Software Requirements Specification for Process-Based-Activity-Manager

Version 1.2 approved

Prepared by

Apetrei Bogdan-Gabriel, 1309A

Atomulesei Paul-Costin, 1309A

Cornea Radu-Valentin, 1309A

Universitatea Tehnică „Gheorghe Asachi” din Iaşi

May 21, 2022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Cornea Radu	02.04.22	Prepared the requisite	Alpha 0.1
Apetrei Bogdan	29.04.22	Structured the Application and organized Tasks	Alpha 0.2
Atomulesei Costin	29.04.22	Created basic User Interface	Alpha 0.3
Cornea Radu	29.04.22	Added a Database for Processes	Alpha 0.4
Apetrei Bogdan	29.04.22	Merged Interface with Database to list on UI	Alpha 0.5
Apetrei Bogdan	07.05.22	Refactored the Code and improved its Logic	Alpha 0.7
Apetrei Bogdan	09.05.22	Program is ready to add Timestamps for Tasks	Alpha 0.8
Cornea Radu	13.05.22	Fixed but with Absolute Path for Database	Alpha 0.9
Cornea Radu	13.05.22	Database Tables create if they are not already	Alpha 1.0
Cornea Radu	15.05.22	Added Timestamps Table and minor Fixes	Alpha 1.1
Atomulesei Costin	15.05.22	Timestamps for Processes show on Interface	Alpha 1.2
Cornea Radu	15.05.22	Adapted Data Classes and Database to changes	Alpha 1.3
Cornea Radu	16.05.22	Timeslots with Active Running Processes	Alpha 1.4
Apetrei Bogdan	17.05.22	Merged timestamp functionality from M&V	Alpha 1.5
Apetrei Bogdan	17.05.22	New features for displaying Timeslots	Alpha 1.6
Apetrei Bogdan	18.05.22	Processes are now dormand and active	Alpha 1.7
Apetrei Bogdan	19.05.22	Fixed UI Bugs	Alpha 1.8
Atomulesei Costin	20.05.22	Interface additions with new Functionalities	Alpha 1.9
Apetrei Bogdan	21.05.22	Fixed Forms Bug with names	Alpha 2.0
Cornea Radu	21.05.22	Added Unit Tests	Alpha 2.1
Atomulesei Costin	22.05.22	Fixed UI bugs and added Functionalities	Alpha 2.2
Apetrei Bogdan	22.05.22	Merged Database with User Interface	Alpha 2.3
Cornea Radu	22.05.22	Fixed Database Singleton and Adapted it	Alpha 2.4
Apetrei Bogdan	22.05.22	Major Fixes and Additions in User Interface	Alpha 2.5
Cornea Radu	22.05.22	Improved Unit Tests, especially for Database	Alpha 2.6
Apetrei Bogdan	22.05.22	The application is ready to use	1.0
Cornea Radu	23.05.22	Removed redundant Code and Refactored it	1.1
Atomulesei Costin	24.05.22	Added Help for Application	1.2

1. Introduction

1.1 Purpose

There are many situations where people might want to see how much they've spent on a specific task, such as keeping track of someone's priorities. The application monitors what processes are running in the background and their durations within a day.

The list on the interface updates according to what processes are run or killed in the background, making it easier for everyone to keep track of the changes made. The user can click on a process name and see the timestamps when the application was used during the last 24 hours but is not limited to choosing 1 hour, 30, 5, 1 minutes intervals.

The database stores only the tasks that have less than a day. The other processes having a timestamp older than a day are removed from the database. The application also includes Help, to make it easier for everyone to get accustomed to the application.

1.2 Document Conventions

In making this document, IEEE standard formatting for software development was followed. The standard followed is for the text to be done in 3rd person, using passive voice, keeping the text clean, simple, informative, and correctly syntactically and grammatically.

The first page, which contains information regarding who the contributors are and the title, uses Times New Roman font, 12, 14, 32 sizes, Bold and Normal. Heading 1, which is used to describe the more general chapters of this document, uses Times New Roman font with size 18, Bold. Respectively, for Heading 2, Times font, size 14, and Bold were preferred. The phrases from each Heading block are written using Times New Roman font, size 12.

Depending on the things someone might look for when skimming this document, the bigger the size is, the more it might interest the reader. If the reader decides to dive into the details of a chapter, he should be able to see more words per line, hence using a littler size than the previously mentioned.

1.3 Intended Audience and Reading Suggestions

The targeted audience consists of team members (software architects, developers, tech leaders, designers, business analysts), directly involved clients, and any potential issuer or bug-finder who wants to report the error.

The SRS document contains the Revision History, in which the team commits new changes, updates, and fixes.

It is recommended for business people to read sections 2.1 and 2.2. For designers, the whole section 4, where they can find the features the application must have and should have. The architects can read section 5, where they can find the generalities about the application's requirements. The developers can read sections 3 and 5 to understand and later on, write the code to make the application itself.

1.4 Product Scope

The Process Based Activity Manager was made for everyone who needs to keep track of the most recent activities being used and their period. Not only does it give all the dates when an application was closed or opened, but also the total time used since a day.

