DP II - 2022-2023

ANALYSIS - INDIVIDUAL - REPORT

Repositorio: ***https://github.com/pedlopruz/C1.04.12***

Tutor: RAFAEL CORCHUELO GIL

Alumno: CAROLINA CARRASCO DÍAZ

GRUPO - C1.04.12

1 Índice

2 Resumen Ejecutivo

La intención de este documento es la de recopilar y documentar el análisis de todos los problemas encontrados a la hora de implementar de forma individual los diferentes requisitos.

En esta segunda entrega, he encontrado una ambigüedad a la hora de expresar el periodo .El problema recae en cómo interpretar ese periodo de tiempo, ya que según la forma de interpretarlo afectará a todas las clases que tengan atributos de navegación con la entidad enrolment de dicho estudiante.

A continuación se detalla dicho problema, así como la propuesta de cambio proporcionada al cliente, a la espera de su aprobación u correcciones necesarias, siempre y en todo momento de manera consensuada con el cliente.

3 Tabla de Versionado

Versión	Fecha	Descripción
1.0	23/02/2023	Primera versión del documento
2.0	17/03/2023	Documentados los cambios llevamos a cabo mediante la aprobación del cliente a dicha propuesta y mejora de los requisitos

4 Introducción

La intención de este documento es la de recopilar y documentar el análisis de todos los problemas encontrados, (los cuáles han ido surgiendo a lo largo del desarrollo del proyecto) así como las propuestas de cambios proporcionadas al cliente en espera de su aprobación de cambio u correcciones necesarias, siempre y en todo momento de manera consensuada con el cliente.

5 Análisis de los requisitos

Contexto: *Every enrolment has a workbook that is composed of activities. The system must store the following data about them: a title (not blank, shorter than 76 characters), an abstract (not blank, shorter than 101 characters), an indication on whether it can be considered a theory activity or a hands-on activity, a time period (either in the past or the future), and an optional link with further information.*

Problema: El problema recae en cómo interpretar ese periodo de tiempo, ya que según la forma de interpretarlo afectará a todas las clases que tengan atributos de navegación con esta entidad. Por ejemplo, la entidad Enrolment debe tener el un atributo derivado que indique el tiempo de trabajo total , en horas, de las actividades que pertenecen a esa inscripción. Y aunque el requisito explica que ese periodo de tiempo puede ser tanto pasado como futuro, esto no afecta al problema porque decir que un periodo de tiempo sea válido tanto en el pasado como en el futuro he interpretado de que es equivalente a decir que cualquier periodo de tiempo es válido.

Alternativas propuestas:

Alternativa 1: Considerar periodo de tiempo como la diferencia de fechas de inicio y fin , ambas fechas teniendo como anotaciones @Temporal(TemporalType.TIMESTAMP), considerando el periodo como un Double, haciendo que las horas serán la parte entera del resultado y la parte fraccionaria como un porcentaje de los minutos:

Ej: 1.5 serían 1h y 30 minutos

- Ventajas:
  - Facilita el cálculo del atributo derivado de Enrolment, ya que se tendrían que sumar todos los periodos de tiempo ya calculados en la entidad Activity.
  - Al utilizar la parte fraccionaria como un porcentaje de los minutos, no hay que aplicar ninguna regla sobre la parte fraccionaria.
  - Cómo utilizas TemporalType.TIMESTAMP, se consideran los días en el cálculo del periodo.

- Desventajas:

- El periodo de tiempo sería un atributo derivado que devuelve la diferencia entre las fechas de inicio y de fin.

- -Se necesitaría una restricción customizada sobre las fechas de inicio y fin para evitar que la fecha de inicio sea superior a la fecha de fin y que la fecha de fin sea inferior a la fecha de inicio.

- No se cumpliría con la cantidad de atributos solicitados por el cliente, ya que tendrías que añadir los atributos de fecha de inicio y fecha de fin

Alternativa 2: Considerar periodo de tiempo como la diferencia de fechas de inicio y fin , ambas fechas teniendo como anotaciones @Temporal(TemporalType.TIMESTAMP), considerando el periodo como un Double, haciendo que las horas serán la parte entera del resultado y la parte fraccionaria como minutos

Ej: 1.3 serían 1h y 30 minutos

- Ventajas:
  - Facilita el cálculo del atributo derivado de Enrolment, ya que se tendrían que sumar todos los periodos de tiempo ya calculados en la entidad Activity.

  - Cómo utilizas TemporalType.TIMESTAMP, se consideran los días en el cálculo del periodo.

- Desventajas:
  - El periodo de tiempo sería un atributo derivado que devuelve la diferencia entre las fechas de inicio y de fin.

  - Se necesitaría una restricción customizada sobre las fechas de inicio y fin para evitar que la fecha de inicio sea superior a la fecha de fin y que la fecha de fin sea inferior a la fecha de inicio.

  - Sería necesario aplicar una regla sobre este valor Double para incrementar la parte entera cuando los minutos sean superiores a 59.

