Aram Serobyan
Vahagn Tumanyan
CS 312: OOAD
Armen Kostanyan
Vacation Tracking System

Vacation Tracking System - 6

Primary use cases:

- **Making a vacation request**
  During this procedure an employee makes a vacation request for a certain period of time. System checks whether it can be met (for example, it might be a restriction on maximum number of employees of a given department that can be on vacation at the same time). If the vacation request does not contradict the company's policy, System creates a vacation request with a unique ID and adds it to a catalog of vacation requests.
- **A vacation request cancelling**
  During this procedure a vacation request is removed from the catalog of vacation requests.
- **Vacations initiating**
  During this procedure the vacation request objects with starting date coinciding with current date become vacation objects and shifted to a vacations catalog. The vacation catalog thus keeps the list of employees that are currently on vacation.
- **Vacations completion**
  During this procedure all vacation objects with completion date coinciding with current date are become archive records and added to an archive. This way System keeps information about the vacations passed.

Recommended classes:

Company,

ListOfDepartments, Department,

ListOfEmployees, Employee,

CatalogOf VacationRequests, VacationRequest,

CatalogOfVacations, Vacation,

Archive, Record.

Use Case 1 : Making a vacation request

1. Stakeholders and Interests

Primary Actor: Employee

Stakeholders and interests:

-- Employee. Wants to fix a time interval for going to vacation.

-- Company. Wants to make sure Employee can go to vacation during that time interval and register time interval that employee intends to be on vacation.

2. Preconditions and Postconditions

Preconditions: Identified and authenticated employee with a knowledge of time interval he wants to go to vacation.

Postconditions: A vacationRequest was created for the employee, and he was provided with its ID.

3. Main Scenario.

1. Employee enters his Id and time interval that he intends to be on the vacation.
2. System checks if employee's request doesn't violate any policy. i.e
3. System takes the department name and employee ID and uses some strategy to determine avaibility
4. If vacation violates a policy system asks if employee wants to add it into the wishlisht.

Steps 1-4 are repeated until entered time interval doesn't violate any policy.

5. Employee asks system to creates a new entry in the vacationRequestCatalog.
6. System decreases employees unusedVacationDays by the amount of days in time interval.
7. System returns the generated Id for the requested vacation.

4. Extensions (or Alternative Flaws).

3a. Employee doesn't have enough vacation days.

1. System tells the employee to shorten the time interval and prints how many days he has. System doesn't suggest to add into the wishlisht in this case.

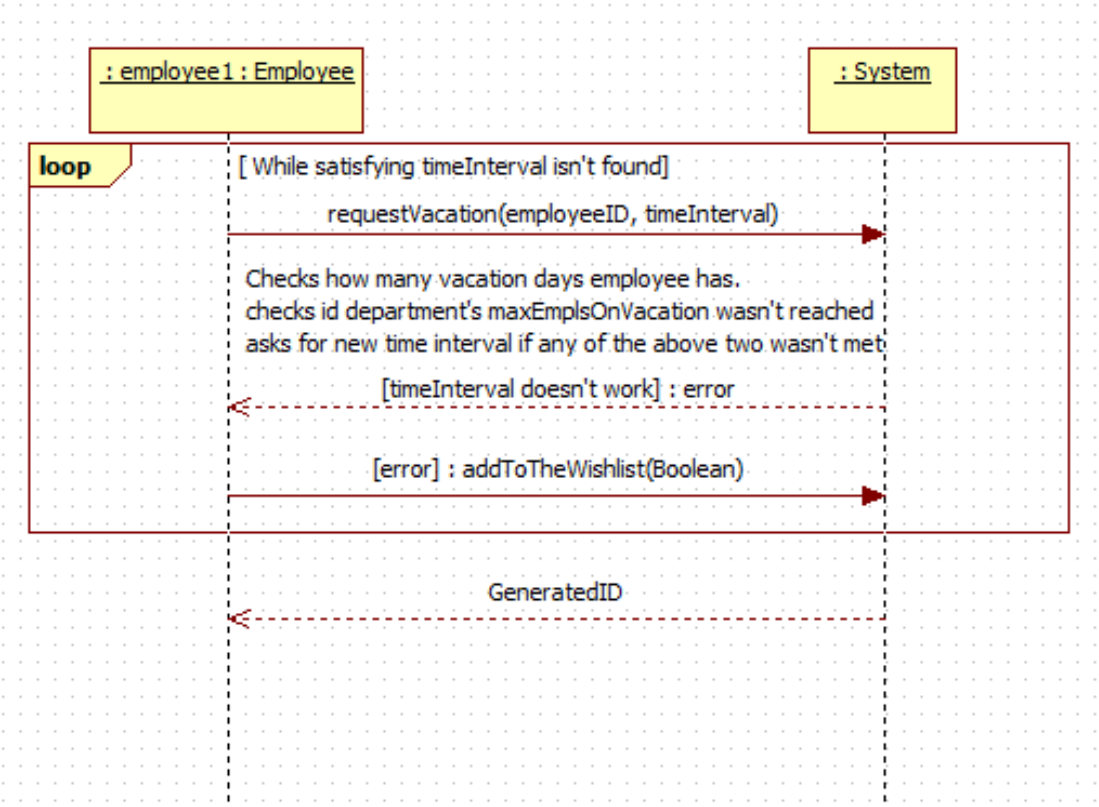
1a. If employee has 0 days System tell's that he can't request vacation as of now.

