UADY

UNIVERSIDAD
AUTÓNOMA
DE YUCATÁN

Software Engineering

-Software Engineering Fundamentals-

First Increment

Osiris Team Members:
Fernan Enrique Cetina Escalante
Jorge Teodoro Dawn Rodriguez
Rodrigo Alejandro Castrejón Cervantes
Cinthia January Huchin Pedrero
Ricardo Reyes Balam Cupul

Mérida, Yucatán. November 30th, 2020.

Application definition

Objective:

Our project intends to help students manage their time, keep track of their future activities and allow them to retrieve time missed. For this purpose, we will develop a tool that allows its users to add and remove objectives. It will also have several options to keep track of them with constant notifications, progress updates and a tool that helps them find a path to complete those activities they missed without overlapping with new tasks. A limitation inherent to the system is that the user won't be able to change the rate of notification because we want them to actively seek for completion.

Users:

Primary: Students of 1st semester of UADY studying Software Engineering. We expect them to be around 18-19 years old, from middle to lower economic class and unemployed. 20% Women - 80% Men

Secondary: Students of higher semesters of UADY studying Software Engineering. We expect them to be around 19 to 25 years old, from middle to lower economic class and employed. 20% Women - 80% Men

Potential: All students at University level that struggle with time management. We expect them to be around 18 to 25 years old, from middle to lower economic class and employed. 50% Women - 50% Men

There is a distinction between this profiles: The primary users are the ones in which we detect needs that should be solved and everything is intended to help them and they also provide the requirements; the secondary users are the ones who can take advantage of our project, that we know somethings about them but are not actively seeking to solve necessities as we don´t ask for requirements; and for the potential users we describe all the users that no matter their background, if they need our solutions, they can use it and also are a broader sample of people what our primary users show but on a bigger social scale, they also represents a bigger market if we intended to expand the project or develop requirements with more complexity.

Customers:

We don't have a specific client as no one is paying for the development and stakeholders are pending to be actively involved in the project like Scrum strongly suggest, so we decided to consider our customers as the same of our primary users because we have access to a current pool of students in first semesters to guide our vision, although for timing they won't be the ones using it, they can provide the depth needed to the project definition.

Innovation:

We will be the first product to propose a new way in which students can improve their organization by tracking their activities step by step and having reminders that will suggest new dates for continuing their activities and show their advance, so they not only finish on time but also have tools to retrieve time missed, the former might be provided by *reminder apps* but a combination with the latter, something that can help you “go back” in time, haven't been done. We noticed on a survey that a major problem is time management and as

existing organization solutions are not working, we take part by proposing a product to give these methods a boost and help students move forward on their academic commitments.

Requirements or user stories

Distinguish which will be used:

Requirements, both functional and non-functional, are essential for the development process of software. They concisely describe, in a way that our client can easily comprehend, what our project should do and what its constraints are. That is why we have preferred them over user stories in our project.

Functional requirements and not functional requirements:

- The system shall provide a short introductory tutorial that teaches users its main features: add, edit and remove tasks, update progress, and change notification frequency.
    - It should always appear on the first use and be accessible via a separate tab at any time.
- The user shall be able to see a list of all of their current active tasks. It is very important to display due date and progress status.
    - Each user should only be allowed to access their own task list.
    - By default, the tasks should be organized by their due date, nearest one first.
- The user shall be able to add, edit or remove a task from their task list.
    - These actions should appear right next to each task in the list view and should take one click to use them.
    - The user should be able to format the text on the task.
    - Our system should not save any identifiable information from users, these being: name, username, and email.
- The user shall be able to set a due date when creating a task and also be able to modify it [due date] via the ‘update task’ action.
    - The due date should be set using a date picker or parsed from text.
- The system should reassign a task when a user misses its due date.
    - The user should be alerted when this occurs.
- The system shall display the current progress of each task and allow the user to update said progress.
    - The system should allow for progress to be reverted in case of a revision or an accidental update.
    - The ‘progress update’ action should be accessible directly on the task list, right next to each task’s progress.
    - It should only take one click to update the progress.
- The user should be able to share their current task list.
    - Other users should not be able to modify in any way a task list that does not correspond to them.
- When the user has active tasks, the system shall keep pushing daily notifications saying “You have pending duties” until they complete them.
    - By default, the system should only send one notification per day.

