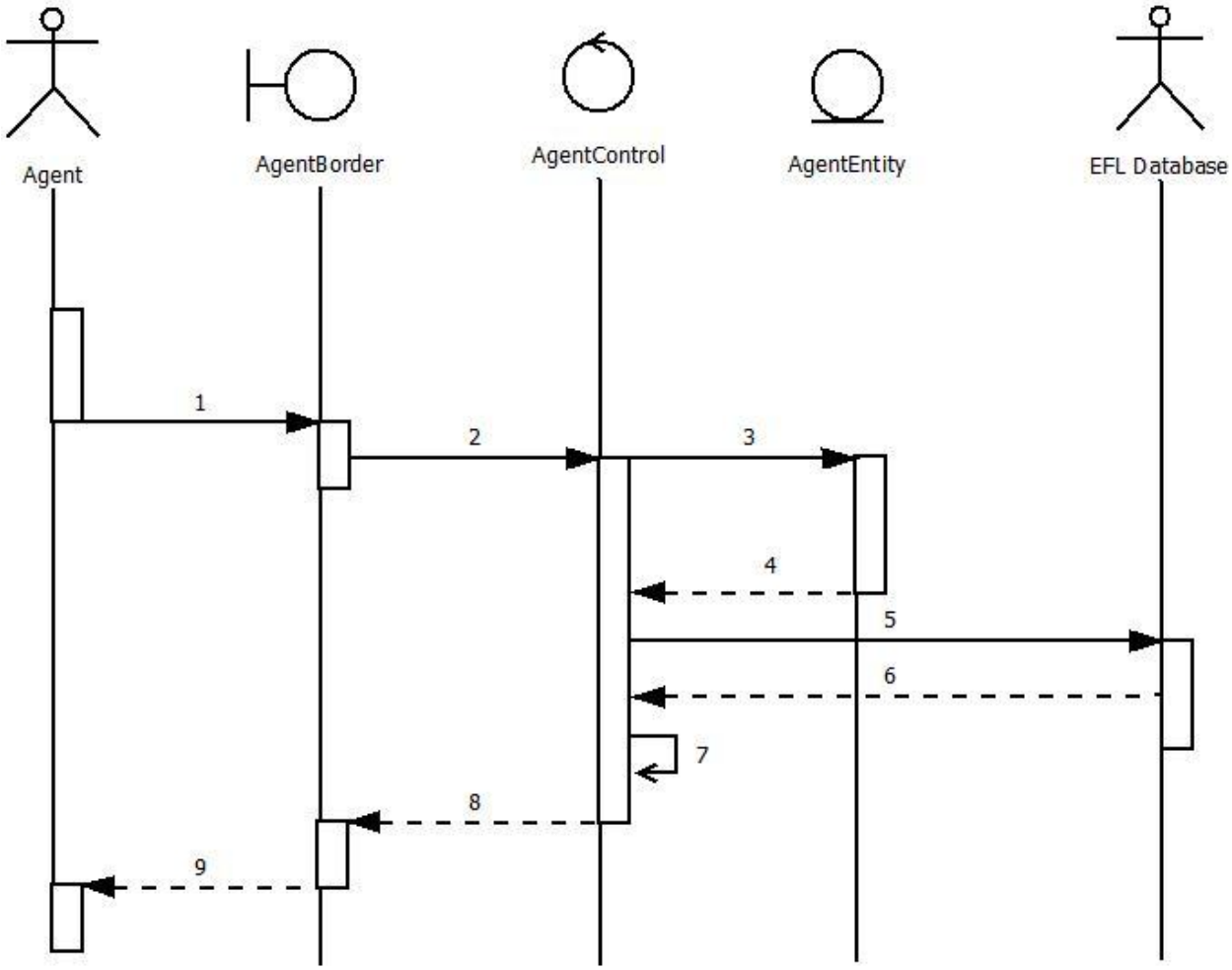
Sequence Diagrams

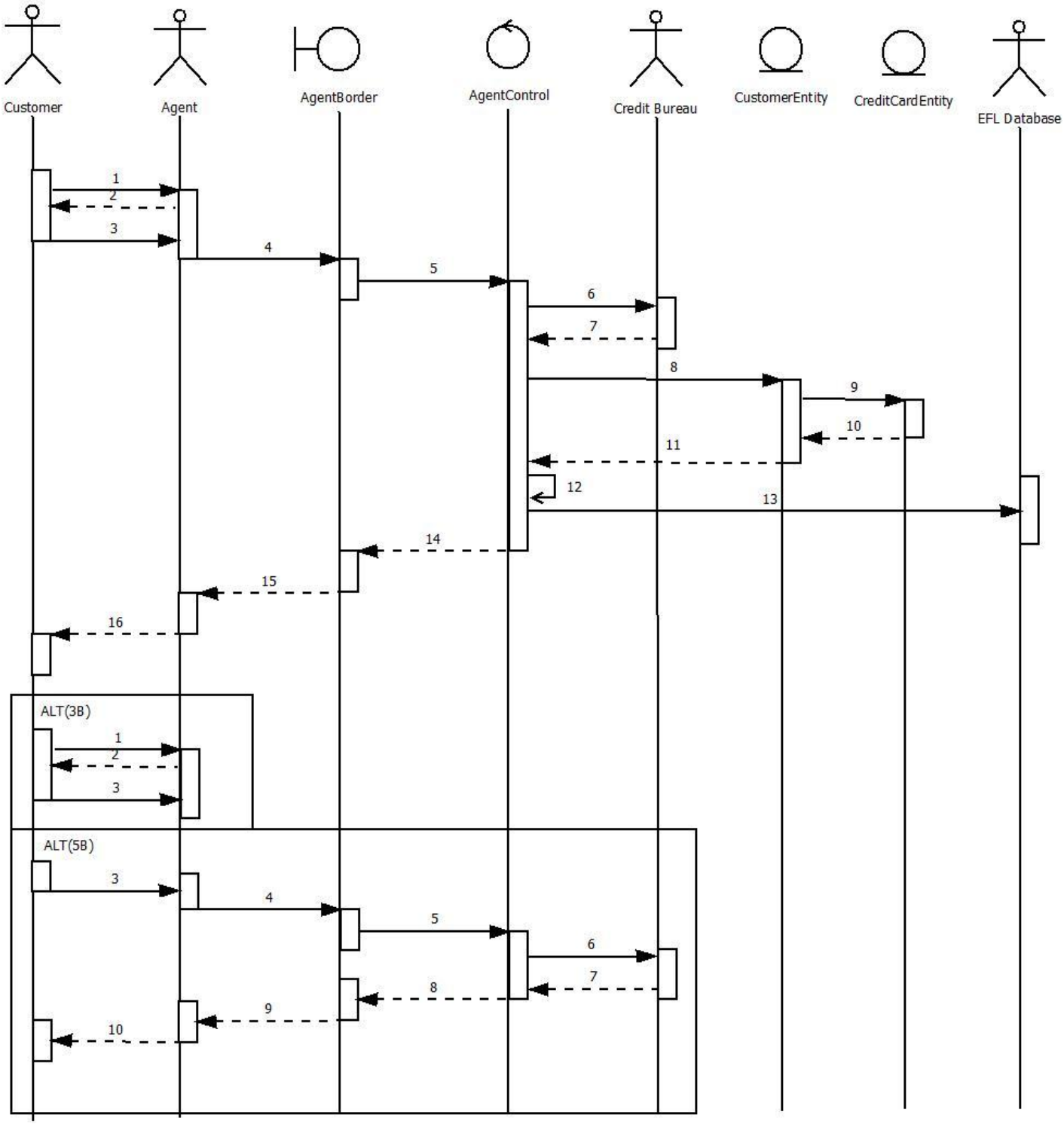
Agent Log-In



Agent Log-In(Normal)

1. Agent enters name and password
2. Agent hits enter and passes information to AgentControl
3. AgentControl calls a new instance of AgentEntity
4. AgentEntity returns a new instance of itself to AgentControl
5. AgentControl requests Agent Information from EFL Database
6. EFL Databse returns Agent Information
7. AgentControl updates information to the new Instance of AgentEntity
8. AgentControl sends info back to AgentBorder
9. AgentBorder updates the display to show info and successful log-in to Agent

Create New Customer Profile



Create New Customer Profile (Normal)

1. New Customer calls or enters the travel agency and would like to create an account
2. Agent explains fee structure and requests information from customer
3. Customer accepts services and provides information and credit card
4. Agent enters information into CreateAccount subscreen
5. Agent Verifies information is correct and hits submit to send information to AgentControl
6. AgentControl sends credit card info to be verified by Credit Bureau
7. Credit Information is cleared (valid)
8. AgentControl calls to create a new instance of Customerentity
9. CustomerEntity calls to create a new instance of CreditCardEntity
10. CreditCardEntity returns a new instance of itself to CustomerEntity
11. CustomerEntity returns a new instance of itself to AgentControl
12. AgentControl updates the new class with current info
13. AgentControl sends customer to EFLDatabase
14. AgentControl sends updated display to AgentBorder with customer instance
15. AgentBorder updates display to Agent
16. Agent Reports a successfully created account