3b. The amount of people requested a vacation from the department is equal to the maximal amount of people who can simultaneously be in the vacation.

1. System tells the employee to change time interval.

5. Special Requirments

Access to the database.



Use Case 2 : A Vacation Request Cancelation

1. Stakeholders and Interests

Primary Actor: Employee

Stakeholders and interests:

-- Employee. Wants to cancel previously made vacation request

-- Company. Wants to delete the request from CatalogOfVacationRequests and add unused vacation days back to the employee.(or other information). Also wants to check the wishlisht for any vacations which became avaible.

2. Preconditions and Postconditions

Preconditions: Identified and authenticated employee with an id of made vacationRequest.

Postconditions: A vacationRequest was deleted from the CatalogOfVacationRequests, and employee's vacation days were updated. If any vacation became avaible an employee received an email.

3. Main Scenario.

1. Employee enters his Id and the Id of premade vacationRequest.
2. System finds the VacationRequest in the CatalogOfVacationRequests.
3. System updates employee's vacationDays by adding back the time interval in the vacationRequest
4. System removes VacationRequest from the CatalogOfVacationRequests.
5. System goes throught the wishlisht to see if any of Vacations there became avaible and sends email to the employees if any of them were.
6. System returns a successful completion message.

4. Extensions (or Alternative Flaws).

2a. No VacationRequest matched the provided requestID.

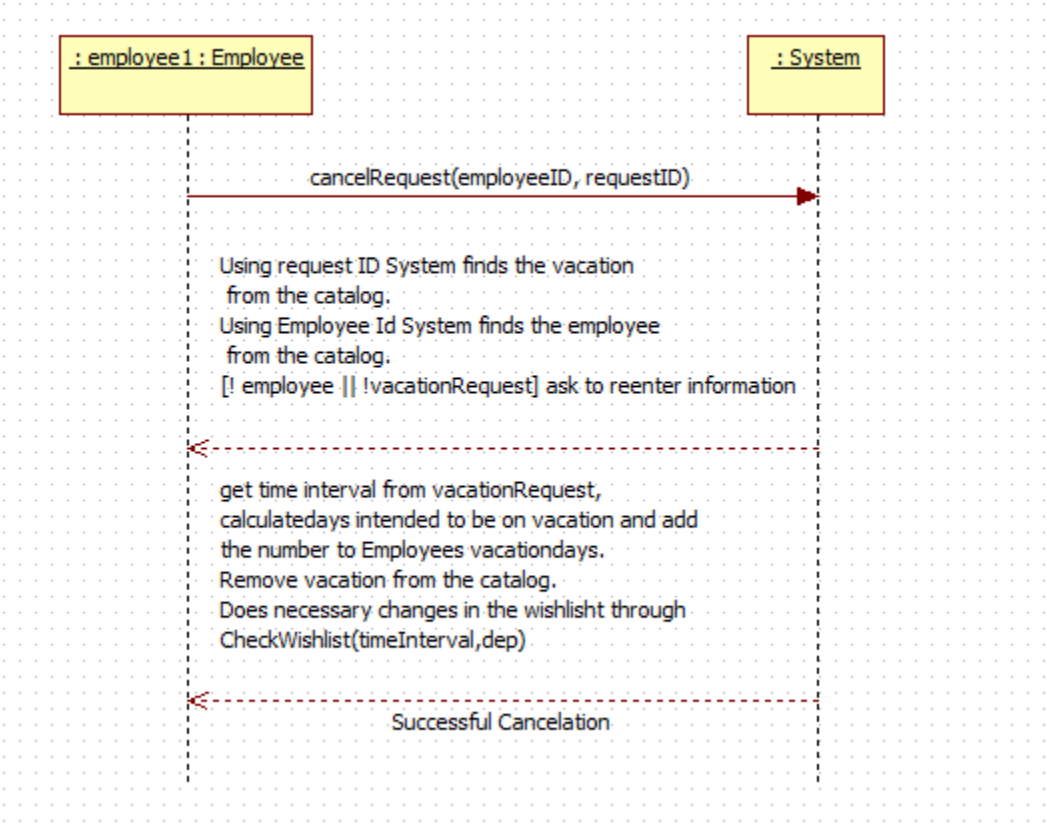
1. System asks to reenter the VacationID.