- The user shall be able to set the notification frequency per day.
    - The user should be able to choose the notification frequency of each day of the week.
- Each task shall allow users to append ‘update notes’ to them in order to keep better progress.
    - The notes should be displayed alongside its corresponding task in the task list.
    - The notes should allow text formatting.
    - Update notes should not be obligatory for the user.
- Periodically, the system shall write a text encouraging the user to work on finishing their tasks.
    - By default, these notifications should be sent each Monday.
    - These texts must be less than seven words long.
- The system shall perform constant backups of all saved data.
    - These backups should be performed every Saturday in the morning.
    - Only developers should have access to them.
    - It must notify developers in case of failure.

Prioritization methodology:

For this project, we decided to implement the MoSCoW methodology. We believe it is perfect for the prioritization of requirements of a small scale team like ours. This technique is easy to understand, it assigns each requirement one of the following prioritization categories:

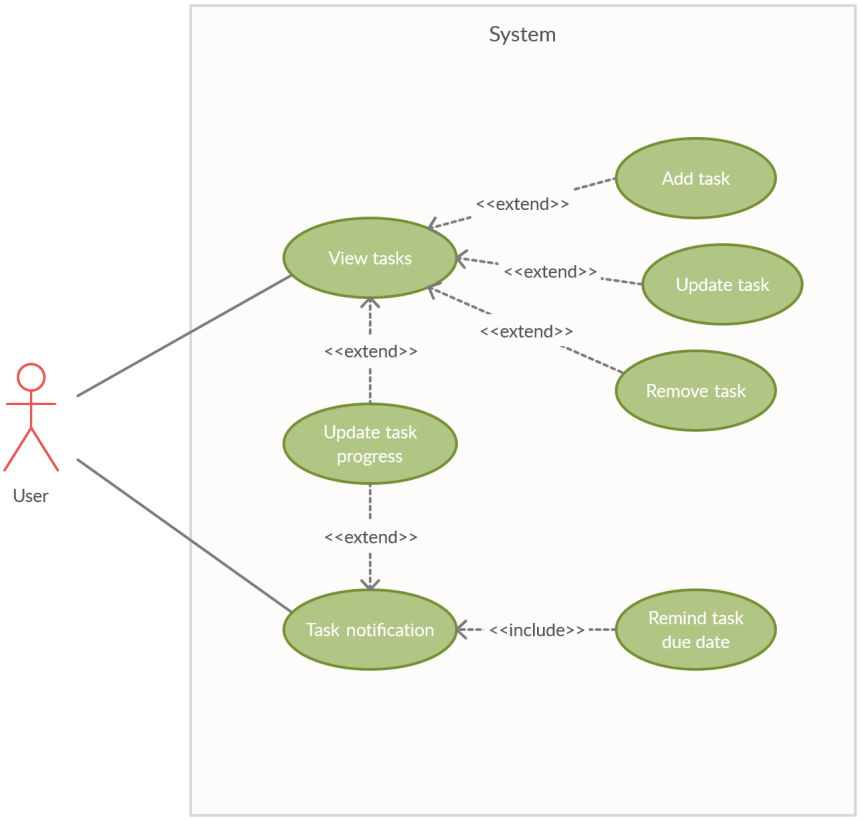
- **Must have**: Critical to the delivery of the current Sprint. If ‘*must have’* requirements are missing, the project delivery should be considered a failure.
- **Should have**: Important but not necessary for the increment delivery.
- **Could have**: Desirable requirements that could improve the user experience.
- **Won’t have**: Lowest-payback requirements that have been agreed not to be planned for the time being. These requirements are either dropped or changed into more fitting ones.