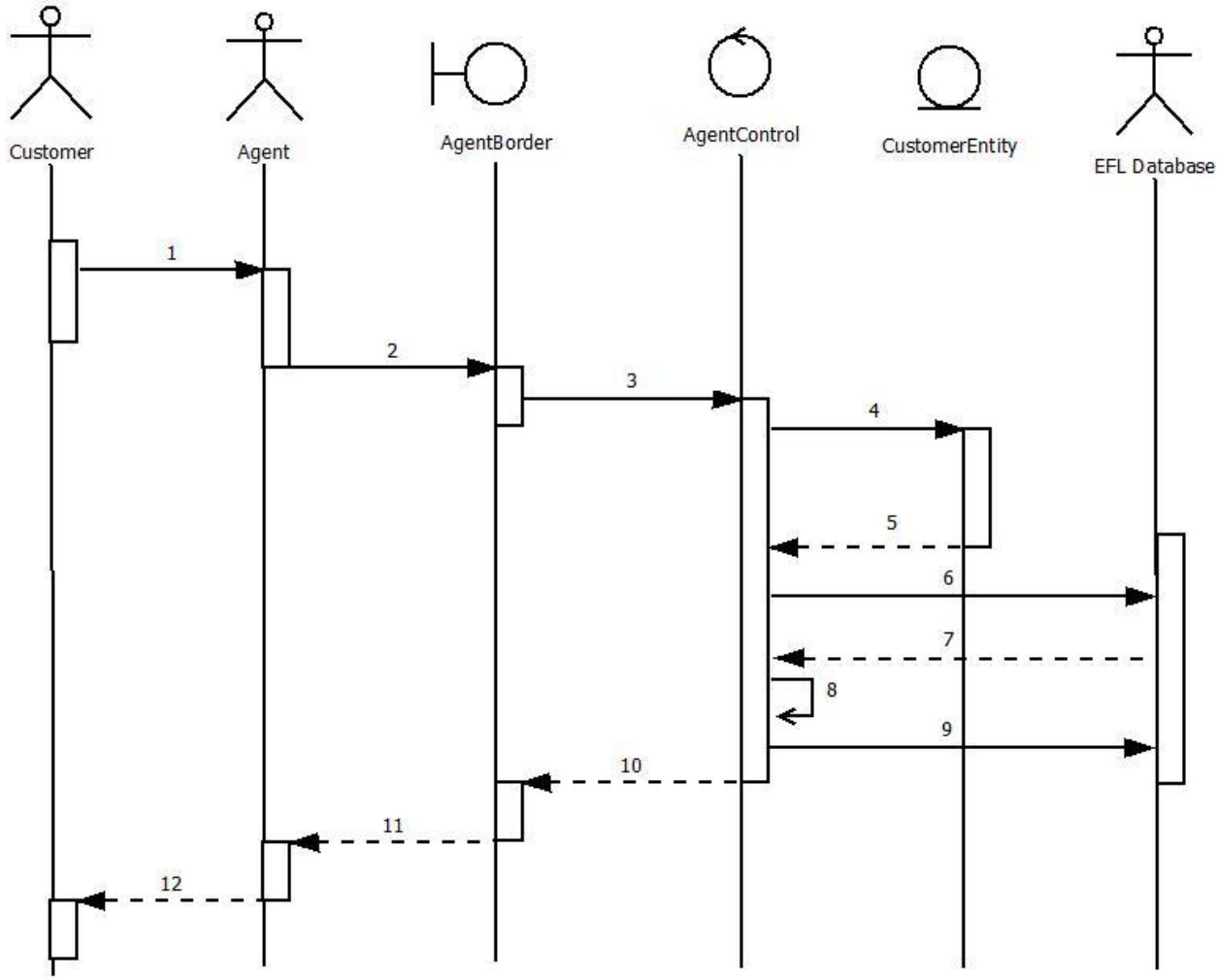
Sequence Diagram – Create New Customer Profile Scenario (Alt 3B)

1. New Customer calls or enters the travel agency and would like to create an account
2. Agent explains fee structure and requests information from customer
3. Customer rejects services

Sequence Diagram – Create New Customer Profile Scenario (Alt 5B)

3. Customer accepts services and provides information and credit card
4. Agent enters information into CreateAccount subscreen
5. Agent Verifies information is correct and hits submit to send information to AgentControl
6. AgentControl sends credit card info to be verified by Credit Bureau
7. Credit Information is invalid or unsatisfactory
8. AgentControl sends updated display to AgentBorder with error message
9. AgentBorder updates display to Agent
10. Agent Reports a successfully created account

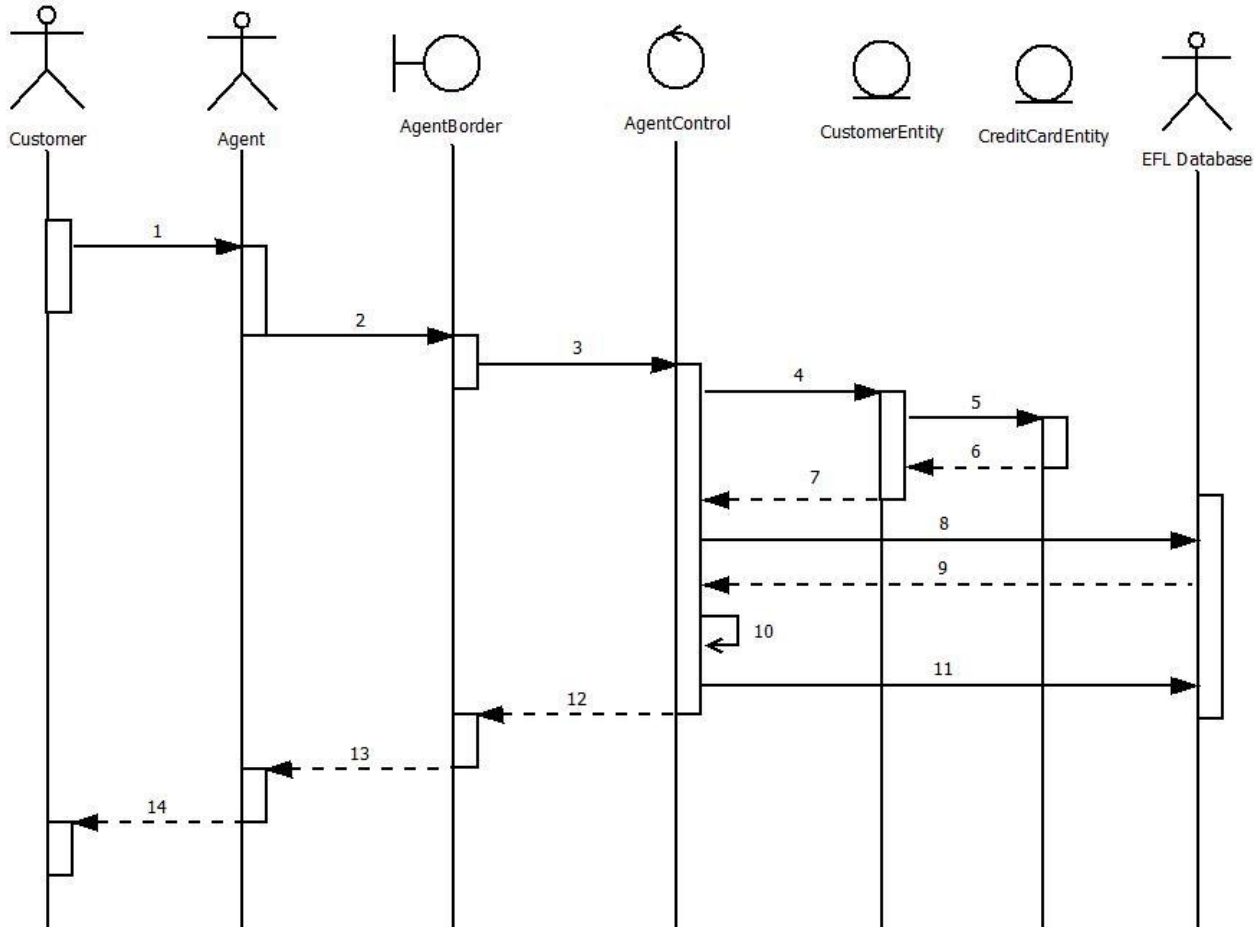
Add Credit to Customer Profile



Add Credit to Customer Profile

1. Customer brings in referral or Manger sees reason to provide credit to customer account
2. Agent requests to add credit to Customer Account giving the customers profile number
3. Agent hits submit and AgentBorder forwards profile number to AgentControl
4. AgentControl calls for a new instance of CustomerEntity
5. CustomerEntity returns a new instance of itself
6. AgentControl requests for customer information from EFL Database
7. EFL Database returns customer information
8. AgentControl updates CustomerEntity with customer information and updates credit in account
9. AgentControl sends information to the EFL Database
10. AgentControl reports successful operation
11. AgentBorder updates display to report successful operation
12. Agent informs Customer the credit was adde

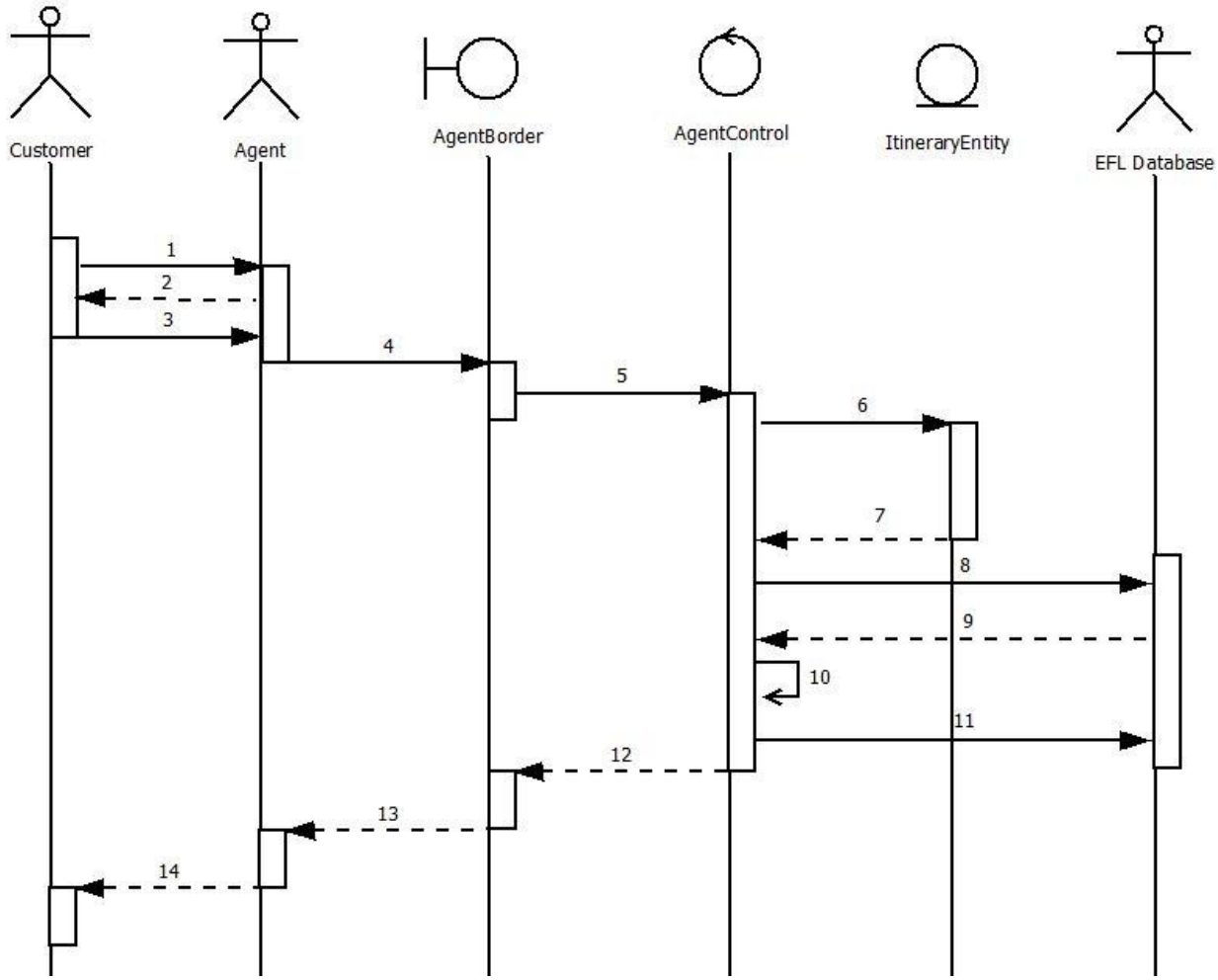
Manage Existing Customer Profile



Manage Existing Customer Profile(Normal)

