8 DE MARZO DE 2024

ANALYSIS REPORT

GROUP C-1.047

Acme SF, Inc.

Repository link: https://github.com/Cargarmar18/Acme-SF.git

Rodríguez Millán, María C.
Email: marrodmil@alum.us.es

Table of contents

1. Executive summary

The contents of this analysis report present a listing of the requirements mandated by the client for the Acme-SF project in its D02 version. For each requirement, a breakdown is provided (if applicable), including a verbatim copy of the referenced requirement, detailed analysis conclusions and decisions made to address the requirement. A breakdown will be deemed necessary when the requirement requires any sort of analysis; otherwise, no commentary will be necessary.

2. Revision table

Revision number	Date	Description
1.0.0	08/03/2024	Document elaboration.

3. Introduction

The content of this document aims to conduct a comprehensive study of the requirements provided by the client for the successful completion of the current project, Acme-SF. By meticulously analysing and documenting them, it focuses on accurately identifying potential conflicts or ambiguities that may arise.

This objective is intended for the eight detailed requirements; four of them involving information requirement's functionality implementation, one involving testing and three involving the preparation of managerial documentation which include an analysis report, a planning and progress report and a UML domain model.

The document is structured starting with a cover page containing relevant information about the document, such as its title, the project's repository, the group number, the author or authors' names with their corresponding corporate email addresses, and the date. The document proceeds with a table of contents, an executive summary, and an introduction before delving into its content. The analysis report focuses on the individual requirements specific for each student, being in this case, the third one (student #3). As mentioned earlier, a breakdown of each requirement will be provided whereas applicable. Finally, the document ends with a conclusion summarizing the presented information and an appropriate bibliography. This structure, as described and used, is outlined in the "Annexes" document, provided by the University of Seville.

4. Contents

The listing of requirements and their corresponding breakdowns can be found as follows.

Requirement 002:

> *A **training module** consists of one or several short-term training activities aimed at extending or updating knowledge and skills related to the topic of a **project**. The system must store the following data about them: a **code** (pattern "[A-Z]{1,3}-[0-9]{3}", not blank, unique), a **creation moment** (in the past), some **details** describing the training module (not blank, shorter than 101 characters), a **difficulty level** ("Basic", "Intermediate", or "Advanced"), an optional **update moment** (in the past, after the creation moment), an **optional link** with further information, and an estimated **total time**.*

An exhaustive examination of the requirement has been undertaken through consultations with the client, elucidating the following ambiguities:

1. Constraints

The constraints found in this requirement that may need further analysis are described as follows:

- **Explicit constraint:** the "update moment" date must be after the "creation moment" date. This means that this constraint must be implemented in the service, adding the task to the workload of the project in its D03 version.
- **Implicit constraint:** lower limit for dates. The lower limit for dates is 2000/01/01 00:00, that is, all dates must correspond to a moment after the lower limit date. This lower limit must be implemented in the service as well.
- **Implicit constraint:** upper limit for URLs. The upper limit for URLs is 255 characters.
- **Implicit constraint:** all attributes that are not explicitly mentioned as optional are mandatory.

It is worth noting that the mentioned implicit constraints were agreed upon the client.

2. Estimated total time data type

This particular attribute, due to its lack of explicit specification regarding the associated requirement, introduces uncertainty into the implementation process. It could be interpreted as an Integer, a Duration, or alternatively as a Double. Additionally, the designated unit of time for storing said attribute is not specified (whether in hours, minutes, days, etc.).

The alternatives given to the client were the following:

- **Alternative 1:** Using Integer to store integer values, like whole days or whole hours, with the pro of being easy to implement and handle data, and the con that it may not be correctly modelled.
- **Alternative 2:** Using Duration, with the pro of adjusting better to store real times, and the cons of being pretty complex and the probability of arising errors.
- **Alternative 3:** Using Double, with the pro of being easy to implement and handle data, and the cons of ambiguities in the interpretation of the values (2.50 hours can be two hours and a half or two hours, fifty minutes) and not knowing what "total time" stands for.

The client, after being consulted, has concluded that the time will be measured in whole hours (choosing alternative 1), with the rounding of numbers offsetting those rounded upwards and those rounded downwards among them.

3. Estimated total time computation

It is crucial to keep in mind that requirements are stated individually for clarity and structure, but they cannot be fully understood in isolation but rather must be examined in the appropriate context of requirements elicitation. Thus, when referencing "training modules", we are speaking about an entity that is a conglomerate. Therefore, analysing the "whole" requires an examination of the "components", which in this case are "training sessions" (see *Requirement 003*).

Thus, the estimated total time of a training module may be a derived attribute which results in the summation of the "time period" (see *Requirement 003*) from the "training sessions".

The client, after being consulted, has concluded that it seems reasonable to think that the "total time" of a training module is computed as the summation of the effective labour time of its training sessions, removing possible non-effective hours such as breaks or lunches.

The computation of this attribute, then, would be done in the service of this entity, so it would be added to the task workload for the Acme-SF project in its D03 version.