One purpose of the application is to track someone's activity and how much time a person used on some applications. It can be used to see what employees are working and who are not. Another use might be to keep track of a child's activity and see all their applications used within a day, and in case of something, the parent will be able to act upon the situation and decide which solution is best for his child.

If someone needs to see what they worked on, where they lost time and where they performed well, this application is handy. The great thing about it is that it could be used by larger companies, but also by parents and individuals since it doesn’t demand training.

The only thing the user needs is a .NET framework to be installed, a computer with an internet connection, some space on the drive, and the desire to use that application in the first place. Besides that, if someone needs assistance with the application and its tools, a Help option was also provided. But the odds are people will get used to it relatively fast.

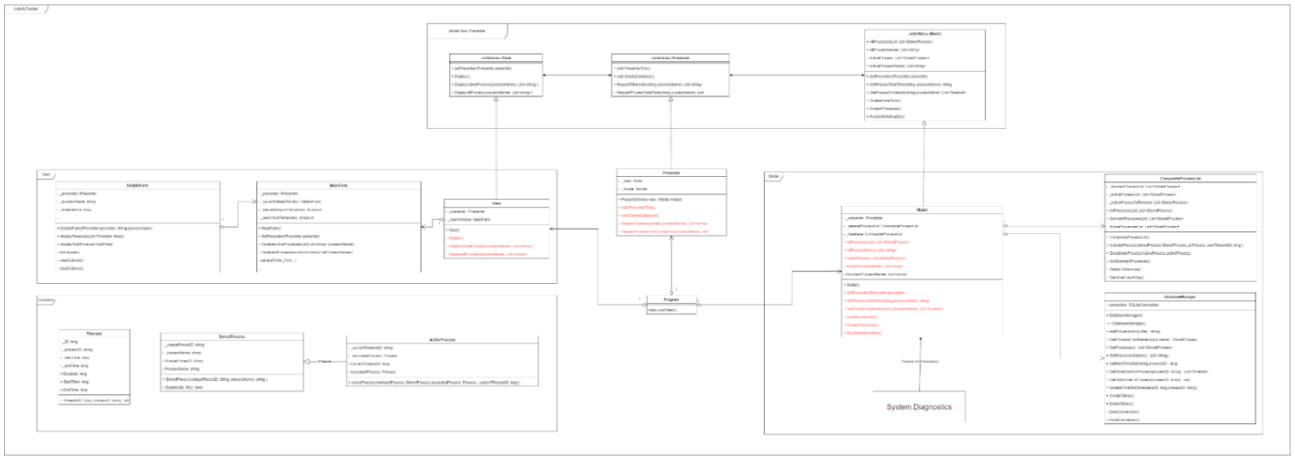
1.5 References

Some references used to make this particular document are:

- SRS Hero, Prepared by Nathan Harvey, Nathan Paul, Jacob Pearse, Kyle Spahn, Chris Steinberg, Version 1.0
- https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database

2. Overall Description

2.1 Product Perspective



Process-Based Activity Manager is an application that monitors and keeps evidence of what processes have been used and when. It is similar to Window's Task Manager, but it was designed so that instead of monitoring the Memory, GPU, Network, and other details regarding the performance and resources used by an application, it simply keeps some timestamps with the most recent applications.

The interface will be implemented with Windows Form App, using Visual Studio 2019. The database used is SQLite. A tool used to store and visualize the database is DB Browser for SQLite. GitHub was used to separate everyone's branch on each task individually using the Model View Presenter design pattern.

2.2 Product Functions

The fundamental functions that the Application must perform include:

- Help Tab – in case the end-user needs assistance with how to use the application and doesn't have anyone to answer their questions;
- Visualizing the processes that are currently running;
- Visualizing all the processes, including those who are and who are not running;
- Search processes by their names, Sort Ascending and Descending;
- Clean Database, from Options Tab, where all the processes details will be reset;
- Exiting the Application when the end-user decides to, without altering anything in back, for instance, a database.

2.3 User Classes and Characteristics

Process-Based-Activity-Manager can be used by everyone, including the end-user and the experienced fellows who have made applications by themselves. Due to its simplicity, it can serve many purposes. Some users that could use this particular application are a person who wants to know how much time they've spent on their text document, to keep track of what they've done and what they haven't; a mother (and her child), avoid any malicious places and see, in case the child does something, where the problem came from; a company who wants to see if their workers are indeed working or not, to see who is more productive and why and the vice-versa.

2.4 Operating Environment

The application can run only on Windows, with at least 30 MB RAM and 30 MB Space.