1. Customer needs Agent to retrieve or update their profile with some new information
2. Agent inputs all necessary information into the terminal
3. Agent Verifies info and hits send and passes information from AgentBorder to AgentControl
4. AgentControl requests a new instance of the CustomerEntity class
5. CustomerEntity requests a new instance of the CreditCardEntity class
6. CreditCardEntity returns a new instance of itself
7. CustomerEntity returns a new instance of itself
8. AgentControl requests current customer information based on the customer ID or name
9. EFL Database returns information
10. AgentControl loads information into the new classes
11. AgentControl sends updated information to EFLDatabase
12. AgentControl returns updated information to the AgentBorder class
13. AgentBorder updates the terminal screen to reflect updated information
14. Agent reports a successful update to Customer

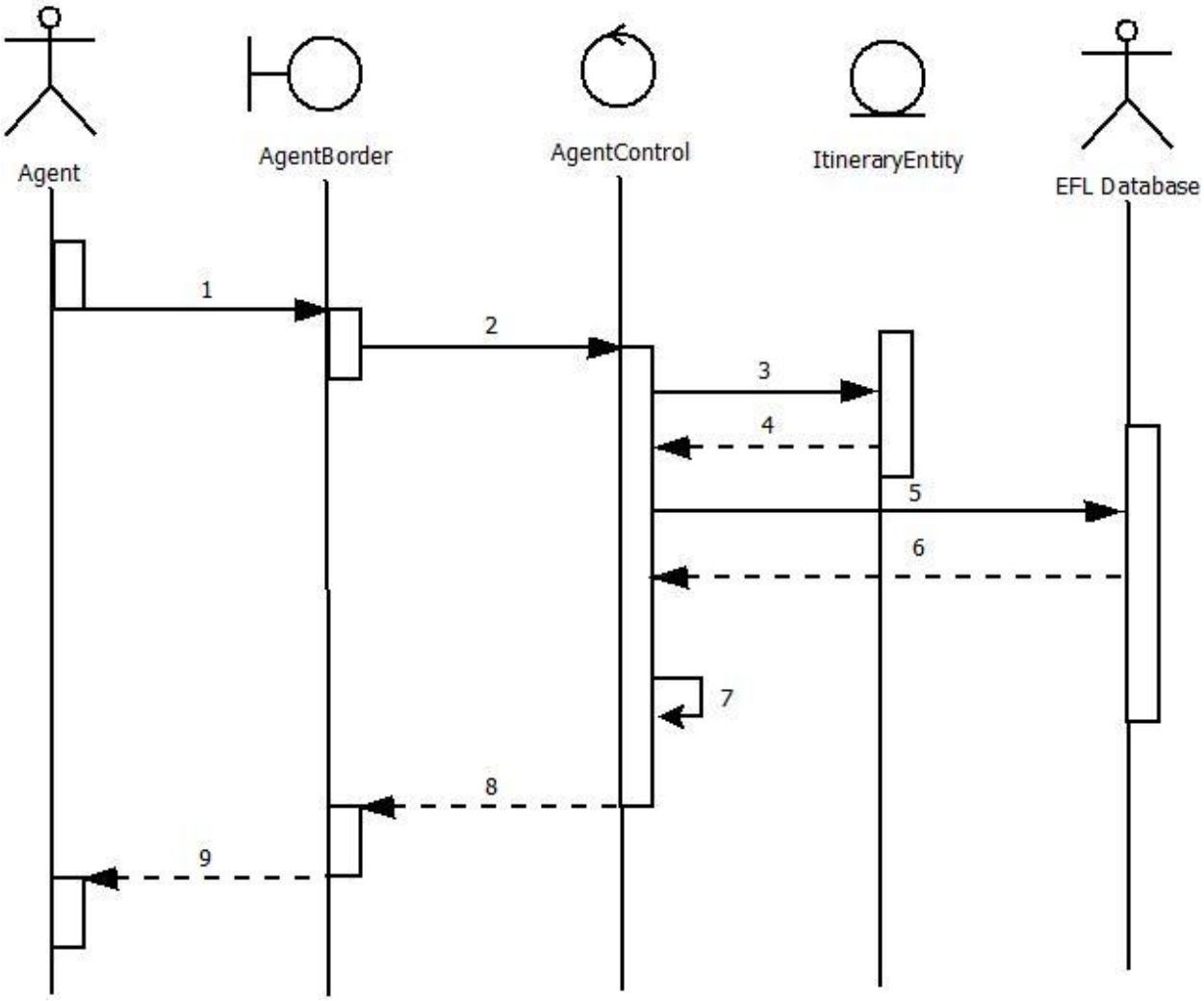
Create New Itinerary Case



Create New Itinerary Case

1. The Customer has requested to book a flight without a case already opened
2. The Agent requests information about the flight from the customer
3. The Customer gives the Agent the needed informations
4. The Agent Inputs all the information in the AgentBorder Terminal
5. The agent verifies the info is correct and hits submit sending the info to AgentControl
6. AgentControl calls to instantiate a new instance of ItineraryEntity
7. ItineraryEntity returns a new instance of itself to AgentControl
8. AgentControl requests for all matching flight information from EFLDatabase
9. EFLDatabase returns a text document containing all the info for the Itinerary
10. AgentControl updates all needed info into the ItineraryEntity
11. AgentControl sends Itinerary class to the EFLDatabase
12. AgentControl returns class instance to AgentBorder
13. AgentBorder updates info on the terminal
14. Agent Reports Back a successful Itinerary Creation

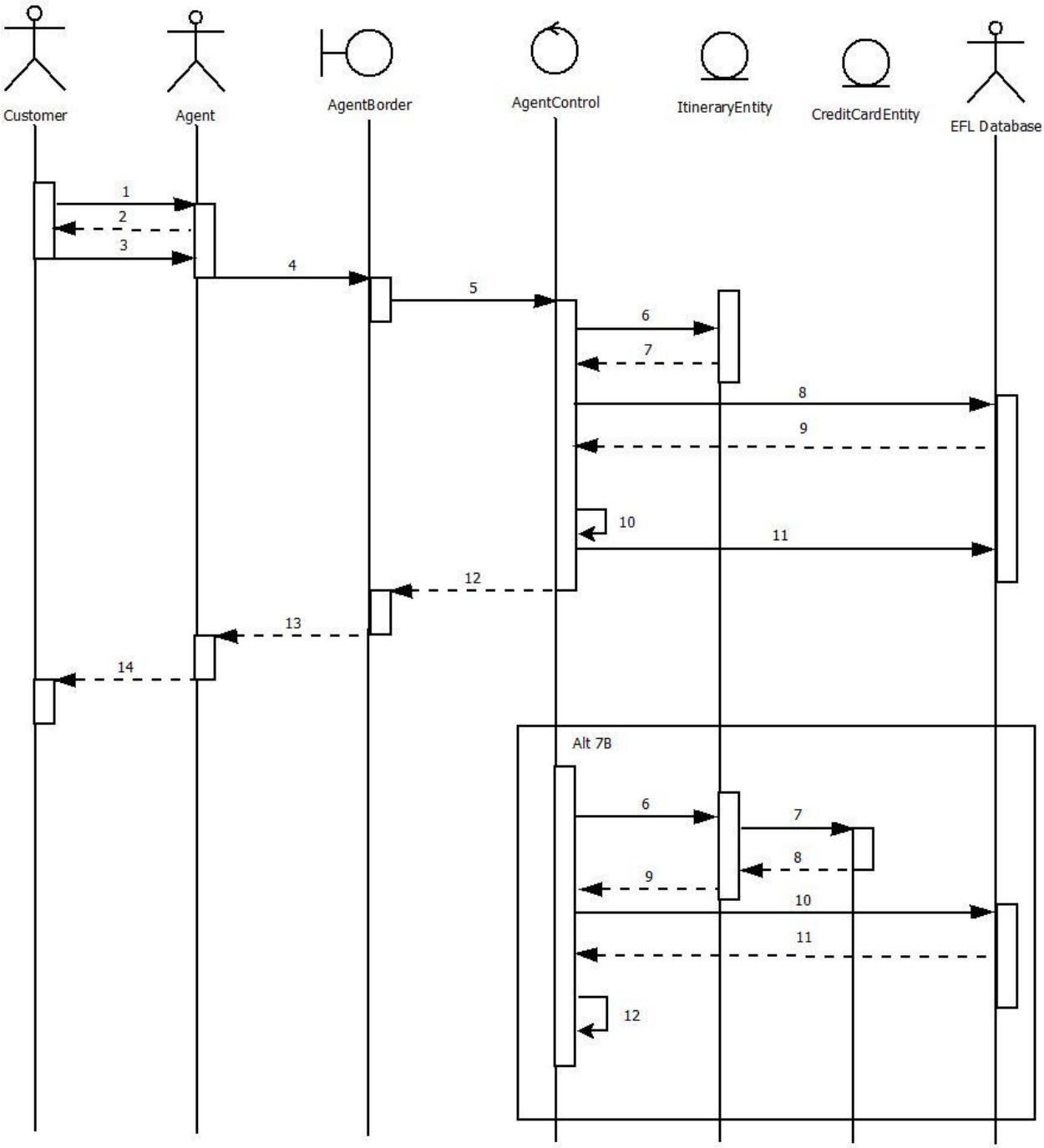
Produce Flight List



Produce Flight List(Normal)

1. Agent sends request to produce flight list by inserting customer and itinerary number
2. Agent clicks submit and AgentBorder forwards info to AgentControl
3. AgentControl creates new instance of ItineraryEntity
4. ItineraryEntity returns a new instance of itself
5. AgentControl sends a request for Itinerary info from the EFL Database
6. EFL Database returns information to the AgentControl
7. AgentControl updates the ItineraryEntity instance with data
8. Updates values in the AgentBorder class
9. AgentBorder updates its values in the view to display back