Requirement 003:

> *Each **training module** is made up of **training sessions**. The system must store the following data about them: a **code** (pattern "TS-[A-Z]{1,3}-[0-9]{3}", not blank, unique), a time **period** (at least one week ahead the training module creation moment, at least one week long), a **location** (not blank, shorter than 76 characters), an **instructor** (not blank, shorter than 76 characters), a mandatory **contact email**, and an **optional link** with further information.*

An exhaustive examination of the requirement has been undertaken through consultations with the client, elucidating the following ambiguities:

1. Time period implementation

The "time period" implementation, due to its lack of explicit specification regarding the associated requirement, introduces uncertainty into the implementation process. It could be interpreted as a Date, a numerical type or a custom data type.

The alternatives given to the client were the following:

- **Alternative 1:** Using Date, with the pro of being comparable with other date attributes (needed for a constraint) and the con that it does not make much sense with the concept of "time period".
- **Alternative 2:** Using a numerical type, with the pro that it matches the concept of "time period" and the con that the constraint cannot be checked directly in the entity.

- **Alternative 3:** Using a custom data type, with the pros that the solution would be reusable and would match the concept of "time period", but the cons of the high complexity and the probability of arising errors.

The client, after being consulted, has concluded that the time periods or the duration attributes must be addressed as two new attributes of the Date type; one referring to the beginning moment and other referring to the end moment, with a correct set of constraints. In addition, the client elucidated that a time period or duration cannot be under an hour long.

2. Constraints

The constraints found in this requirement that may need further analysis are described as follows:

- **Explicit constraint:** the beginning moment previously mentioned by the client must be at least one week ahead the creation moment of the training module. This means that this constraint must be implemented in the service, adding the task to the workload of the project in its D03 version.
- **Explicit constraint:** the end moment previously mentioned by the client must be at least one week after the beginning moment of the training session. This constraint must be implemented in the service as well.
- **Implicit constraint:** upper limit for dates. The upper limit for dates is 2200/12/31 23:59, that is, all dates must correspond to a moment before the upper limit date. This constraint must be implemented in the service as well.
- **Implicit constraint:** lower limit for dates. The lower limit for dates is 2000/01/01 00:00, that is, all dates must correspond to a moment after the lower limit date. This lower limit must be implemented in the service as well.
- **Implicit constraint:** upper limit for URLs. The upper limit for URLs is 255 characters.
- **Implicit constraint:** all attributes that are not explicitly mentioned as optional are mandatory.

Again, the implicit constraints were agreed upon the client.

Requirement 004:

> *The system must handle* ***developer*** *dashboards with the following data: total number of* ***training modules*** *with an* ***update moment****; total number of* ***training sessions*** *with a* ***link****; average, deviation, minimum, and maximum* ***time*** *of the* ***training modules****.*

In this requirement, commenting on analysis has not been deemed necessary due to its simplicity.

Requirement 005:

> *Produce assorted sample data to test your application informally. The data must include two* ***developer*** *accounts with credentials "****developer1/developer1****" and "****developer2/developer2****".*

In this requirement, it is worth mentioning that the production of assorted sample data to test the application informally implicitly involves generating sample data for the entities (training module and training session). This sample data will imply generating data by exploring potential values that may lead to errors within the constraints of the entity.

Requirement 013:

> *There is a new project-specific role called **developer**, which has the following profile data: **degree** (not blank, shorter than 76 characters), a **specialisation** (not blank, shorter than 101 characters), list of **skills** (not blank, shorter than 101 characters), an **email**, and an **optional link** with further information.*

An exhaustive examination of the requirement has been undertaken through consultations with the client, elucidating the following ambiguities:

1. Skills attribute implementation

The "skills" attribute implementation, due to its lack of specification regarding the data type to be used, introduces uncertainty into the implementation process. It could be interpreted as a List<String>, or as a String.

The following alternatives were proposed to the client:

- **Alternative 1:** Using List<String>, with the pro that it would be easier to use and to manage operations, and the cons of being more complex to implement and the possible arising errors.
- **Alternative 2:** Using String, with the pro of being easier to implement, and the con of being difficult to perform some operations.

The client, after being consulted, has decided that the best option to implement this requirement in our framework is the second alternative, since the first one would produce too many complications and errors, in addition to be much more complex to implement and to handle.

2. Constraints

The constraints found in this requirement that may need further analysis are described as follows:

- **Implicit constraint:** all attributes that are not explicitly mentioned as optional are mandatory.
- **Implicit constraint:** upper limit for URLs. The upper limit for URLs is 255 characters.

Once again, the implicit constraints were agreed upon the client.

Requirement 014:

> *Produce a UML domain model.*

In this requirement, commenting on analysis has not been deemed necessary due to its simplicity.

Requirement 015:

> *Produce an analysis report.*

In this requirement, commenting on analysis has not been deemed necessary due to its simplicity.

Requirement 016:

> *Produce a planning and progress report.*

In this requirement, commenting on analysis has not been deemed necessary due to its simplicity.

5. Conclusions

Based on the present report and in the in-depth analysis conducted for its elaboration, it can be concluded that the potential ambiguities or conflicts have been properly identified and clarified. Thus, the examination has provided confidence in the clarity and coherence of the mentioned requirements, contributing to build a solid and successful implementation.

6. Bibliography

Intentionally blank.