To assign a category to each requirement, we evaluated them as a team and wrote-down our observations. Then we assessed the difficulty of implementation, and finally, we agreed on a classification. Typically, you would want the client to have input in this decision, but as we mentioned before, our classmates can't have active participation in our project. The following table includes a detailed look at our project's requirements:

Requirement	Observations	Classification
Provide a short tutorial.	Should be easy to implement. It is essential for our users.	M
See a list of all current tasks.	Should be easy to implement. It is necessary for our project’s functionality.	M
Add, edit and remove tasks.	Should be easy to implement. It is necessary for	M

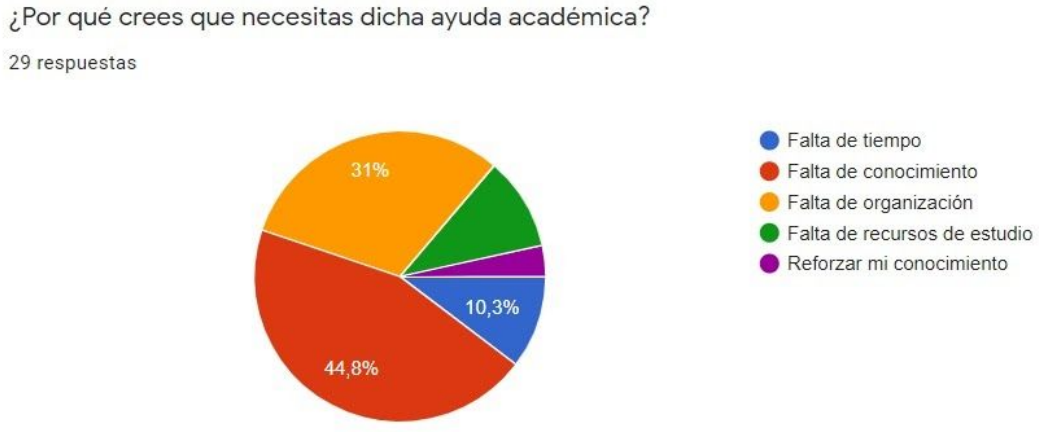
	our project's functionality.	
Be able to set and update due dates.	We expect it to be slightly complicated, but it is essential for our project.	M
Reassign task.	It will be hard to implement, but it is essential.	M
Display and update progress of each task.	Slightly complicated to implement, but it is necessary for our project's functionality.	M
Share task list.	It isn't a priority for us, but it would be a good addition in the future. Shouldn't be hard to implement.	C
Push notifications.	We believe this is the hardest requirement to implement. To not stall the process we have decided to classify it as "S" with future intentions to bump it to "M".	S
Be able to set notification frequency.	If we succeed in implementing notifications, this requirement should be easy. It is somewhat important.	S
'Update notes' on each task.	It should be slightly complicated, but we believe we can achieve it. Not very necessary.	C
Encouraging sentences.	Not really important, not really hard to implement.	C
Perform weekly backups.	The final product should implement it, but right now it is not necessary.	S

Diagram:



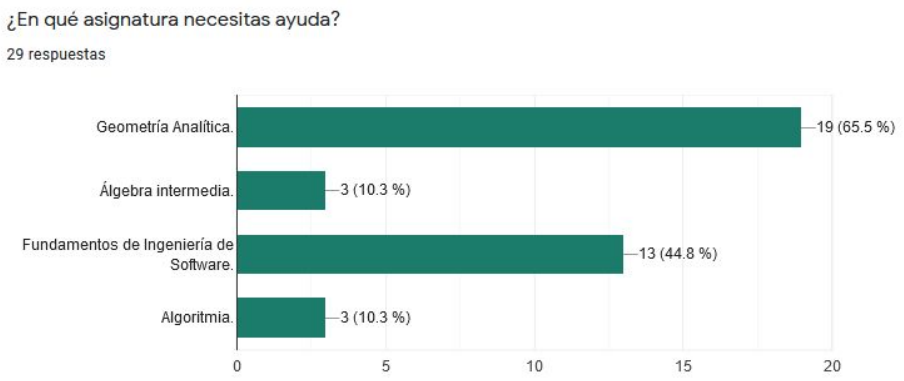
The team counts with artifacts that specify the requirements/user stories, it includes exceptions to take into account (use cases):

To specify our requirements, we conducted a survey directed at our classmates where we asked them to tell us, amongst various things, which areas of study they needed the most help with.