Reserve Flight



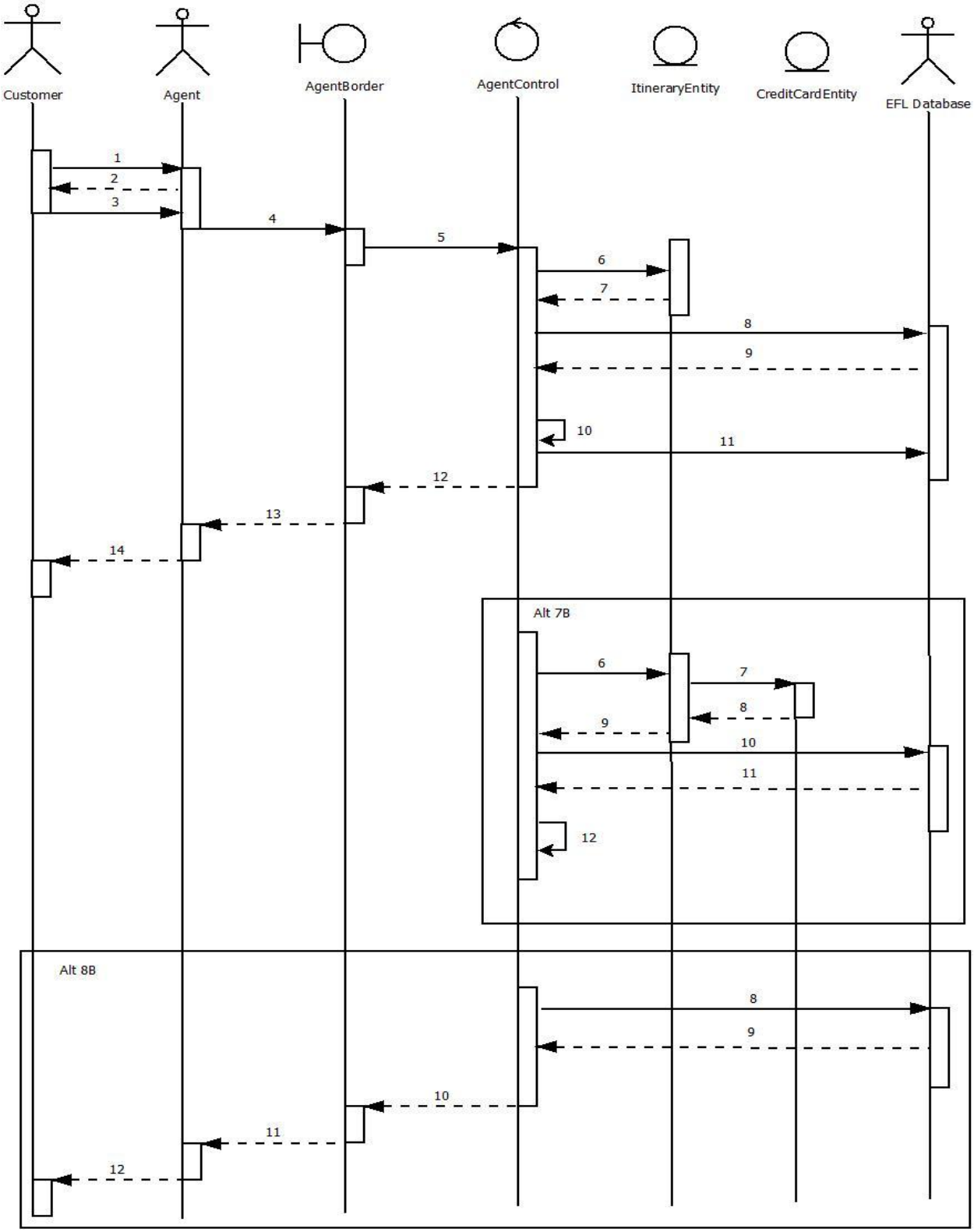
Reserve Flight(Normal)

1. Customer Wants to make a flight reservation
2. Agent requests specific information about the flight
3. Customer provides Agent with the information
4. The Agent Inputs all the information in the AgentBorder Terminal
5. The agent verifies the info is correct and hits submit sending the info to AgentControl
6. AgentControl calls to instantiate a new instance of ItineraryEntity
7. ItineraryEntity returns a new instance of itself to AgentControl
8. AgentControl requests for the saved Itinerary for customer from EFLDatabase
9. EFLDatabase returns a text document containing all the info for the Itinerary
10. AgentControl updates all needed info into the ItineraryEntity
11. AgentControl sends Itinerary class to the EFLDatabase
12. AgentControl returns class instance to AgentBorder
13. AgentBorder updates info on the terminal
14. Agent Reports Back a successful Itinerary Creation

Reserve Flight(7B)

6. AgentControl calls to instantiate a new instance of ItineraryEntity
7. ItineraryEntity calls a new instance of the CreditCardEntity
8. CreditCardEntity returns a new instance of itself
9. ItineraryEntity returns a new instance of itself to AgentControl
10. AgentControl requests for the saved Itinerary for customer from EFLDatabase
11. EFLDatabase returns a text document containing all the info for the Itinerary
12. AgentControl updates all needed info into the ItineraryEntity

Manage Flight



Manage Flight(Normal)

1. Customer Wants to make a flight reservation
2. Agent requests specific information about the flight
3. Customer provides Agent with the information
4. The Agent Inputs all the information in the AgentBorder Terminal
5. The agent verifies the info is correct and hits submit sending the info to AgentControl
6. AgentControl calls to instantiate a new instance of ItineraryEntity
7. ItineraryEntity returns a new instance of itself to AgentControl
8. AgentControl requests for the saved Itinerary for customer from EFLDatabase
9. EFLDatabase returns a text document containing all the info for the Itinerary
10. AgentControl updates all needed info into the ItineraryEntity
11. AgentControl sends Itinerary class to the EFLDatabase
12. AgentControl returns class instance to AgentBorder
13. AgentBorder updates info on the terminal
14. Agent Reports Back a successful Itinerary Creation

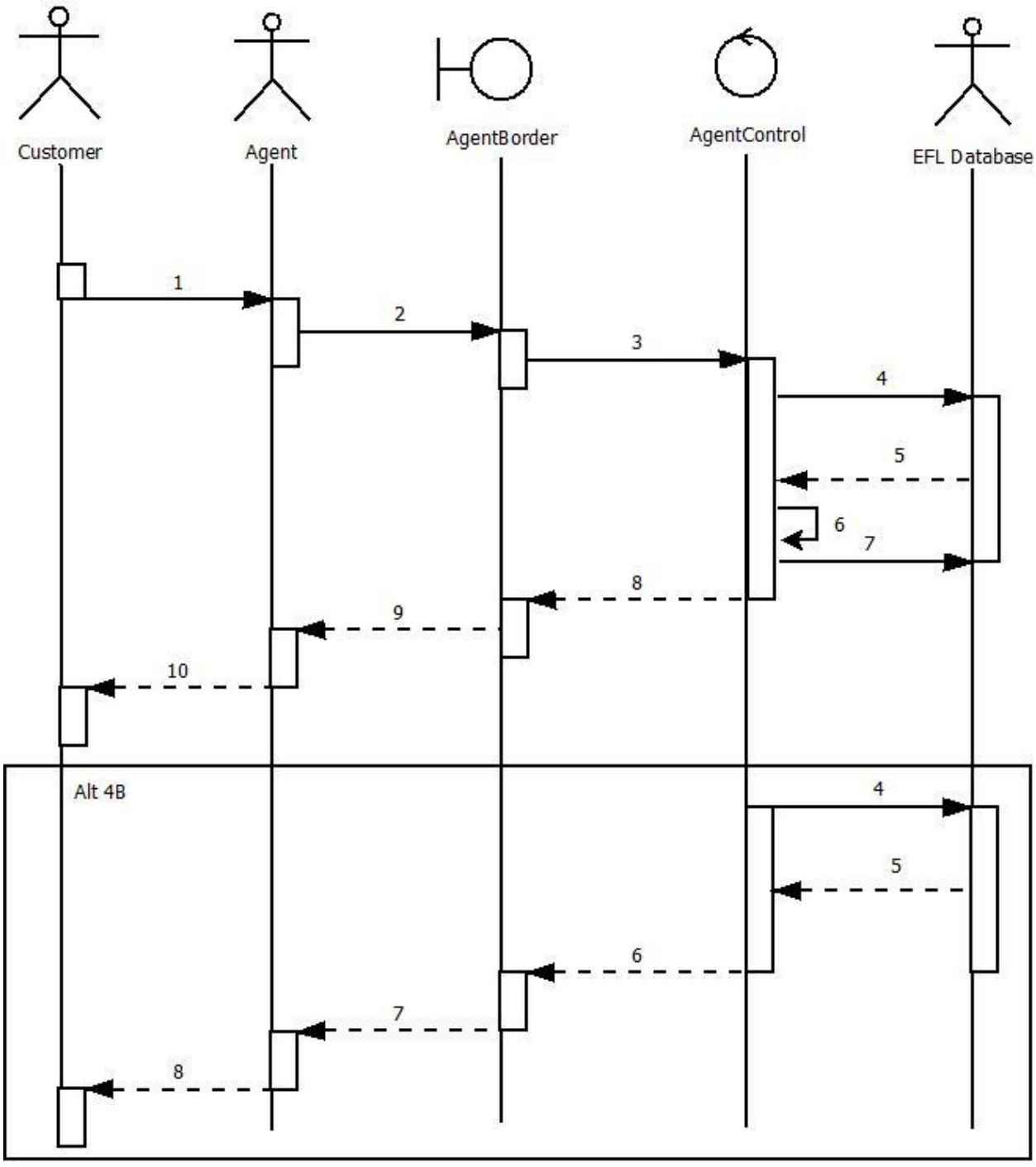
Manage Flight(7B)

6. AgentControl calls to instantiate a new instance of ItineraryEntity
7. ItineraryEntity calls a new instance of the CreditCardEntity
8. CreditCardEntity returns a new instance of itself
9. ItineraryEntity returns a new instance of itself to AgentControl
10. AgentControl requests for the saved Itinerary for customer from EFLDatabase
11. EFLDatabase returns a text document containing all the info for the Itinerary
12. AgentControl updates all needed info into the ItineraryEntity

Manage Flight(8B)*

8. AgentControl requests Itinerary case from EFL Database
9. EFL Database returns a null
10. AgentControl reports a failed operation to AgentBorder
11. AgentBorder refreshes display to reflect failed operation
12. Agent tells customer that the Itinerary doesn't exist and offers to create a new one