  - No se cumpliría con la cantidad de atributos solicitados por el cliente, ya que tendrías que añadir los atributos de fecha de inicio y fecha de fin

Alternativa 3: Considerar como periodo de tiempo un único atributo tipo Date y con la anotación @Temporal(TemporalType.TIME) y luego cambiar el formato a Double para que sea interpretable por Enrolment:

Ej: 01:30:00 como 1h y 30 minutos

- Ventajas:
  - Cumpliríamos con la cantidad de atributos solicitados por el cliente, ya que el propio atributo marcaría una duración al ser del tipo TemporalType.TIME

  - No sería necesario crear una restricción customizada ya que sólo es un atributo

- Desventajas:
  - Sería necesario pasar este atributo a un formato numérico que pueda interpretar la entidad Enrolment

  - Para pasarlo a Double, habría que establecer que entiendo como parte fraccionaria(si es un porcentaje de los minutos o si son la cantidad de minutos)

  - Cómo sería un TemporalType.TIME, no se tendrían en cuenta los días, por lo que habría que aplicar una regla que guardase los días del periodo

Solución Propuesta: Tras analizar las alternativas, la solución por la que nos hemos decantado es la primera, ya que aunque para aplicar el atributo solicitado será necesario crear 2 atributos que indiquen la fecha de inicio y la de fin y también será necesario crear una restricción que evitase los conflictos entre estas fechas, facilitará el cálculo del tiempo total de la entidad Enrolment.

Respuesta Cliente:

"La opción 1 que ustedes plantean me parece buena para tratar el problema. La única cuestión es que esto no es un problema de análisis. El análisis de los requisitos está relacionado con identificar incorrecciones, incompletitudes, ambigüedades, y misclassifications en los requisitos y en todos los casos debemos informar al cliente dado que los requisitos forman parte de nuestro contrato con dicho cliente. El problema que aquí hemos tratado es cómo implementar un requisito, lo que supone una decisión de diseño que incumbe tan sólo al equipo de desarrollo. Lo único que afectaría a nuestro cliente es cómo se muestran en pantalla los períodos y sus duraciones, pero entiendo que ahí no hay ninguna duda: el framework ya muestra los objetos de tipo date con un formato estándar en español e inglés y las duraciones pueden Udes. mismos decidir teniendo en cuenta lo que su cliente les ha dicho: 2.5h = 2 horas, 30 min.

Conclusión: cliente aprueba la propuesta 1; propuesta validada.

Contexto:

D02-S02-05: An enrolment is a registration of a student in a course. The system must store the following data about them: a code (pattern "[A-Z]{1,3}[0-9][0-9]{3}", not blank, unique), a motivation (not blank, shorter than 76 characters), some goals (not blank, shorter than 101 characters), and a work time (in hours, computed from the corresponding activities).

D02-S02-06: Every enrolment has a workbook that is composed of activities. The system must store the following data about them: a title (not blank, shorter than 76 characters), an abstract (not blank, shorter than 101 characters), an indication on whether it can be considered a theory activity or a hands-on activity, a time period (either in the past or the future), and an optional link with further information.

Problema: The first problem is that it does not specify what the time period is for, which from the context of the requirement and the previous one I am assuming it is the time that it woud take to complete the task, this is because the 5th requirement calculates the "work time" of an enrolment based on its activities. The second one being that the format in which it should be stored nor any constrains are specified either.

The second problem and the rest of my analysis on this requirement is based on the avobe assumption. So, how should we store this time period

Alternativas propuestas:

Alternative 1: We use a "Double" type field, which will store in a decimal form the ammount of hours it takes to complete the activity.

- Ventajas:
  - We use a simple field with which we can easily make the calculations for requirement #5.

- Desventajas:
  - None that I could think of.

Solución Propuesta: Using a "Double" type field to store the time it takes to complete the activity, with any constrains provided by the client.

Respuesta Cliente:

"Generally speaking, a time period must be modelled using a couple of attributes that make it explicit the starting moment and the ending moment:

```
@NotNull
@Temporal(TemporalType.TIMESTAMP)
protected Date startPeriod;

@NotNull
@Temporal(TemporalType.TIMESTAMP)
protected Date endPeriod;
```

Please, recall that you must add a custom constraint to check that starting moment is before the ending moment and possibly other custom constraints to check the the starting moment is in the future, or that the duration is not smaller or greater than a given duration.

Note that it's relatively easy to compute the amount of time between any two moments in hours (*). Thus, you only have to compute the work time as total number of hours in the activities in the workbook that corresponds to a particular enrolment. The conclusion is that the worktime is a derived attribute and it seems simple to define.

The real problem is: can you easily implement it as an attribute in an entity class, e.g., the class that implements the enrolments or the workbooks? It all depends on whether you can easily navigate from an enrolment/workbook to the corresponding activities or not. Generally speaking, we have a recommendation: do not use relations other than single-way, many-to-one relations. If you take that piece of advice into account, then you cannot directly navigate from an enrolment/workbook to its corresponding activities. That means that you can't easily implement a derived attribute like "worktime" using a transient attribute in class "Enrolment". Transient attributes allow to implement derived attributes that can be computed from the other attributes in the same entity; if you need data that are not in the same entity, then you're dealing with a more complex derived attribute and you have to wait until next lesson."

Conclusión: cliente aclara la duda.

6 Conclusión

Queda documentado en este reporte de análisis cualquier cambio llevado a cabo durante la realización del entregable en curso respecto a cualquier entidad que sugieran los requisitos del Student#2. Quedando fuera y desestimado por tanto cualquier cambio no aprobado por el mismo.

7 Bibliografía

[1] Enlace de la propuesta de cambio frente al requisito mencionado y respuesta y aprobación del cliente:https://ev.us.es/webapps/discussionboard/do/message?action=list_messages&course_id=_63009_1&nav=discussion_board&conf_id=_303964_1&forum_id=_206215_1&message_id=_359274_1