3a. No EmployeeId matched the employeeID

1. System Asks to reenter the employeeID

5. Special Requirments

Access to the database.



Use Case 3 : Vacation Initiating

1. Stakeholders and Interests

Primary Actor: DateController

Stakeholders and interests:

-- Date Controller. Notifies the system about a new date.

-- Company. Wants to make sure for all such vacationRequests starting on that day a vacation was created and was put on CatalogOfVacation. Also all such vacationRequests in the wishlisht were deleted.

-- Employee. All employees having vacationRequests starting on the currentDate Want their status changed to *unavaible* so they don't receive any work to do.

2. Preconditions and Postconditions

Preconditions: The current day coincides with the *timeInterval.first* of at least one vacationRequest.

Postconditions: For All such vacationRequests vacations were created and added to CatalogOfVacation. All such vacationRequests in the wishlisht were deleted. All employees status corresponding to requests was changed to *unavaible.*

3. Main Scenario.

1. System receives a notification about a new date.

2. System looks at CatalogOfVacationRequests and finds a vacation request s.t. its *timeInterval.first* corresponds with the currentdate.

3 System gets the linked employee and updates their status.

4. System creates a new vacation with information from the vacation request and adds to the CatalogOfVacations.

Steps 2-4 should be done for all vacationRequests with timeInterval.first = currentDate.

5. System goes over the WishlishtOfRequests and finds vacation matching the vacation *timeInterval.first = startDate.*

6. System deletes the vacationRequest.

Steps 5-6 should be done for all vacationRequests in the WishlishtOfRequest satisfying the condition.

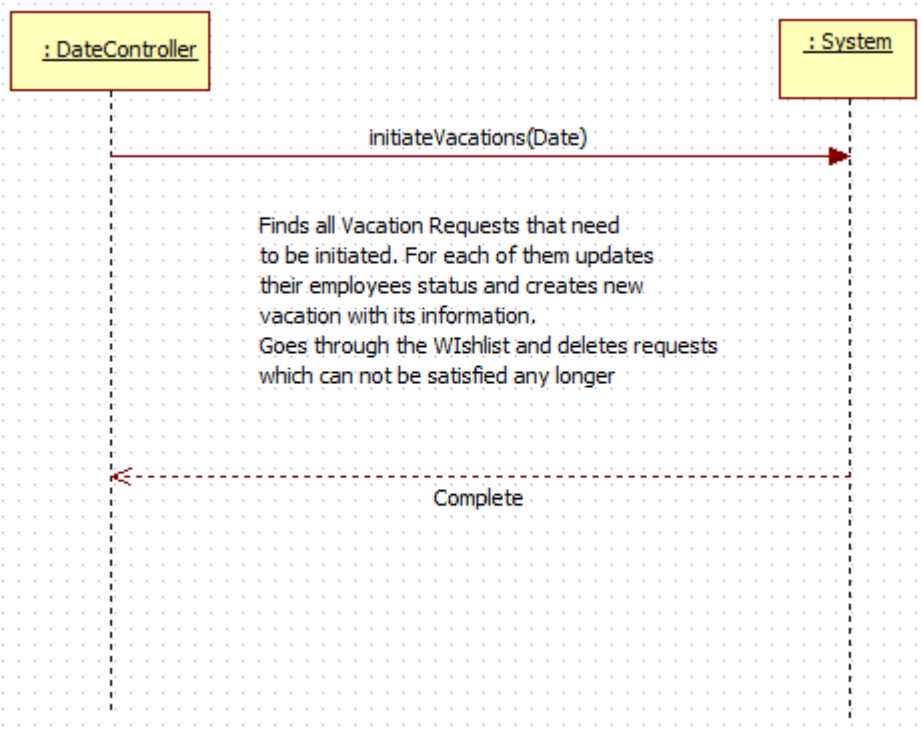
7. System notifies that all necessary changes were done.