Cancel Reservation



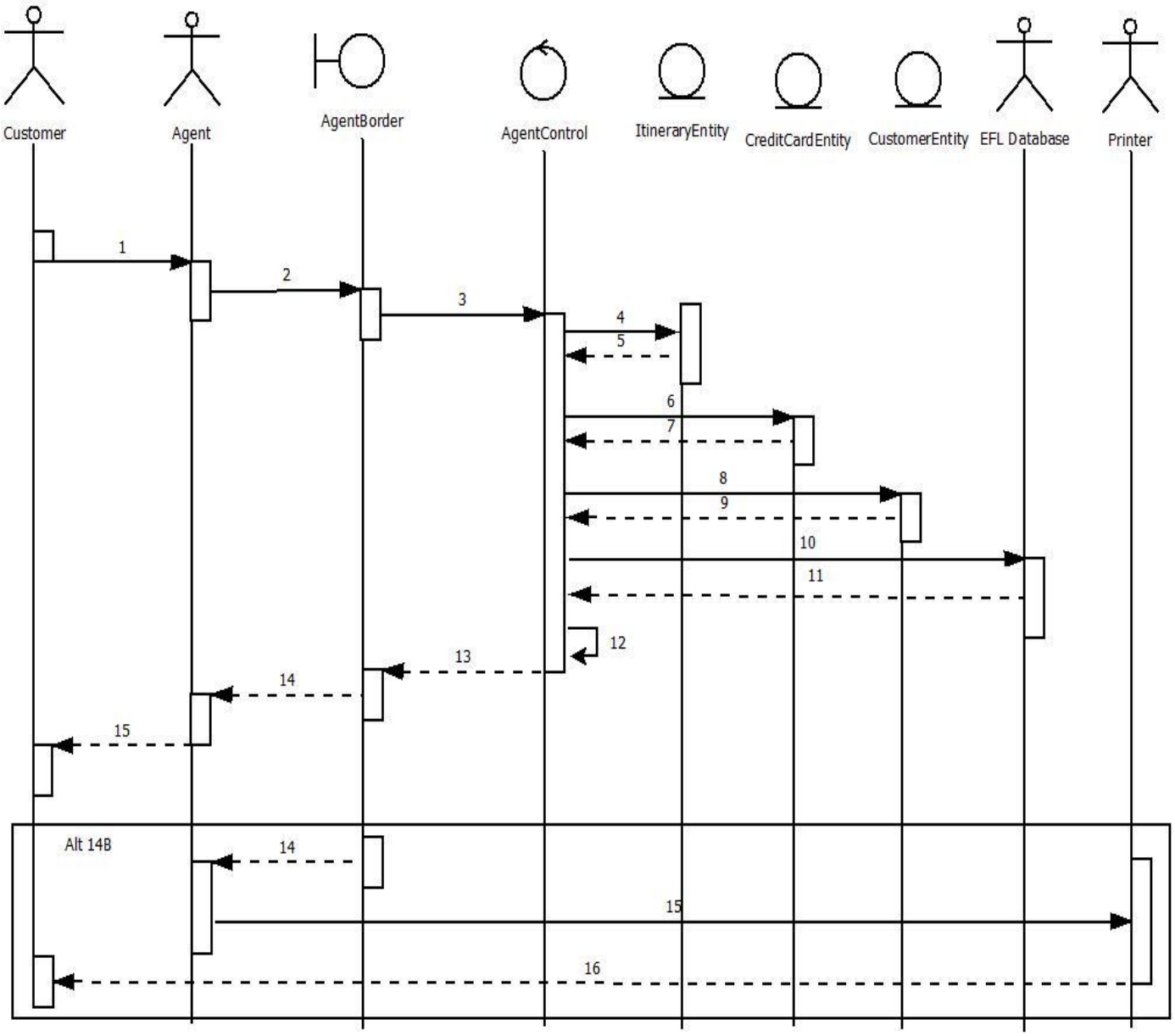
Cancel Reservation (Normal)

1. Customer Requests that their flight reservation is cancelled
2. Agent looks up the Itinerary to cancel by the customer and itinerary number in Agent Border
3. Agent submits request and information is forwarded to Agent Control
4. AgentControl retrieves information for Itineraries from EFL Database
5. EFL Database returns itinerary information
6. AgentControl deletes flight reservation from itinerary
7. AgentControl sends updated information to the EFL Database
8. AgentControl returns successful cancellation to AgentBorder
9. AgentBorder updates screen to reflect a successful cancellation to AgentBorder
10. Agent reports to Customer a successful

Manage Flight(4B)*

4. AgentControl requests Itinerary case from EFL Database
5. EFL Database returns a null
6. AgentControl reports a failed operation to AgentBorder
7. AgentBorder refreshes display to reflect failed operation
8. Agent tells customer that the Itinerary doesn't exist and offers to create a new one

Produce Flight Receipt



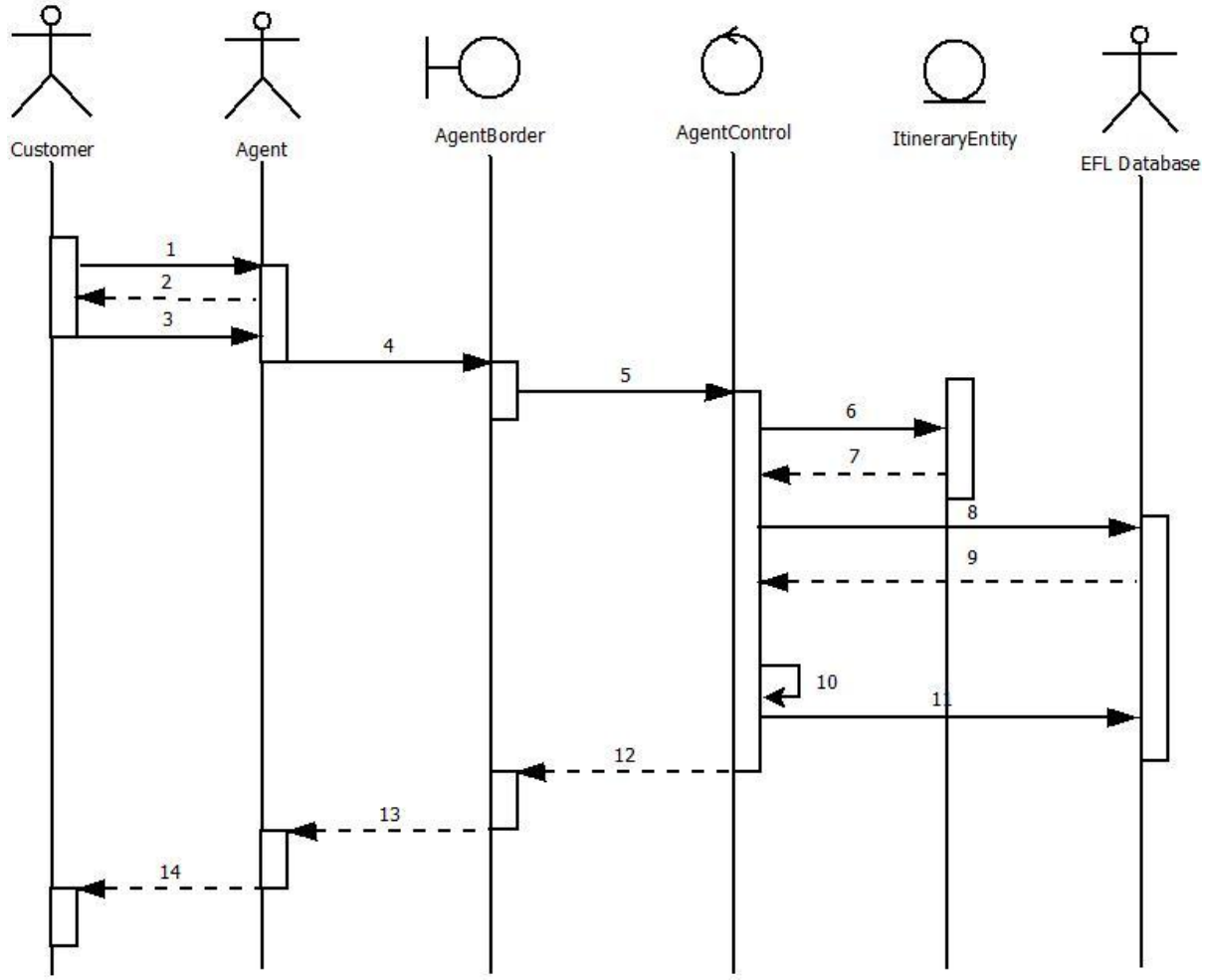
Produce Flight Receipt (Normal)

1. Customer Requests to print their flight reservation
2. Agent looks up the Itinerary to print by the customer and itinerary number in Agent Border
3. AgentBorder forwards information to AgentControl
4. AgentControl calls ItineraryEntity to instantiate a new instance of itself
5. ItineraryEntity returns a new instance of itself
6. AgentControl calls CreditCardEntity to instantiate a new instance of itself
7. CreditCardEntity returns a new instance of itself
8. AgentControl calls CustomerEntity to instantiate a new instance of itself
9. CustomerEntity returns a new instance of itself
10. AgentControl requests information for Itineraries from EFL Database
11. EFL Database returns itinerary information
12. AgentControl sets information to get returned as document flight reservation from itinerary
13. AgentControl returns document to AgentBorder
14. AgentBorder updates screen with document to AgentBorder
15. Agent sends soft copy to customer via email

Produce Flight Receipt (14B)*

14. AgentControl returns document to AgentBorder
15. AgentBorder displays document and Agent chooses print
16. Printer prints the flight Receipt to the customer