The picture above shows the results of said survey. As seen there, 31% of them said they needed help with organization, and 10.3% said they needed help with their lack of time. Those two areas combined almost match the one where people answered that they needed help with their lack of knowledge. But we decided to focus on the former because knowledge is gained as time passes, while time and task management depend on the student's habits.

Also, poor time management can negatively impact someone's ability to learn, like when they miss classes or forget to turn in homework.



Then, we saw that the curriculum assigns around 224 total hours to Analytic Geometry and Software Engineering Fundamentals, which were the subjects our classmates needed the most help with. At this point, we concluded that our project should focus on providing a way to track people's tasks and activities and allowing them to keep updating their progress.

Development process

Methodology:

Scrum is the methodology that is being implemented due to its flexibility and its ability to adapt to the project. So far, the Product Backlog has been established and only user requirements were described; so what it corresponds to the Sprint Backlog, there will be times in which it will be the same as the Product Backlog and others the Scrum master on a Sprint Planning stage will decide which requirements to work with. For the assignment of activities, we are using Trello, in which every week the activities that we must present to the Scrum master are updated (the equivalent to daily scrums), once the activity is finished, the testing is carried out and we proceed to identify the errors to be corrected, this process is repeated until that the activity complies with our quality control, then the commits are carried out in the GitHub repository.

For the following increments, the same process would be carried out, that is, the assignment of activities, which will be reflected in the Trello which will have a time and deadline established by the scrum master, once the activity is finished, the activity will be delivered, once approved, they will be presented to the team for a final review (sprint review), if necessary the respective corrections will be made and the changes will be uploaded to the GitHub repository. It is estimated that a meeting will be scheduled for the reviews on the teams platform, which all participants must attend. A day later of the release of an increment a Sprint Retro should be performed under an hour.

Process description:

Role of the members:

Scrum master:

- Fernan Enrique Cetina Escalante.

Developers:

- Jorge Teodoro Dawn Rodriguez.
- Rodrigo Alejandro Castrejón Cervantes.
- Cinthia January Huchin Pedrero.
- Ricardo Reyes Balam Cupul.

A metric was designed in which criteria were established to rate the contribution of each member.

The Scrum Master is the one who will carry out the assignment of the activities. The responsibilities are according to the role of each member, however, at a certain moment, one of the members may occupy two roles, either as a developer or as a tester. For the creation of the repository and the Trello, it was assigned to the Scrum Master, where he is the one in charge of reflecting the changes in the Trello, and the changes in the repository are assigned to a developer person or by the Scrum master himself.

Process management:

Monitoring tools:

- ❖ Trello.
  - ■ Tool that allows us to maintain an order of the activities, as well as the delivery time of each of them in a visual way.
- ❖ GitHub.
  - ■ It allows us to keep track of individual contributions and sprint products.
- ❖ Excel.
  - ■ It allows us to create the Contribution tracker and Description of the development process.
- ❖ Teams
  - ■ It allows us to have access to group meetings, to organize ourselves and contribute ideas about the process and development of the work.

Monitoring Activities and tasks:

- ➢ Activity:
  - Establish the means of organization and follow up.
- ➢ Task:
  - Create a work team in the trello and assign the sprints.
- ➢ Activity:
  - Creation of follow-up of contributions:
- ➢ Task:

- Assign the contribution values of each individual in the tracking of contribution, according to the progress of the member.

➢ Activity:
- Product definition.

➢ Task:
- Schedule meetings where we discuss the product we want to make.