4. Extensions (or Alternative Flaws).

N/A

5. Special Requirments

Access to the database.



Use Case 4 : Vacation Completeion

1. Stakeholders and Interests

Primary Actor: DateController

Stakeholders and interests:

-- Date Controller. Notifies the system about a new date.

-- Company. Wants to make sure for all such vacatios a record was made, following their removal. Also all employees had their status changed back to avaible.

-- Employee. All employees corresponding having vacations ending on the current Date want their status changed to *avaible* so they receive money for working.

2. Preconditions and Postconditions

Preconditions: The current day coincides with the *endDay* of at least one vacation.

Postconditions: All such vacations were removed from the catalog, and were kept in archive

3. Main Scenario.

1. System receives a notification about a new date.

2. System looks at CatalogOfVacations and finds a vacations.t. its *endDay* corresponds with the *currentDate.*

3 System gets the linked employee and updates their status.

4. System creates a new record storing the employee Id, the starting day and the return date.

5. System removes the vacation.

Steps 2-5 should be done for all vacations with endDate = currentDate

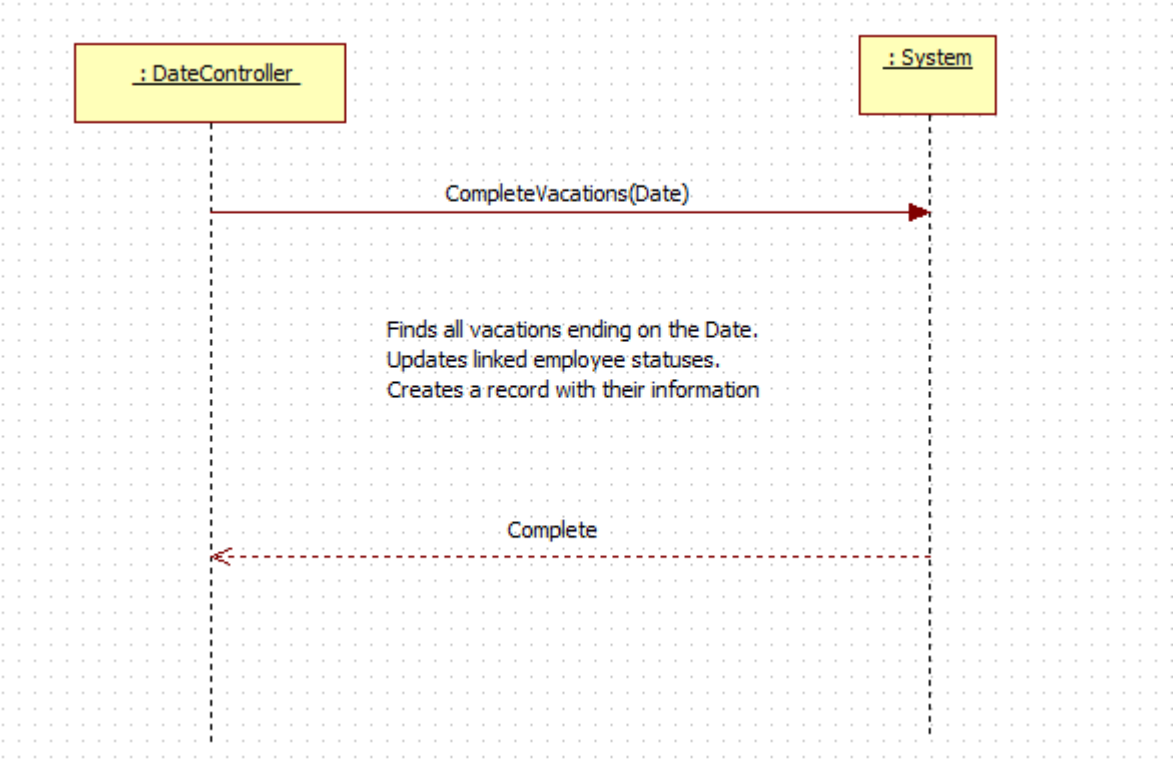
6. System notifies that all necessary changes were done.