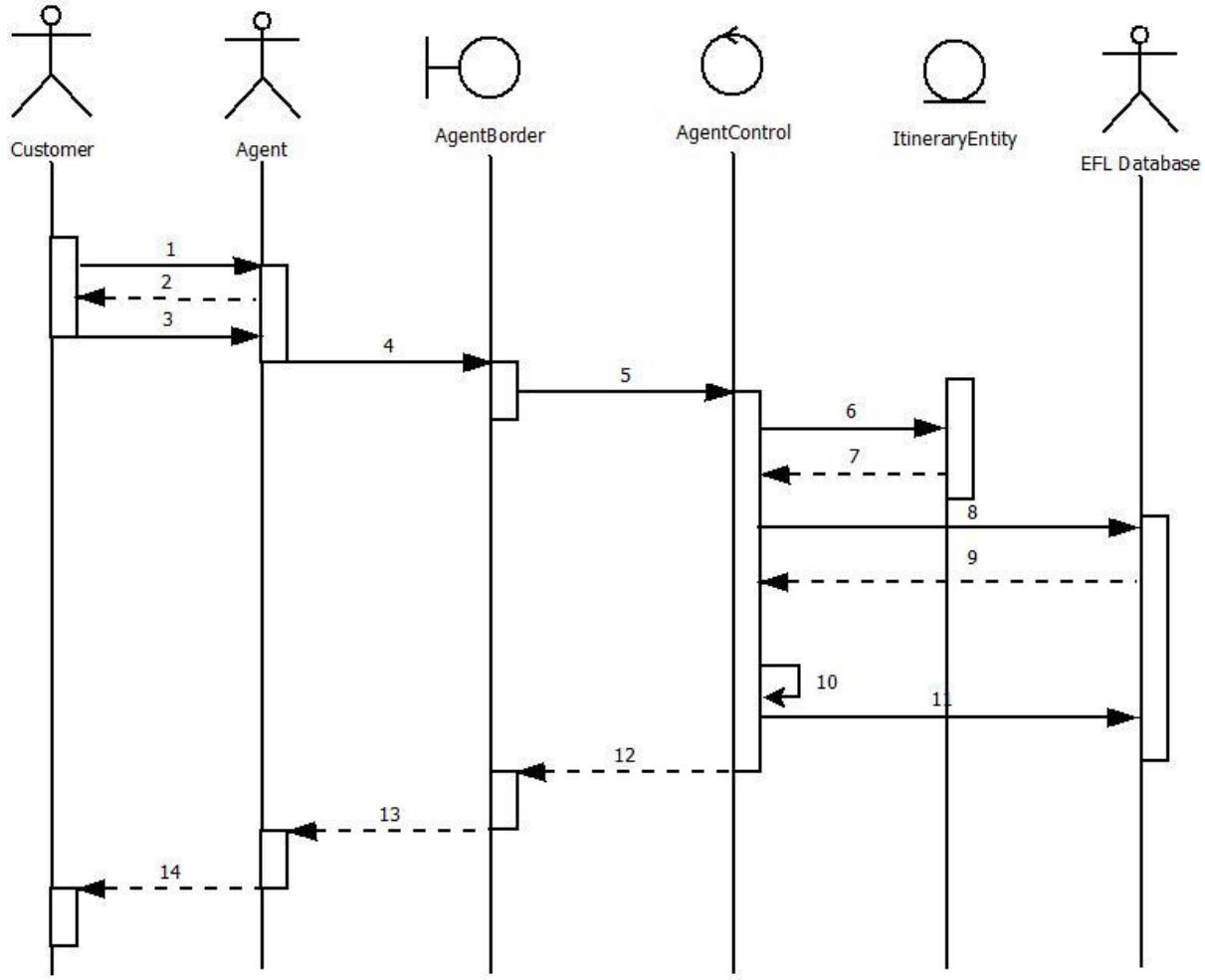
Create Price Watch



Create Price Watch(Normal)

1. The Customer doesn't like the prices for their flights
2. The Agent offers to create a price watch and requests their desired price
3. The Customer gives the Agent the needed information
4. The Agent Inputs all the information in AgentBorder
5. The agent verifies the info is correct and hits submit sending the info to AgentControl
6. AgentControl calls to instantiate a new instance of ItineraryEntity
7. ItineraryEntity returns a new instance of itself to AgentControl
8. AgentControl requests for all matching flight information from EFLDatabase
9. EFLDatabase returns a text document containing all the info for the Itinerary
10. AgentControl updates the price watch info into the ItineraryEntity instance
11. AgentControl sends Itinerary Case with updated price watch to the EFLDatabase
12. AgentControl returns class instance to AgentBorder
13. AgentBorder updates info on the terminal
14. Agent Reports Back a successful price watch creation

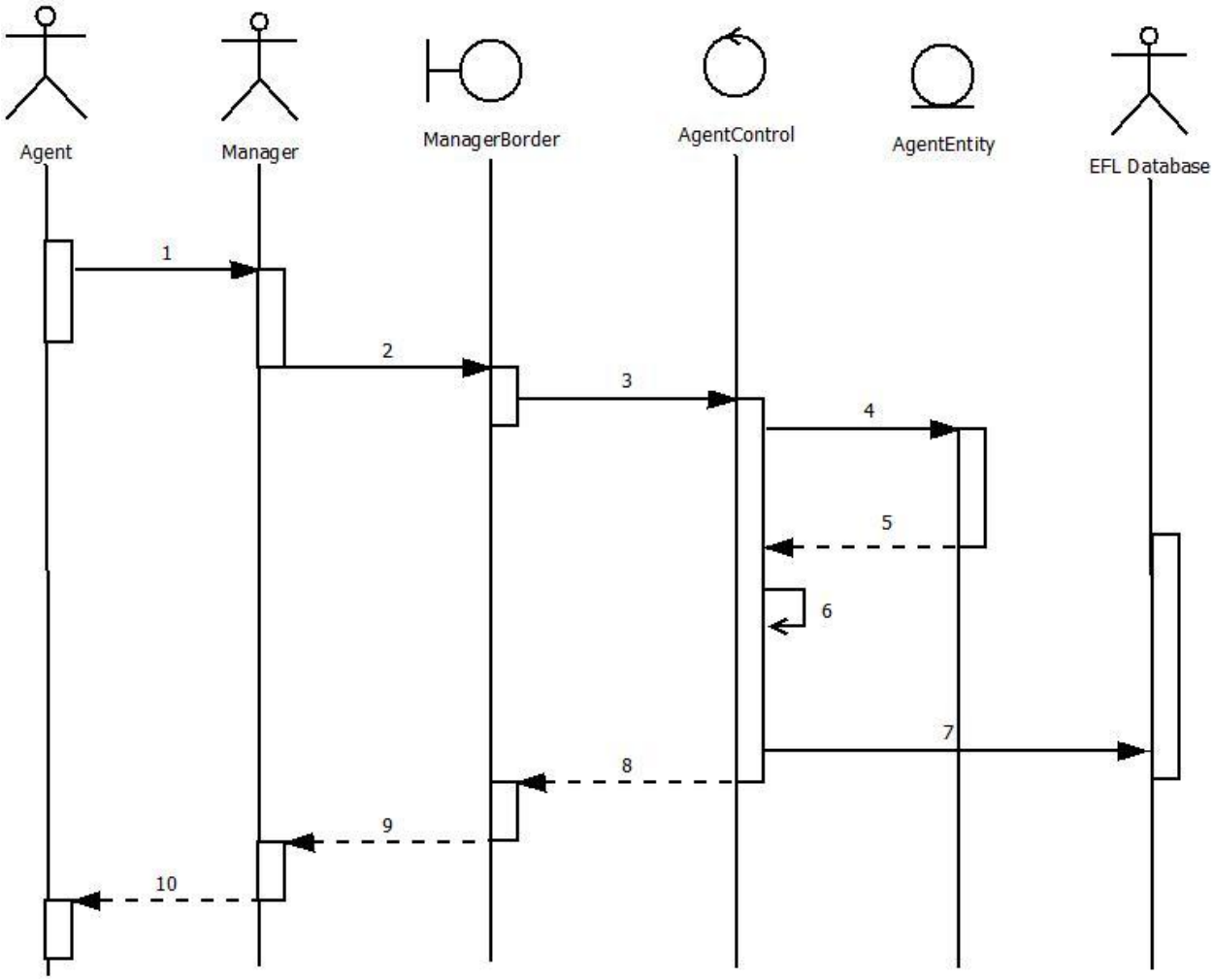
Cancel Price Watch



Cancel Price Watch(Normal)

1. The Customer Requests to book a price watch flight or wants to cancel price watch
2. The Agent requests the itinerary number
3. The Customer gives the Agent the needed information
4. The Agent Inputs all the information in AgentBorder
5. The agent verifies the info is correct and hits submit sending the info to AgentControl
6. AgentControl calls to instantiate a new instance of ItineraryEntity
7. ItineraryEntity returns a new instance of itself to AgentControl
8. AgentControl requests for all matching flight information from EFLDatabase
9. EFLDatabase returns a text document containing all the info for the Itinerary
10. AgentControl cancels the price watch to the ItineraryEntity instance
11. AgentControl sends Itinerary Case with updated information to the EFLDatabase
12. AgentControl returns class instance to AgentBorder
13. AgentBorder updates info on the terminal
14. Agent Reports Back a successful price watch cancelation

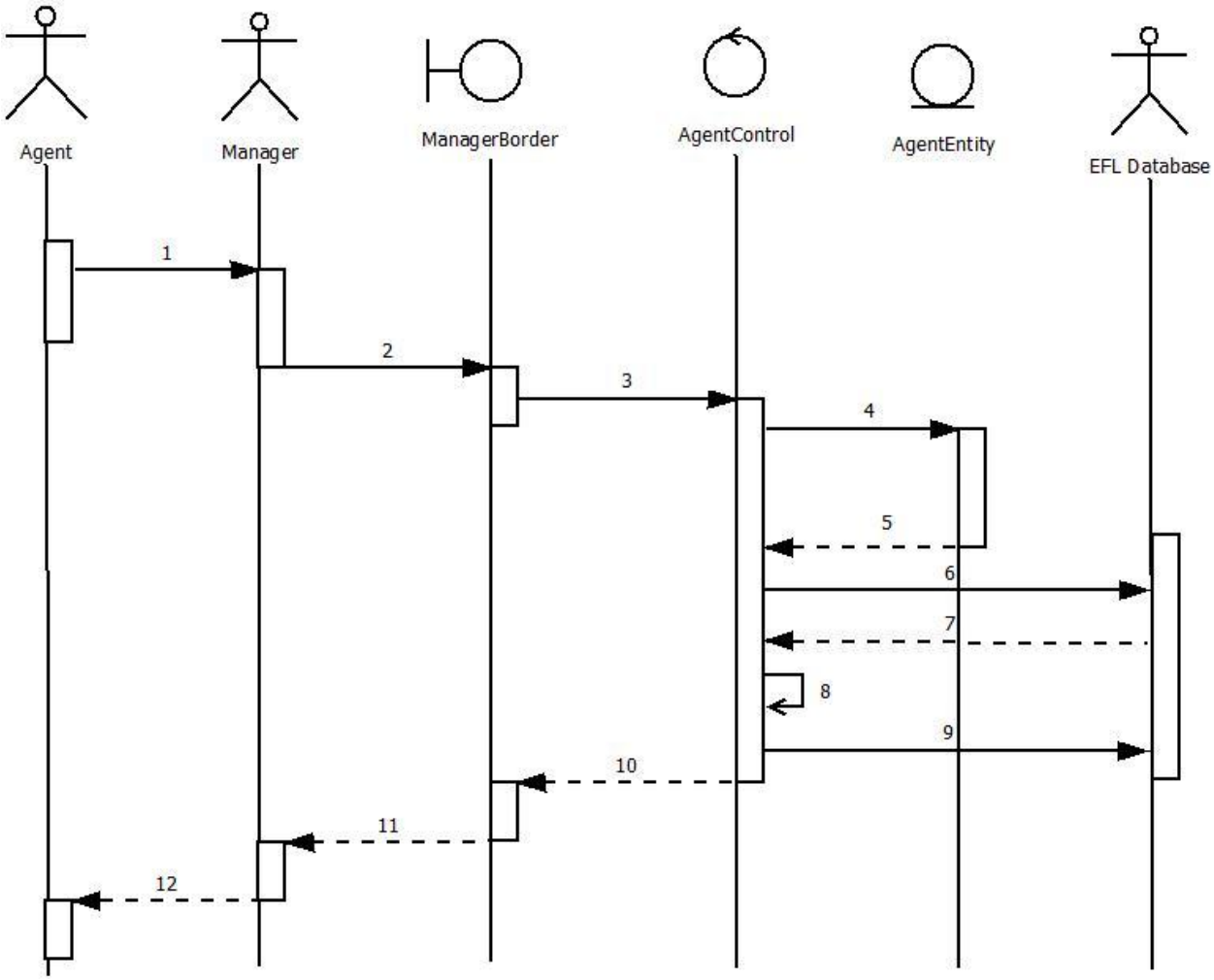
Create Agent Account



Creat Agent Account(Normal)

