UML Domain Model changes

UML Domain Model changes

Relationships

We have decided to change all the navigability to remove problems in future deliverables, because we've considered they are difficult to implement.

(Done third delivery)

- ### Relationship: FixUp Task-Warranty:

  Before: Bidirectional.

  We believe this modification improves the efficiency of the system, although the resulting JPQL queries are more difficult. Previously, when consulting a fix-up task, we bring all the associated warranties and vice versa, so we consult a fix-up-task all the time, and we don't need them at all, that's why we've decided to implement the following one:

  After: FixUp Task ← Warranty

(Done third delivery)

- ### Relationship: FixUp Task-Category:

  Before: Bidirectional.

  We believe this modification improves the efficiency of the system, although resulting JPQL queries are more difficult. Previously, when consulting a fix-up task, we bring all the associated warranties and vice versa, so we consult a fix-up-task all the time, and we don't need them at all, that's why we've decided to implement the following one:

  After: FixUp Task ← Category

(Done third delivery)

- ### Relationship: Referee-Report:

  Before: Bidirectional.

  In this case, we have decided this implementation because, if we request a referee, we don`t always need to bring their complaints, so we considered to implement the navigability in this way, so that only every time we consult a complaint, we bring its associated referee.

  After: Referee ← Report

(Done third delivery)

- Relationship: Box-Actor:

  Before: Bidirectional.

  We thought that bringing the actors related to a box we want to obtain is the most efficient way to implement this relation to the system. With a bidirectional relationship between these two classes (as it was made at first), the system obtains either a collection of boxes and a collection of actors.

  After: Box ← Actor

(Done third delivery)

- Relationship: HandyWorker-Curriculum:

  Before: Bidirectional.

  With a bidirectional relationship, when consulting a handy worker, we also bring all the associated Curriculum, and vice versa. This results in an inefficient implementation, so we've decided to change the relationship, although resulting JPQL queries are more difficult.

  After: HandyWorker ← Curriculum

(Done third delivery)

- Added a new relationship between Customer and FixUpTask.

(Done third delivery)

- Multiplicity between FixUpTask and Application has been changed from 0. * (FixUpTask) and 1 (Application) to 1 (FixUpTask) and 0..* (Application). Their navigability has also been changed, making the resulting queries easier.

(Done third delivery)

- Added a new relationship between Complaint and FixUpTask because we found problems trying to get the complaints of a specific FixUpTask, with a multiplicity of 1, and 0..* on the Complaint side.

(Done third delivery)

- Relationship: FixUpTask ---Warraty

  Before: FixUpTask ← Warraty

  The way we implemented this relationship is very inefficient. So, we have decided to change its direction, although resulting queries are more difficult.

  After: FixUpTask←Warraty

(Done third delivery)

- Relationship: Finder--- FixUpTask

  Before: Finder ← FixUpTask

  We decided to change the navigability, to make the queries, repositories and services easier.

  After: Finder❼FixUpTask

(Done fourth delivery)

- Relationship: SponsorShip ← CreditCard ← Customer

  We have added this relation because a Customer may have a credit card and a SponsorShip, since each of this Actor could have a credit card, as it is specified in the requirements.

(Done sixth delivery)

- Relationship: SponsorShip ← Tutorial

  We have eliminated the relationship of Sponsorship and Tutorial as we think it is unnecessary because it does not give us a solution to any of the specified requirements.

(Done sixth delivery)

- Relationship: SponsorShip ← CreditCard ← Customer

  We have removed the customer and sponsor relationships of CreditCard and have linked it directly with the Actor entity in order to simplify the domain model as much as possible.

Attributes

(Done third delivery)

- We also had to add an attribute to the finder entity, lastUpdate (Date type), in which we save the date of the last update. This is necessary because we need to know when the browser is not updated in order to update it.

(Done fourth delivery)

- We have added new attributes to the configuration entity. These attributes are positiveWord, spamWord, countryCode, banner,VAT, nameSystem and welcomeMessage.

(Done fourth delivery)

- We have read requirement number 38 and we have decided to implement two attributes, isBanned and isSuspicious, in the Actor entity in order to be able to distinguish from one of those actors who are suspected of being banned. Those who have already been banned and also know which are the actors who have no suspicion and have not been banned.

(Done fourth delivery)

- We have decided to change the type of the derived attribute *score* since we think it should be nullable:

double ----- Double

(Done sixth delivery)

- We have decided to remove the *isBanned* attribute because the same function that performed that attribute is also performed by the AccountNonLocked attribute, which gives us the userAccount class by default.

(Done third delivery)

- We have renamed the attribute *updatedMoment* to *moment*.

(Done sixth delivery)

- We have added the attribute, unique to the name of the warranty and CreditCard entity, because we will give the user the possibility to choose the warranty and the credit card by its names, so we need all their names to be different to distinguish them.

Entities

(Done third delivery)

- We also added a new entity called configuration, in which we store the results of the finder, the finder cache time, and a phone number pattern.

(Done third delivery)

- In the section entity we have changed the name of the id attribute to “number”, because we had problems with the implementation.

(Done fourth delivery)

- We have decided to create a new entity called CreditCard, in which we store the data of the related credit card of a Customer and a Sponsorship. We didn’t previously add this entity since we believed that we could not store the data of the credit card of our clients.

(Done fourth delivery)

- We have removed the attribute creditCard from SponsorShip due to the creation of the new entity named CreditCard .

(Done sixth delivery)

- We have added the attribute holderName to the entity CreditCard, because it were necessary, in order to register a fix-up-task.