4. Extensions (or Alternative Flaws).

N/A

5. Special Requirments

Access to the database.



Use Case 5 : Updating a vacation Request

1. Stakeholders and Interests

Primary Actor: Employee

Stakeholders and interests:

-- Employee. Wants to change the dates of a requested Vacation, or an ongoing vacation.

-- Company. Wants to make sure the change is possible. If it is it wants to make sure all the necessary updates are done.

2. Preconditions and Postconditions

Preconditions: An employee with a valid VacationRequestId

Postconditions: The Vacation Dates are updated. The wishlist is updated. The Employees information is updated

3. Main Scenario.

1. Employee enters a VacationRequestId and the new time interval.

2. System finds the VacationRequest and checks if the time interval can be changed to the new one.

Step 1-2 should be done until a new satisfactory time interval is found.

3 System applies all the necessary changes.

4. System sends a successful update notification.

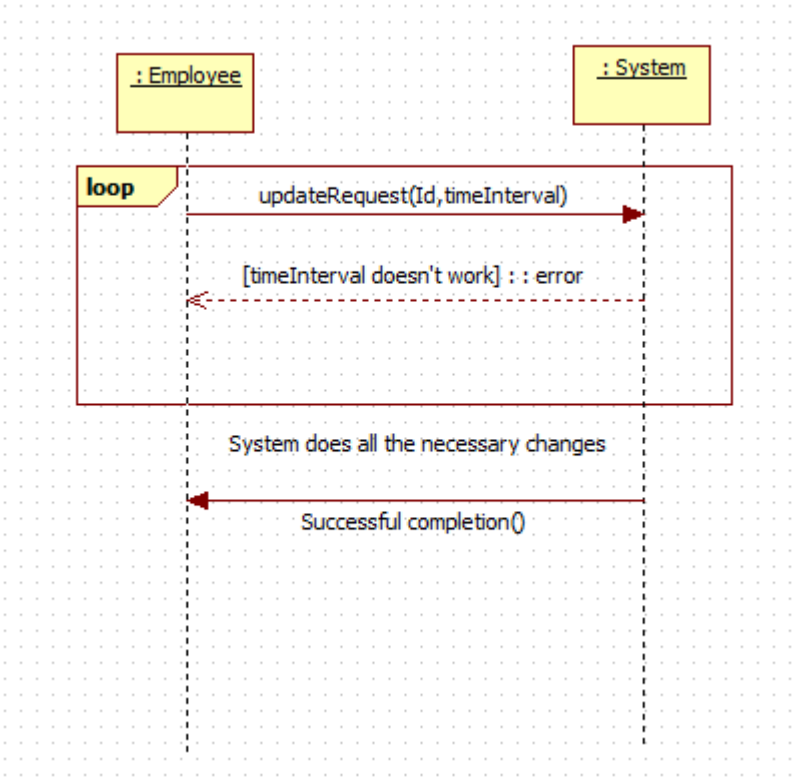
4. Extensions (or Alternative Flaws).

2a. System asks employee if he still wants to change. If Employee says no the operation is terminated.