1. Agent is a new hire and gives manager information to create an agent account
2. Manager enters information into ManagerBorder subscreen
3. Manager Verifies information is correct and hits submit to send information to AgentControl
4. AgentControl calls to create a new instance of AgentEntity
5. AgentEntity returns a new instance of itself to AgentControl
6. AgentControl updates the new class with current info
7. AgentControl sends agent info to EFLDatabase
8. AgentControl sends updated display to AgentBorder with customer instance
9. ManagerBorder updates display to Manager
10. Manager Reports a successfully created account

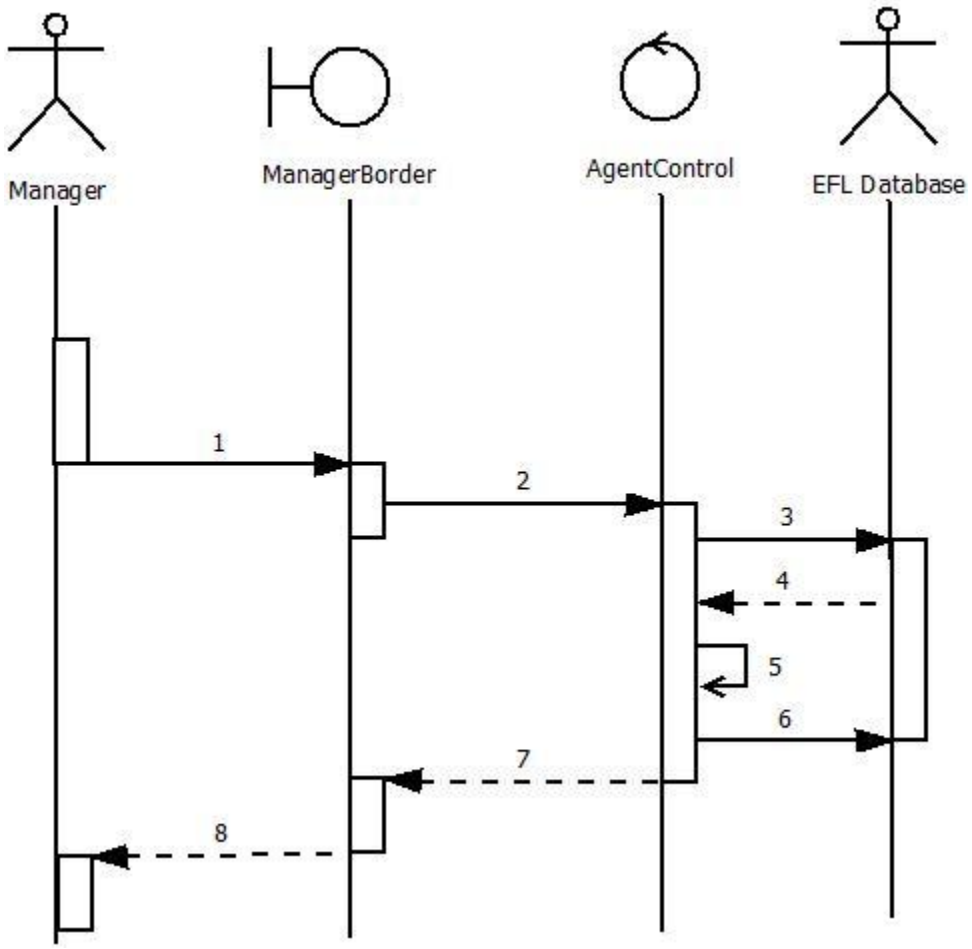
Manage Agent Account



Manage Agent Account(Normal)

1. Agent gives manager updated information to change their account
2. Manager enters information into ManagerBorder subscreen
3. Manager Verifies information is correct and hits submit to send information to AgentControl
4. AgentControl calls to create a new instance of AgentEntity
5. AgentEntity returns a new instance of itself to AgentControl
6. AgentControl requests saved info for Agent from EFL Database
7. EFl Database returns information to AgentControl
8. AgentControl updates the new class with current info
9. AgentControl sends agent info to EFLDatabase
10. AgentControl sends updated display to AgentBorder with customer instance
11. ManagerBorder updates display to Manager
12. Manager Reports a successfully created account

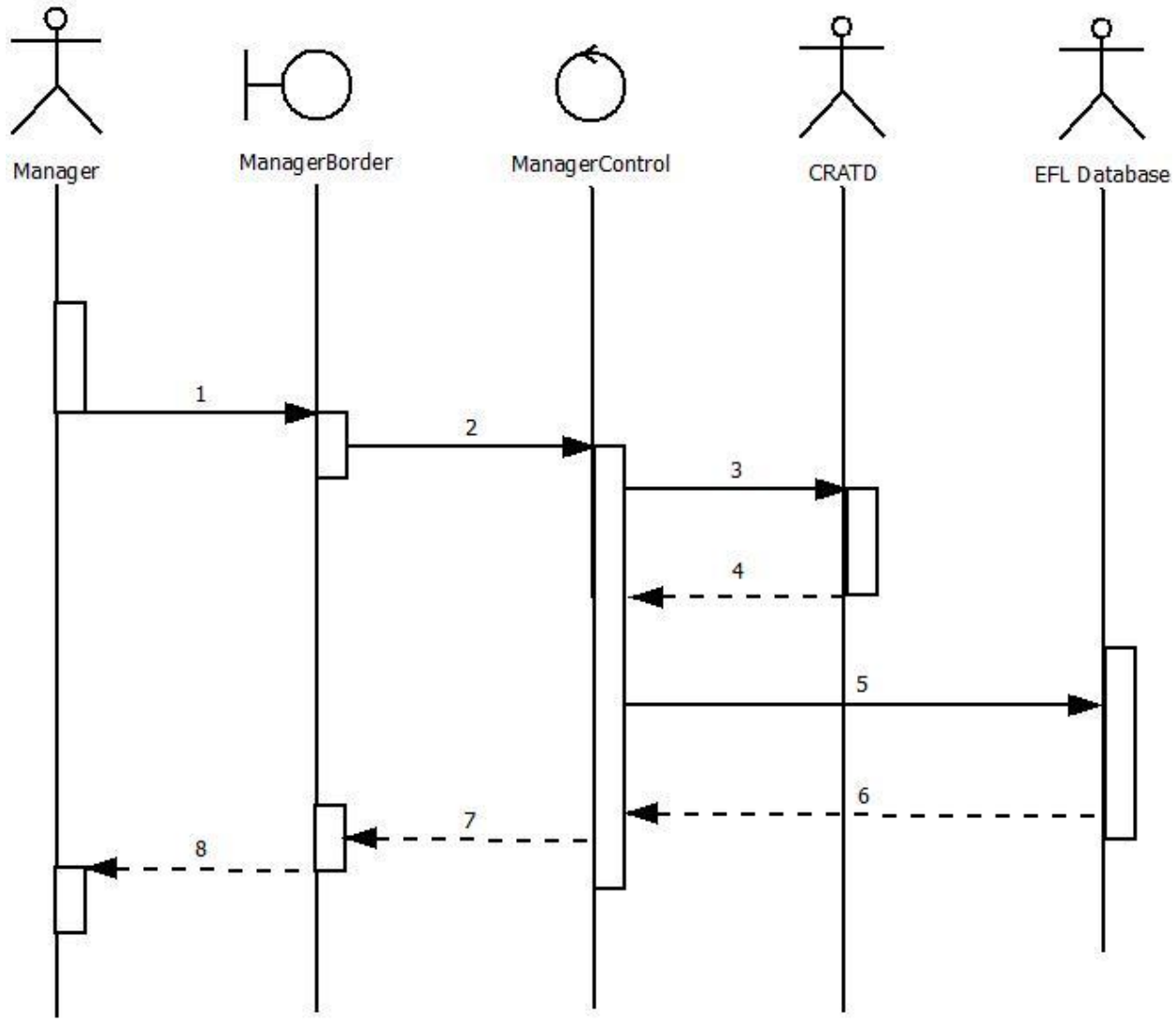
Produce Daily Reports



Produce Daily Reports(Normal)

1. Manager requests for daily reports to be computed
2. ManagerBorder will pass action on to AgentControl
3. AgentControl gets information from EFL Database
4. EFL Database returns requested information
5. AgentControl perfoms computations
6. Updated reports are updated to the EFL Database
7. AgentControl returns information from Daily reports
8. ManagerBorder updates terminal screen to display reports.