➢ Activity:
- Project planning.

➢ Task:
- Establish a meeting to discuss the parameters of the project and the actions we will carry out throughout the development process.

➢ Activity:
- User requirements establishment.

➢ Tasks:
- Apply surveys to identify user needs.

➢ Activity:
- Deployment of advances.

➢ Tasks:
- Schedule a meeting to establish the progress made each week.

Evidence of meetings:

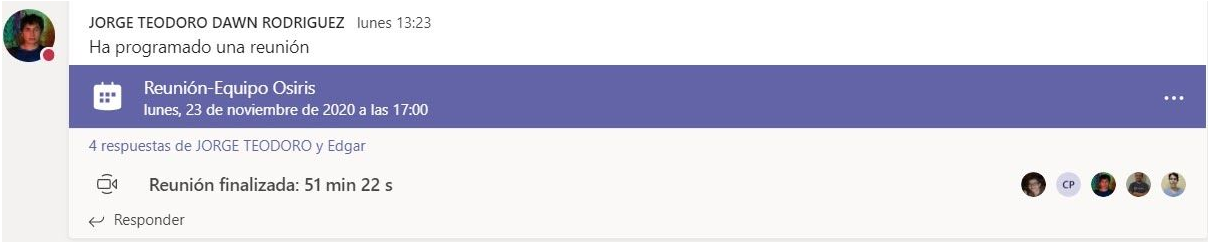
First scheduled meeting: October 22, 2020, at the first meeting the software proposals to be developed were presented, as well as the methods to be used and their implementation.

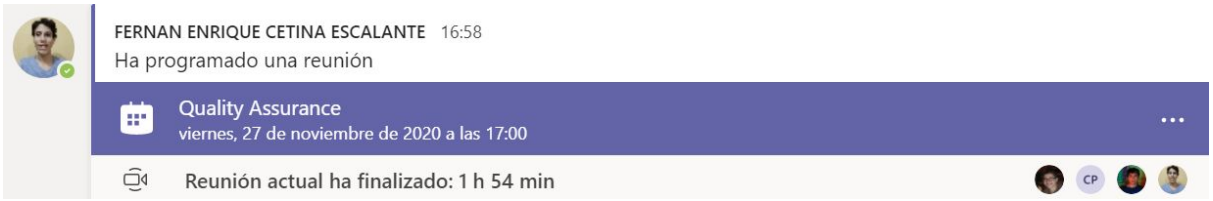
Second scheduled meeting: November 5, 2020, at this meeting the mentor was present. The participants presented their requirements and a re-evaluation of the user's needs was carried out, which influenced the orientation of the product.

Third meeting: November 13, 2020, once again the mentor was present. The data collected from the survey of first-semester students of the degree in software engineering were presented, a new approach to the product was also proposed.

Fourth meeting: November 22, 2020, the contribution and requirements were presented to the mentor, and in turn, the points to be corrected were discussed.

Fifth meeting: November 23, 2020, in this session the progress was reviewed and it was agreed to redefine some parameters of the project.





Sixth meeting: November 23, 2020, in this session we performed Quality Assurance. We scanned the First Increment Document (this Document) and the PowerPoint Presentation for error checking and for the final delivery and upload to Github.

Quality control:

The quality control was designed to be carried out through a meeting on the teams platform. The objective is that as a group we review the first advances and determine which are the points to improve, who should make these corrections, or where appropriate, confirm that the first phases have been carried out correctly.

Another important aspect of quality control that we just implemented is a strict version control workflow. Firstly, no member is allowed to make direct commits to the main branch of the repository, only pull requests should be merged. Also, every pull request should be reviewed by at least one more team member before merging it. We believe that this is the best way to deliver artifacts that comply with the required needs.