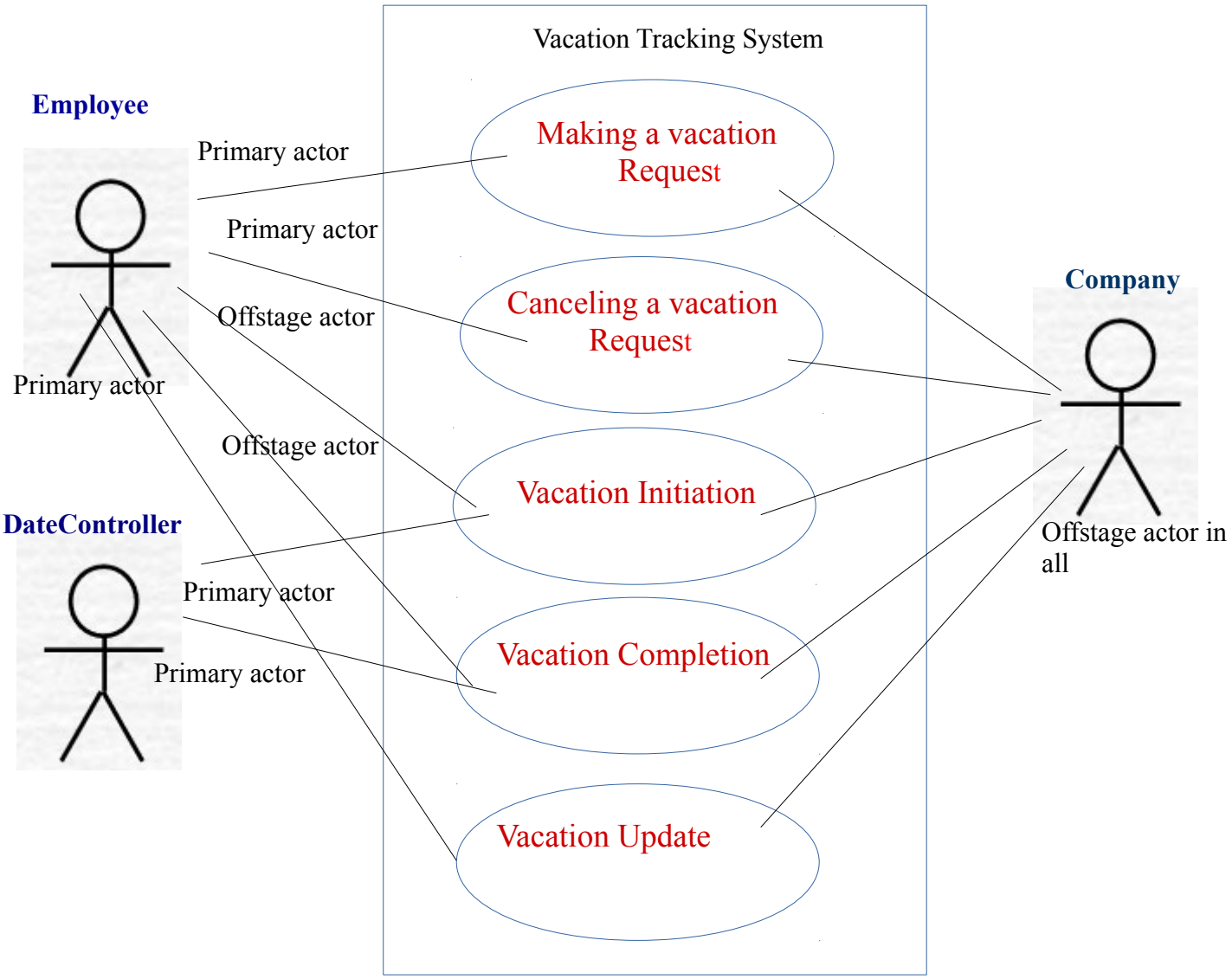
3a. If a new time interval has either later starting date or an earlier ending date system looks the wishlist for a vacations which may have became avaible.

5. Special Requirments

Access to the database.



Use Case Diagram



System Operation Contracts

Contract 01: check(employeeID, timeInterval)

Operation: checkVacation(employeeId, timeInterval) : VacationId;

References: "Making a vacation request" use case.

Preconditions:

-There is a c :Controller underway.

- There is an empList : ListOfEmployees linked to c.

-There is an employee :Employee linked to empList. such that employee.Id = *employeeId*.

- There is an dep : Department linked to employee.

-There is a vrc : CatalogOfVacationRequests linked to c.

- There is a str : CheckingStrategy linked to vrc.

Postconditions:

- if str.check(employee,timeInterval,vrc) returned an error code, an error message was returned.

(number or just false)

- before false was returned employeeId and timeInterval are kept for further use.

- Else, *true* was returned.

Contract 02: addToTheWishlist

Operation: addToTheWishlist(bToAdd);

References: "Making a vacation request" use case.

Preconditions:

-There is a c :Controller underway.

- There is an *employeeId* and *timeInterval* kept from check().

- There is an empList : ListOfEmployees linked to c.

-There is an employee :Employee linked to empCat such that employee.Id = *employeeId*.

-There is a w : WishlistOfRequests linked to c.