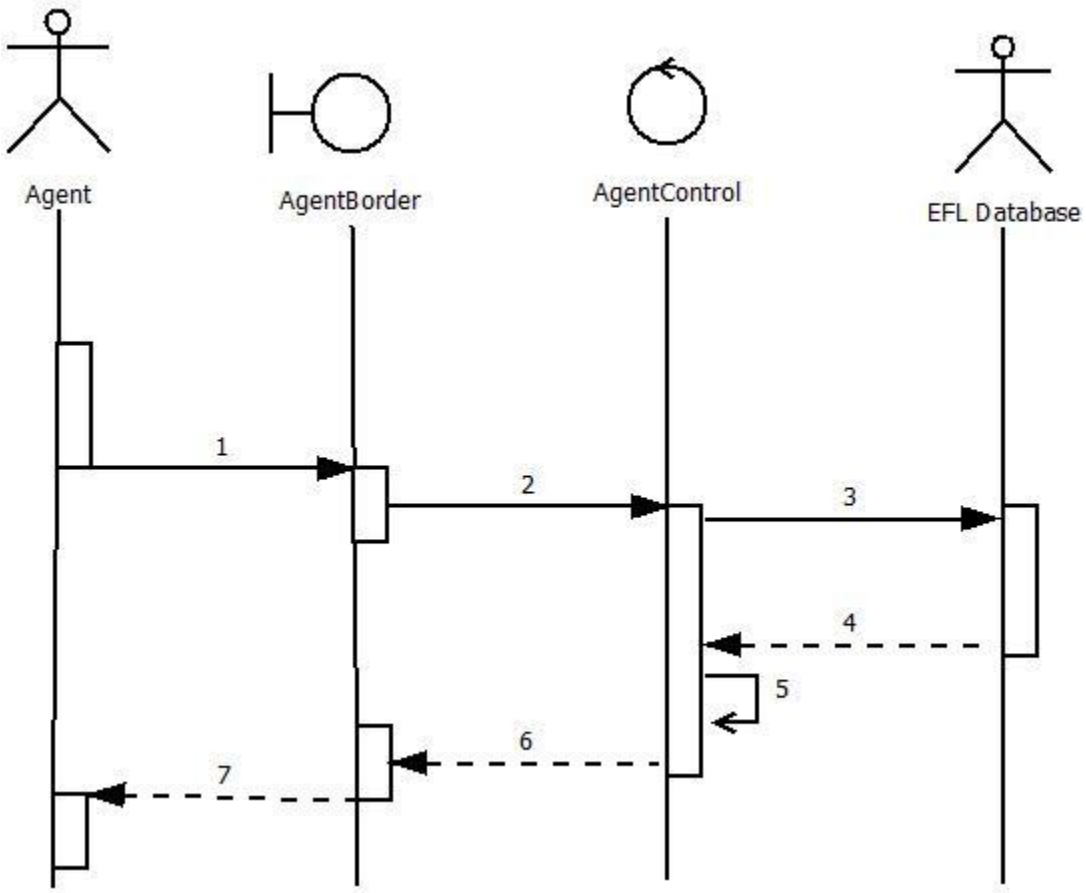
Get Updates From CRATD



Get Updates From CRATD (Normal)

1. Manager initiates downloading of updates from CRATD
2. Manager sends an update request signal from ManagerBorder to MangerControl
3. ManagerControl gets download information from the CRATD
4. CRATD returns updates to the ManagerControl
5. ManagerControl writes updates to the EFL Database
6. EFL Database returns success signal to the ManagerControl
7. ManagerControl returns success signal to the ManagerBorder
8. ManagerBorder returns success signal to the Manager

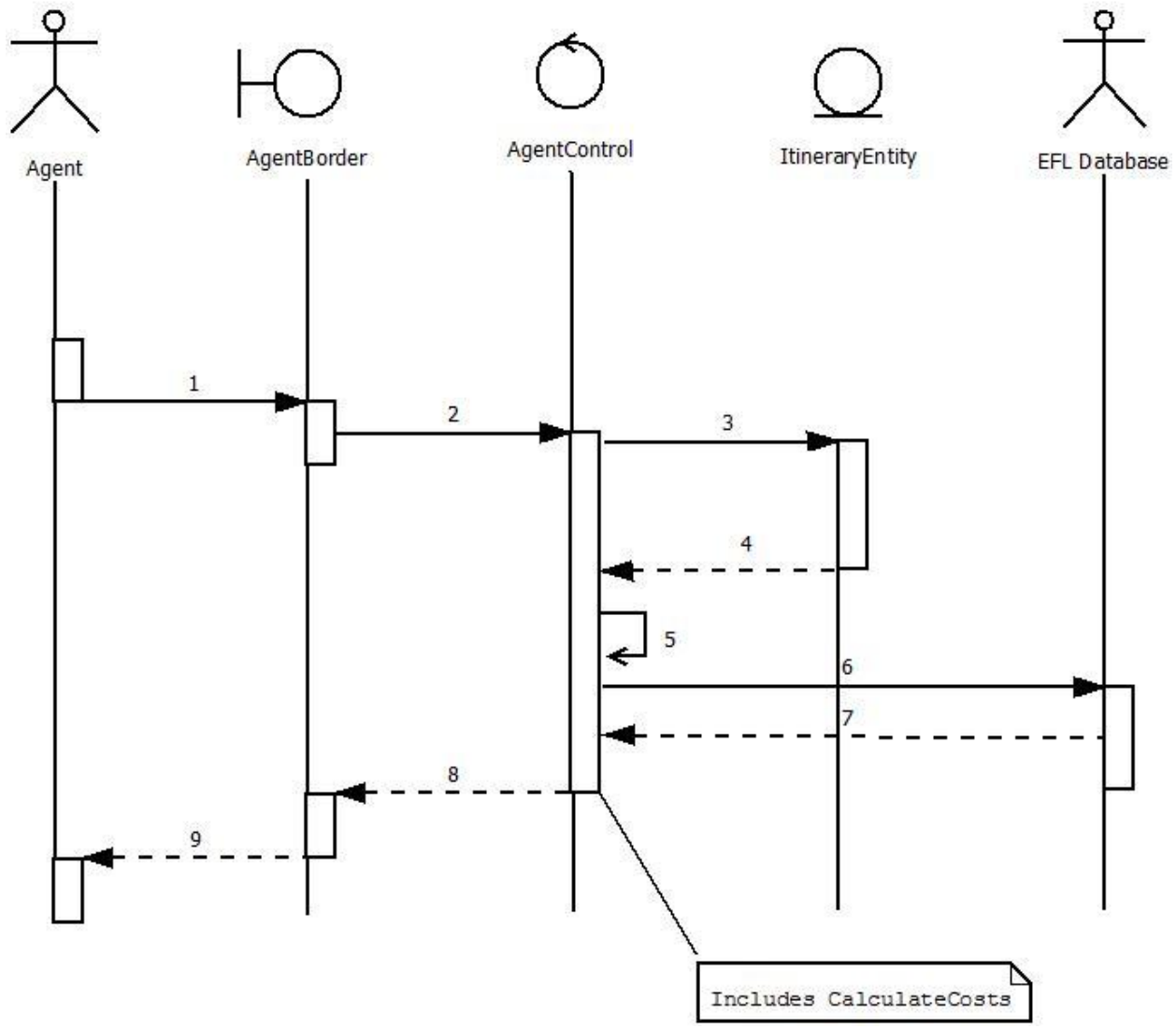
Calculate Costs



Calculate Costs (Normal)

1. Agent requests for a calculate costs action by submitting flight information into AgentBorder
2. AgentBorder forwards information to AgentControl
3. AgentControl requests for flight information from EFL Database
4. EFL Database returns information to AgentControl
5. AgentControl Computes Costs
6. AgentControl returns updated information to AgentBorder
7. AgentBorder updates display

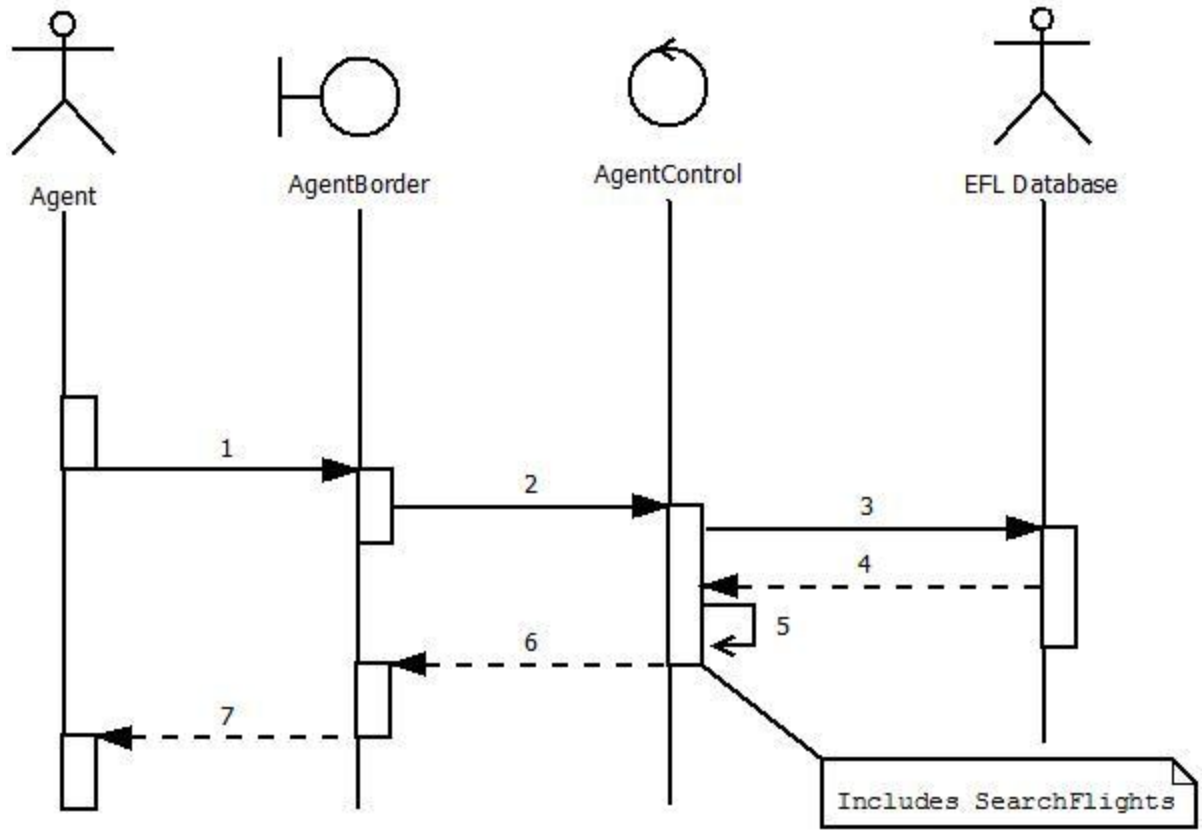
Search Flights



Search Flights (Normal)

1. Agent Requests for a list of current flights that matches an Itinerary given by the Itinerary number
2. Agent hits submit and AgentBorder forwards the information to AgentControl
3. AgentControl calls for a new instance of ItineraryEntity
4. ItineraryEntity returns a new instance of itself to AgentControl
5. AgentControl gets all the information needed for the flights
6. AgentControl then requests for a list of flights from the EFL Database
7. EFL Database returns the list to AgentControl
8. AgentControl returns the list to AgentBorder
9. AgentBorder updates display to show flight list

Provide Met Watch Scenarios



Provide Met Watch Scenarios (Normal)

1. Agent requests all met Price Watch Scenarios to be listed
2. Agent hits submit and AgentBorder forwards request to AgentControl
3. AgentControl requests Price Watch List From EFL Database
4. EFL Database returns a list of all current Prices Watches
5. AgentControl Processes which price watches have been met
6. AgentControl returns a list of Met Price watches with flight information
7. AgentBorder updates display with list