Exit – Basic Flow Pseudocodes:**

The use case stops at the exit point.

* Obtain RegNo from the camera
* Get CarInCarPark from database
* Get current date time

Calculate totalCharge:

**If** chargingMethod = “RM”

     set totalCharge = 0

**Else**

getParkingCharge:EarlyBird

IF EntryDateTime > EarlyBird:StartTime

      totalCharge = earlyBirdPrice

        -- add object ParkingChargeApplied

setChargeAmountApplied = totalCharge

Else

   -- Calculate for hourly rate

  getParkingCharge:HourlyCharge

        totalCharge = (hourlyCharge \* unitHours)

        -- add object ParkingChargeApplied

setChargeAmountApplied = totalCharge

   Calculate calculatePaidAtExit:

**If** chargingMethod = “RC”      i.e. RegisteredCasual

**Get RegisteredCar**

**If** -- sufficient balance in account

balance > totalCharge

      set balance = balance - totalCharge

      set totalCharge = 0

set ChargeAmountApplied = totalCharge

**Else**

-- insufficient balance in account

      Set totalCharge = totalCharge – balance

      set balance = 0

set ChargeAmountApplied = totalCharge

**EndIf**

**EndIf**

**EndIf**

* Subtract 1 from totalNoOfCars
* Save CarInCarPark to database
* Save ParkingChargeApplied to database
* Save RegisteredCar to database

**Part 3 Requirements: – Sequence Diagram & Class Diagram**

Open the student files (**Yourname\_Part3\_Record\_Exit.UML**) which includes the use case model, use case analysis model, and a use case specification (Record Exit) for the CarParkSA car parking system. Draw a sequence diagram for the Record Exit use case.

1. Copy the domain classes under the CarParkSA Analysis Elements. Stereotype the classes appropriately (i.e. assign classes to specific stereotype) such as:

* entity class
* boundary class
* and control class

1. Model the behaviour of the use cases by conducting the Use Case Realizations (i.e. draw a Sequence Diagram showing the object collaborations) for the **Record Exit** use case.

Your **sequence diagram** must comply with the following:

* The sequence diagram must be drawn in the appropriate package under the concept of RUP Methodology.
* All messages (i.e. flows of event) must be named appropriately and spelled in camelCase
* All messages should have appropriate parameter signatures (parameters & return type). Must provide 6 messages with operation signatures.
* All messages must be generated as operations in the class diagram
* The flows of event must follow the logic as the given pseudocode (with the loops / IF...ELSE conditions) which uses the procedural programming methodology.

1. Additional requirements for refinement of behaviours and classes: the management requests to add a payment method called “**W**” - **Waived**. There are special reasons that the payment should be waived e.g. “RAA Road Service” happened in the car park. This is done by an additional use case. (Note: you do not need to design this additional use case).

Due to the change of the requirement, you are required to incorporate the following changes:

* Add a new attribute called "**comments**" to the CarInCarPark class.
* Add the extra operations in the sequence diagram to change the state of the object with the total charge set to zero and assign text to the comment attribute.

i.e.

…… If Payment Method = “W”

**set Total Charge = 0**

Assign text "Charge waived and exit the car park." To the comment attribute

EndIf

**\*\* Note:**

* *Make sure that these additional operations all generated in the CarInCarPark class in the Class Diagram.*
* *Capture the screen shot of these operations in the sequence diagram and class diagram have been added as evidence of completion.*

**Part 4. (Communication Diagram)**

**Given:**

* The Record Entry pseudocode
* The Record Entry sequence diagram
* The student file ICTPRG441 - ASDS - CarParkSA Assignment (Part 4) Student Files.zip which includes the use case model , the class diagram and the sequence diagram.

**Part 4 Requirements: – Communication Diagram**

Develop a **communication diagram** for the **Record Entry** use case. In the communication diagram, you must include all objects and the correct paths that shows how objects are interacted with other objects.

**Record Entry – Basic Flow Pseudocodes:**

* Obtain the RegNo through the camera
* Create a new CarInCarPark object
* Checking parking space availability or not:

**If** totalNoOfCars >= MAXIMUM\_NO\_OF\_CAR

Display message “Full, please wait”

**Else**

Display message “Available – please drive in”

End If

* Check CarInCarPark is a Registered car
* If RegisteredCar found

Check the CarInCarPark car type:

` If registeredCarType = registeredMonthlyCar

Set carType = “RM”

Else

Set carTpe = “RC”