Postconditions:

if *bToAdd=false;*

-*employeeId* and *timeInterval* kept in c were resetted.

else

- *wishId* was generated by w

-vacRequest : VacationRequest was created and was linked to employee and w.

- vacRequest.Id = *wishId.*

- *wishId* was returned.

Contract 03: requestVacation

Operation: requestVacation(employeeId, timeInterval) : VacationId;

References: "Making a vacation request" use case.

Preconditions:

-There is a c : Controller underway.

- There is an empList : ListOfEmployees linked to c.

-There is an employee :Employee linked to empCat such that employee.Id = *employeeId*.

-There is a vrc : CatalogOfVacationRequests linked to c.

Postconditions:

- vrc generated a vacationId.

-vacRequest : VacationRequest was created and was linked to employee and vrc. Also vacRequest.Id = *vacationId.*

-employee.vacationDays were reduced by timeInterval length.

- *vacationId* was returned.

Contract 04: cancelRequest

Operation: cancelRequest(requestId);

References: "Canceling a vacation request" use case.

Preconditions:

-There is a c :Controller underway.

-There is a vrc : CatalogOfVacationRequests linked to c.

-There is a vacRequest:VacationRequest linked to vrc s.t. vacRequest.Id = requestId.

-There is an employee :Employee linked to vacRequest.

-There is a w : WishlistOfRequests linked to c.

Postconditions:

-employee.vacationDays were added by timeInterval.length.

- timeInterval was kept in c for updating wishlist.

- all links with vacRequest were removed after which vacRequest was deleted.

- w was updated via updateWishlist(timeInterval).

Contract 05: updateWishlist

Operation: updateWishlist(timeInterval, CatalogOfVacationRequests: vrc);

References: "Canceling a vacation request" use case.

Preconditions:

-There is a c :Controller underway.

-There is a w : WishlistOfRequests linked to c.

-There is a vrc : CatalogOfVacationRequests linked to c.

- There is a str : CheckingStrategy linked to w.

Postconditions:

For all wishRequest:VacationRequest linked to w such that *timeInterval* intersects wishRequest.timeInterval

if str.check(wish.Request.emp : Employee, timeInterval,vrc) = true

- emp was notified about possibility to add the wishlisted vacation.

Contract 06: initiateVacations

Operation: initiateVacations(Date);

References: "vacation request Initiation" use case.

Preconditions:

-There is a c :Controller underway.

- There is a w : WishlistOfRequests linked to c.

- There is a vrc : CatalogOfVacationRequests linked to c.

- There is a vCat : CatalogOfVacations linked to c.

Either

-There are vacRequests[0...n] : vacationRequest linked to vrc s.t. for all

vacRequests[0...n].timeInterval.first = *Date*;

- There are employees[0...n] : Employee linked correspondingly to vacRequest[0...n].

Or

-There are wishRequests[0...m] : vacationRequest linked to vrc s.t. for all

vacRequests[0...n].timeInterval.first = *Date*;

Postconditions:

- For all employees[0...n] status was changed to unavaible.

- vacations[0...n] were created. s.t. vacations[0...n].startDate = vacRequests[0...n].timeInterval.first,

vacations[0...n].endDate = vacRequests[0...n].timeInterval.second and vacations[0...n].Id =

vacRequests[0...n].Id. vacations were linked to the corresponding employees in vacRequests and vCat.

-wishRequests[0...m] were cleared from links and removed from vrc.

Contract 07: completeVacations

Operation: completeVacations(Date);

References: "vacation request Completion" use case.

Preconditions:

-There is a c :Controller underway.

- There is a vCat : CatalogOfVacations linked to c.

- There is a vRequestCatalog : CatalogOfVacationRequestss linked to c.

- There are vacations[0...n] : Vacation linked to vCat s.t. for all vacations[0...n].endDate = *Date*;

- There are vacationsR[0...n] : VacationRequests linked to vRequestCatalog s.t. for all vacationsR[0...n].timeInterval.second = *Date*;

- There are employees[0...n] : Employee linked correspondingly to vacations[0...n] and vacationsR[0...n].

- There is archive : Archive linked to c.

Postconditions:

- For all employees[0...n] status was changed to available.

- For each vacations[0...n] a corresponding record[0...n] : Record was created. s.t. record[0..n].startDate = *vacations[0...n].startDate,* record[0..n].endDate = *vacations[0...n].endDate,* record[0...n].employeeID = *employees[0...n].Id*. All records were linked to archive.