Teamwork

Repository:

We used a GitHub repository that contains general description of our product, documentation and the three principal increments that are being considered. The documentation mainly includes the documents templates and binnacles of the process.
https://github.com/FernanCetinaE/TeamOsiris

Metric of individual contribution:

The general contribution is calculated according to a formula that considers the average between the activities completed in time divided by Total activities plus the attendance divided by the highest attendance. Considering there exists more than three delays it will apply a sanction of 5% multiplied by the number of delays greater than 3.

Verification of the contribution %:

The verification is registered in a table composed of the total activities and how many were finished on time, times that the activity was delayed and the number of team meetings and meetings with the mentor that each member of the team attended(Contribution Tracker available in the Github repository). We registered the major part of our tasks and meetings in binnacles that are hold up in the GitHub repository, in the binnacles are described the main activities realized in that date and the key points that were
proposed in our meetings.

Subject proficiency

Generic and specific proficiencies in which the development process takes part are distinguished:

Generic proficiencies:

- Works with ICT in his/her professional interventions and in his/her private life in a suitable and responsible way.
- Works with others multi, inter and transdisciplinary environments in a cooperative way.
- Takes decisions in his/her professional and private practice in a responsible manner.

Specific proficiencies:

- Identifies the concepts linked to the phases of requirements, design, development, testing and maintenance, according to the recognized organisms of the discipline.
- Identifies human factors immersed in Software Development that contributes to the success of the Software project.
- Uses software engineering terminology properly in its professional interventions.

Demonstration of how the general proficiencies are met by specific activities of the development process:

- Works with ICT in his/her professional interventions and in his/her private life in a suitable and responsible way.
    - Throughout the whole project different tasks like “Work process” or “Meetings” require a different ICT tool, like meetings that are more accessible via TEAMS, Excel tracks better a checklist of contributions or Trello lets us visualize in a graphic way the pending activities. And for the future, more specific software like the ones used for modeling will also be tested to fit our needs and this abstraction thinking of deciding which one suits best is carried to our professional use of tools but also evolves our personal needs.

- Works with others in multi, inter and transdisciplinary environments in a cooperative way.
    - Some tasks like writing this document(from “Application definition” to “subject proficiency”) requires a different depth of knowledge, some team members are more skillful in english than others and we take advantage of that; in this way, tasks like creating a repository, the project vision, ways to organize and track progress and so on, are led by the ones with more experience that share their insights and helps the overall achievement of new concepts.
- Takes decisions in his/her professional and private practice in a responsible manner.
    - At the beginning of the increment each member makes a commitment based on their time and abilities (“Work process”), so being self aware takes great

relevance to make a responsible decision that affects your work and others. As joined tasks develop, the sub-teams are able to negotiate what is in the best interest of the daily scrum(e.g. in “Subject proficiency” Teodoro works on the repository management and Fernan on quality assurance) and in this sense, commiting to specific actions helps you develop a better decision taking mind.

Demonstration of how the specific proficiencies are met by specific activities of the development process:

- Identifies the concepts linked to the phases of requirements, design, development, testing and maintenance, according to the recognized organisms of the discipline.
    - All the phases(based on SWEBOK by IEEE) were considered in the creation of the timeline and have a determinate place in the organization of the documentation. We also considered making generic templates that can be filled according to the needments of the later phases. Activities like “Timeline” and “Scrum organization” are an example.
- Identifies human factors immersed in Software Development that contributes to the success of the Software project.
    - We considered dividing most of the activities and hash out the possible contribution of each member of the team depending on its abilities. We also prioritized the communication in the meetings and tried to be the most attentive as possible, we always maintained a respectful approach in the activities and valued the contributions of the different stakeholders. On the other hand in the early activity “Product definition” we asked users to give us their needs and for the next increment a new contact with them will occur.
- Uses software engineering terminology properly in its professional interventions.
    - Since our project was conceived and during “Product definition” and “Timeline” we adjust a holistic view of the development process. We take into account activities as: requirements eliciting, approving requirements, communication, scrum framework. In a broader perspective, we try to evaluate the feasibility of our project in question as “How are we going to develop this product?”, “Does there already exist another product that provides this service?”.