End If

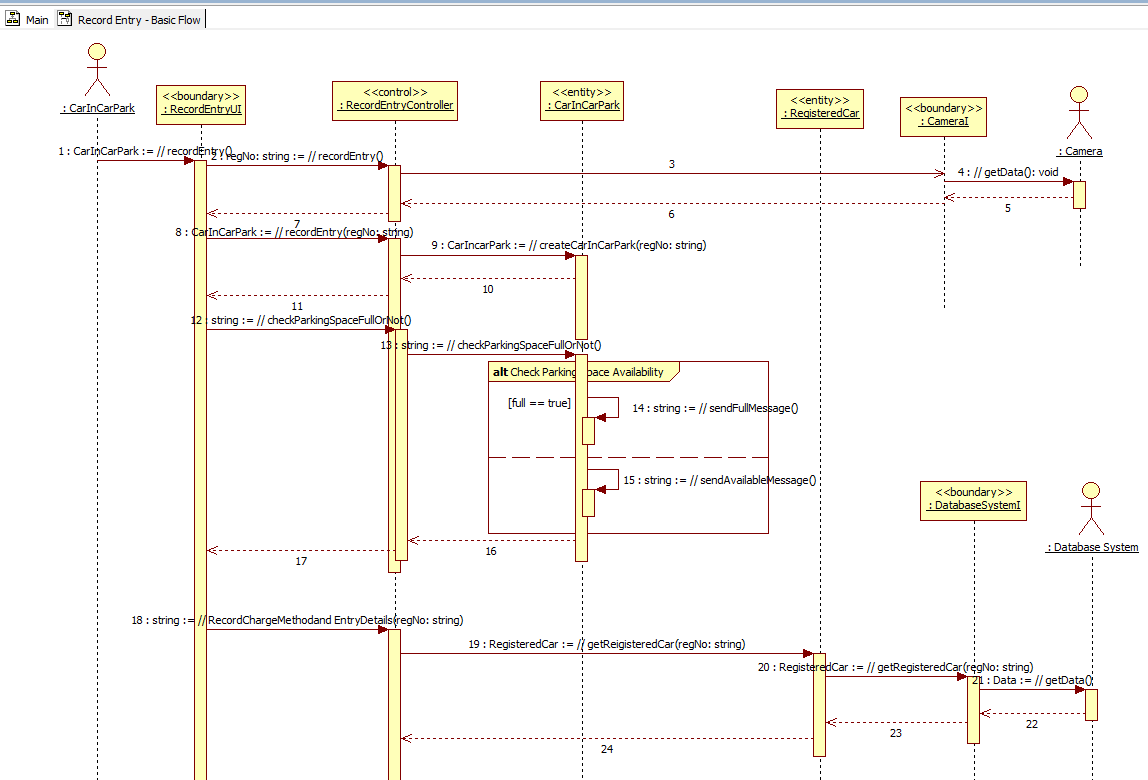
Else

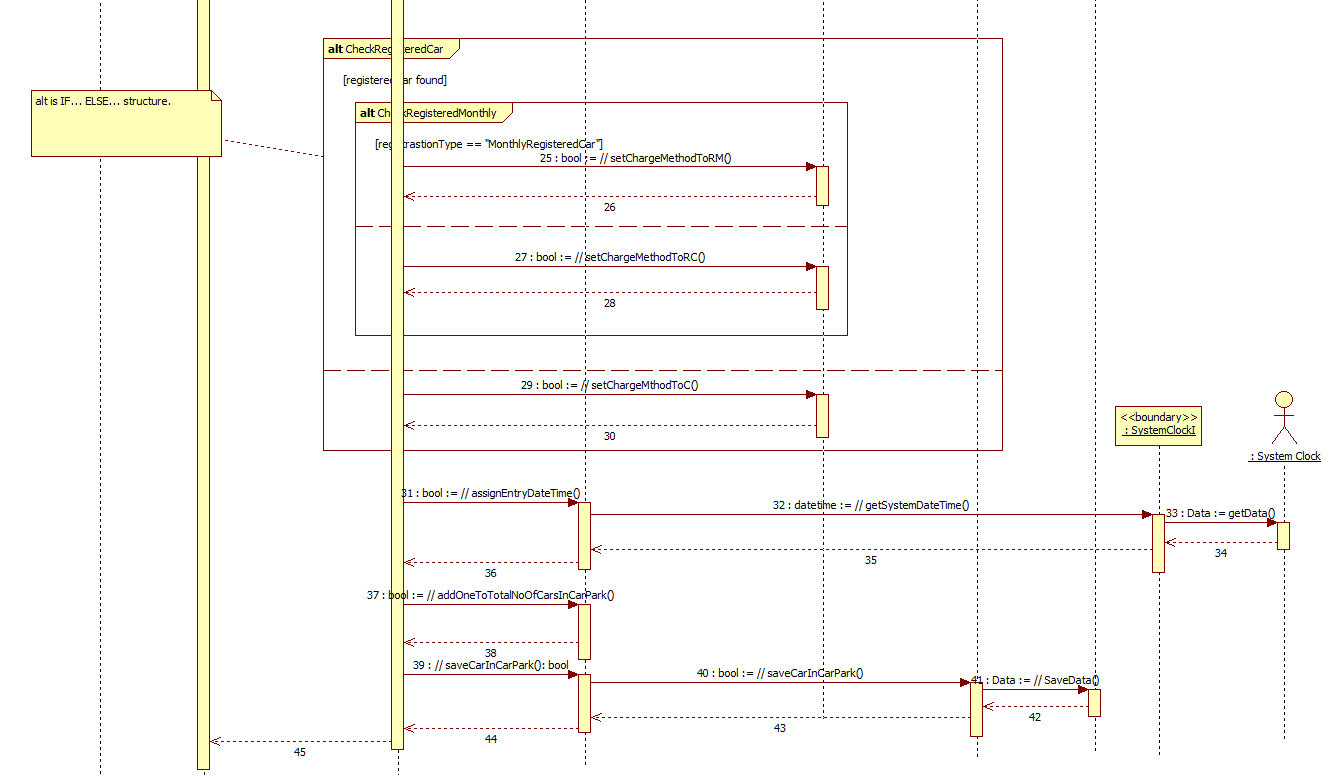
set carType = “C”

End IF

* Assign the entry date/time as the system date/time
* Add 1 to totalNoOfCars
* Save CarInCarPark record to database

**Record Entry Sequence Diagram**





You can find this diagram in the ICTPRG441 - ASDS - CarParkSA Assignment (Part 4) Student Files.zip student file.