-vacatons[0...n] were cleared from links and removed from vCat.

all vacationsR were cleared from links and removed from vRequestCatalog.

Contract 08: vacationUpdate

Operation: updatelRequest(requestId,timeInterval);

References: "Updating a vacation Request" use case.

Preconditions:

-There is a c :Controller underway.

- There is an empList : ListOfEmployees linked to c.

- There is a vrc : CatalogOfVacationRequests linked to c.

- There is a There is a str : CheckingStrategy linked to vrc.

- There is a vacRequest:VacationRequest linked to vrc s.t. vacRequest.Id = requestId.

- There is an employee :Employee linked to vacRequest such that employee.Id = *employeeId*.

- There is a dep : Department linked to vacRequest.

Postconditions:

If

str.check(employee, timeInterval, vrc) is true and timeInterval.First< CurrentDate.

If timeInterval.second< vrc.timeInterval.second then send wishlist update(timeInterval.second, vrc.timeInterval.second)

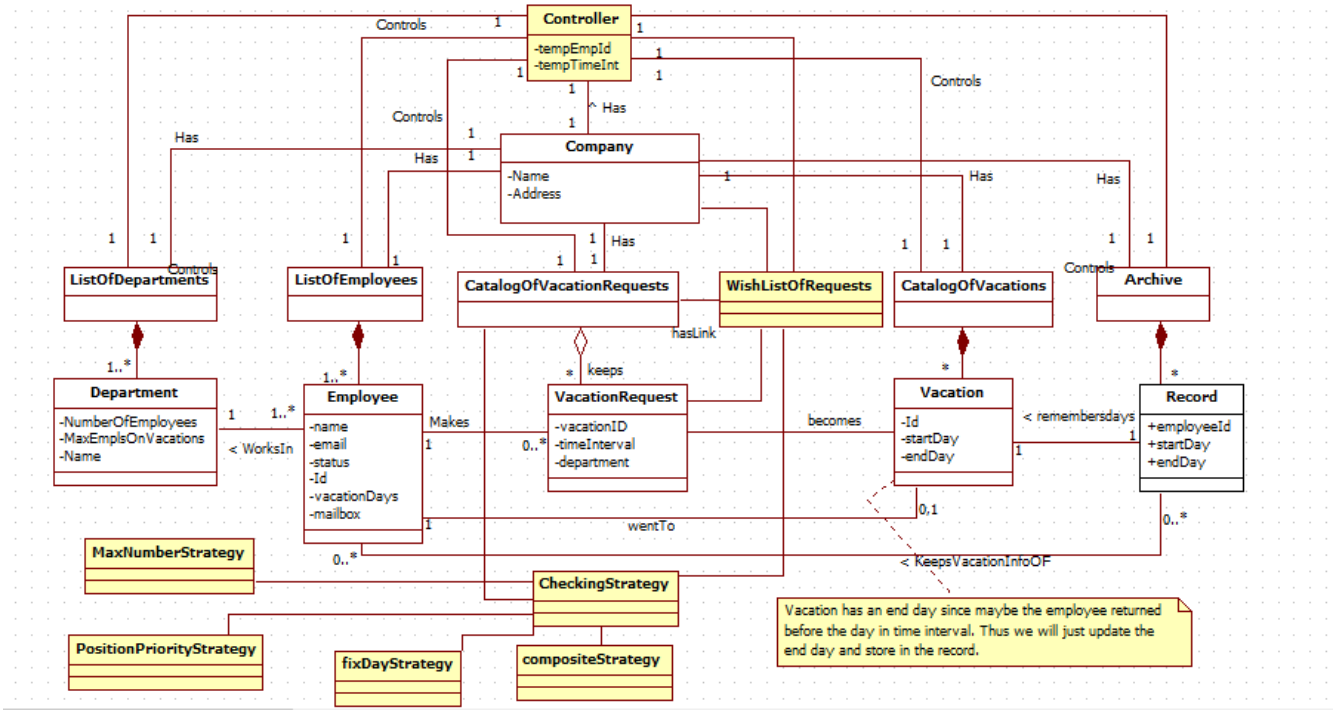
do same for the start day.

Also all necessary changes such as adding back vacation days were done.

Else

return false.

Domain Model



Analysis Model