Navigability and multiplicity character of atributes

(Done third delivery)

- We decided to change the applicable laws restriction because we considered that the previous one was not correct, since we decided that the applicable laws could be optional.

(Done third delivery)

- We have also decided to change the multiplicity of several entities, as we felt they were wrong after a second reading of the information requirements:

    - Complaint---Report (multiplicity changed).
    - Referee --- Report (multiplicity changed.)
    - Message--- Box (multiplicity changed.)

(Done third delivery)

- We also changed the multiplicity from Category to FixUpTask, swapped from 1 to 0..*, since we consider that a category should not be associated mandatory to a fix-up task, that is, it can create a category that does not have any associated fix-up task.

(Done fourth delivery)

- We have changed the multiplicity of the FixUpTask relationship with Warranty because we believe that several FixUp tasks can have the same associated Warranty. We first believed that only one type of warranty should be assigned to a single FixUpTask.

FixUpTask(multiplicity changed)---Warranty

(Done fourth delivery)

- We have changed the multiplicity of the Finder relationship with FixUpTask, since we believe that a fix-up task can be indexed in several finders, because some finders may have stored the same fix-up-task several times. Before we thought that only one finder can only return several fix-up-tasks.

Finder(multiplicity changed)---FixUpTask

(Done fourth delivery)

- We have changed the multiplicity of the relationship from Customer and Sponsor to CreditCard. Now is optional, since both do not have to have a registered CreditCard in the information system.

(Done fourth delivery)

- We have decided to delete the relationship between Phase and HandyWorker, because we have realized that this relationship is wrong.

(Done fourth delivery)

- We have changed the multiplicity of the relationship from Complaint to Referee. Before, a complaint had a mandatory referee; now it's optional (0..1).

(Done sixth delivery)

- We have changed the type of relationship between curriculum and it subtypes, before it was a compasition and now it became a normal relatinship, since we have realice that it was a mistake.

(Done sixth delivery)

- We have changed the subtype of personalRecord to non-madatory since we think that needs to be optional, also as the pther curriculums.

Roles

(Done third delivery)

- We have changed the roles of Application and Endorsement from the conceptual model, since it produced an error in our model because two of them were redundant.

Java Domain Model changes

In the second delivery we didn´t know we had to include relationship attributes in Java classes (excepts composition attributes), so we have implemented them in this delivery.

(Done third delivery)

- Attribute personalRecord in Curriculum class was Collection<PersonalRecord>, but we noticed that multiplicity from Curriculum to PersonalRecord is 1, so we changed the type of that attribute to PersonalRecord.

(Done third delivery)

- *@DateTimeFormat* annotation from getters of Date type attributes has been deleted because they don´t matter in this deliverable and in the previous one.

(Done third delivery)

- We have decided to remove the *@NotNull* attribute from the Curriculum class, because we considered that there were more positions, since a handy worker does not have to have a miscellaneous record mandatory. So, the only one that would be necessary should be personalRecord.

(Done third delivery)

- We have added the *@Transitional* annotation to the flagSpam attribute, since we consider that its calculation can be made relatively fast.

(Done third delivery)

- In Spring, the *@NotBlank* annotation includes the not-null restriction. In optional attributes, we don´t want this (it doesn't make sense). So, in the domain model, we have kept this annotation, but, in java, we didn't include it.

(Done fourth delivery)

- We have added new attributes to the java model, because we realized that we needed several attributes in the relationship modelling of the Java domain.
  - **In HandyWorker**: finder attribute.
  - **In Phase**: handyWorker attribute.

(Done fith delivery)

- We have changed the comments attributes of the application entity. We have replaced this attribute with handyWorkerComments, and the rejectedReason attribute with customerComments.

(Done fith delivery)

- We've also changed the type and name of the attachmentNumber attribute. It used to be an integer attribute, and now we've renamed it as an attachment, and changed its type to String.

(Done fith delivery)

- We have corrected spelling mistakes in the complaint class, in the description attribute, modifying, in turn, the PopulateDatabase.xml file.

(Done sixth delivery)

- We have removed the isBanned attribute from the Actor entity, since we have performed the same function in the Useraccount class through the userAccountNonLocked method.

(Done sixth delivery)

- We have changed the restrictions, of the pattern of both e-mail and phoneNumber, before did not conform to the requirements that we were required.

(Done sixth delivery)

- We have changed the finder´s restriction related with the minumun number of fix-up-task displayed as a result from the Finder.

(Done sixth delivery)

- We have added the restriction max(100) to the property VAT, as it is a percentage and this can not be greater than 100.

  (Done sixth delivery)

- We have added the restriction @CreditCardNumber y @NotBlank, to the CreditCard entity.

  (Done sixth delivery)

- To the endorser entity we have decided that the score must have more than two decimals so we have removed the restriction digits from the domain.

  (Done sixth delivery)

- New Populate because the older one was a bit inconsistent

Repository and services

(Done fith delivery)

- We have added the following repositories and their respective services: Credit Card, Tutorial, Sponsor, SponsorShip, Section, Report, Curriculum, Personal Record, Endorser Record, Education Record, Professional Record, Miscellaneous Record.

(Done fith delivery)

- We have edited the methods of saving actors, since we realized that they were wrong, when saving a customer was also saved as endorser and actor.

(Done sixth delivery)

- We have edited all the older services for the sixth derilevable. Also we have added new queries in order to fullfill all the functions that the web page requires.

Views

(Done sixth delivery)

- We have added the rest of views that were necessary to complete the web page, together with this, we have modified the tiles and the messages classes of the views to be able to make a correct project deployment.