2.5 Design and Implementation Constraints

The project was made in around one month of continuous work in a team, but it is recommended to keep in mind that it was developed by a team of students in their 3rd year at the Gheorghe Asachi Technical University of Iași, who needed to do a project for Programming Engineering, and are only getting familiarized with the C# language and its powerful tools. During that time when the document was made, everything seemed to work as it should. However, it is possible that, in the future, versions of .NET, Windows, and other necessary tools to change and get deprecated, and, eventually, not for use anymore. It is also possible for Hardware to change in such a manner that it will become impossible to run the application. That is a great motivation for the team to scale the project and adapt it to the newer technologies at a specific time in the future. It is also possible for some libraries to get obsoleted. One possible way of preventing that is to keep different versions for the application and fork for different versions the specific updates accordingly. Another aspect to keep in mind is that SQLite might or might not update/remove its functions in such ways that they might need additional support in the Application made by the team.

2.6 User Documentation

The support provided for the end-user is within the Application, using the Help button, made with HelpNDoc.

2.7 Assumptions and Dependencies

It has been decided to assume a dark future and keep the worst possible outcomes, even if, as a team, hoping for the best should bring the brightest possibilities. Many situations, for instance, Windows version disappearing or getting obsoleted, .NET not being supported anymore and needing to pay for using it, SQLite getting shut down, and many others. There is no guarantee that the application might survive the time, but there is a guarantee that the team will do its best to make it last.

The project highly depends on using Windows, .NET, SQLite, and some other libraries. If these will disappear, the application might also suffer. The good news, however, is that nowadays many applications get the support they need to survive in the long run.

3. External Interface Requirements

3.1 User Interfaces

The User Interface was made so that anyone can easily access it. There is a tab with current processes running, where the user can double click on a process to view more details and a new tab with these details will be open. There are also two buttons to sort the processes according to their order. And there is a Search bar where one individual can type the name of a process and see only the processes that contain that search in their name.

3.2 Hardware Interfaces

The only things necessary to run this application are a mouse, to select the processes and click on the buttons and different sections from Help, a keyboard to look in the Search and the Help for keywords, and a monitor to have the interface displayed on. Regarding the hardware necessary for the application to run, there are not many requirements needed. Besides having a working computer, there shouldn't be anything else as a prerequisite needed.

3.3 Software Interfaces

The things the user needs to have installed are the .NET framework.

3.4 Communications Interfaces

The application does not require any internet connection and is only local, on the end-user's machine.

4. System Features

4.1 Tab for current running programs

4.1.1 Description and Priority

The application has a Current Running tab where the user can identify the processes that are currently running while the Process Tracker is also running. Running processes in the background are monitored in real-time, meaning that if the end-user quits an application, it will also be updated in the interface. It has high priority and must be working smoothly and efficiently.

4.1.2 Stimulus/Response Sequences

The end-user only needs to run the Process Tracker application and the tasks will be tracked automatically and shown in the Current Processes tab, leaving it to the user to only click on it if they decide to view it.

4.1.3 Functional Requirements

One important note worth mentioning is that only one instance of the process being run is monitored, meaning that if a person opens more instances of the same program, only one will be tracked. That decision was made to avoid redundancy.

4.2 Tab for all programs monitored

4.2.1 Description and Priority

There is also a tab to monitor all the processes that are running and that run at a time in the past. They updated their timestamps according to what the user does on their computer. It has a medium priority because there will only be the processes that ran within a day. However, it must be working smoothly and efficiently as well.

4.2.2 Stimulus/Response Sequences

The user has to run the Process Tracker application and the previous tasks, as well as the current tasks, will be tracked automatically and shown in the All Processes tab. The only thing left to the user is to click on that tab and see all the processes.

4.2.3 Functional Requirements

One important note worth mentioning is that only one instance of the process being run is monitored, meaning that if a person opens more instances of the same program, only one will be tracked. That decision was made to avoid redundancy. Another important aspect is that the processes traced there are at most within a day, the others simply being removed.

4.3 Viewable application use history (timestamps)

4.3.1 Description and Priority

Every process has associated it a list of timestamps in which the application was tracked and marked as running in the background at some time. That particular feature has High priority since that feature actually defines the purpose of the program, which is to track the programs and the time when they were used.

4.3.2 Stimulus/Response Sequences

The end-user only needs to click on a process within the application, either from the tab with Current Running Processes, either All Processes. The response will be a graph containing the times the application was used in the last 24 hours.

4.3.3 Functional Requirements

In order for it to work, the user needs to have the application run in the background on that day, else it is expected from the history of stamps to be empty in the worst case. It is also mandatory to keep whatever application the end-user wants to have or should have been monitored running in the background for it to function properly.

4.4 Help button for documentation

4.4.1 Description and Priority

A button where every user can see what and how should they do it is nice to have. If someone needs to know something more specific about a feature or the application, this is where they could find the most chances. It has a Medium priority and must be clear, informative, and easy to understand.

4.4.2 Stimulus/Response Sequences

The user needs to click on the Help button and they will be greeted by a new window where they can find the documentation for the program and what they need to know about it.

4.4.3 Functional Requirements

Regarding the Help button, if somehow the user manages to delete that file from the program, it is expected for it to not be displayed anymore and do nothing. If someone wants to click on Help and receives nothing, that is why they won't receive any output in that scenario.

4.5 Clean database button

4.5.1 Description and Priority

A button where the user can reset all the stored data, in case they want to get a new fresh start. Despite it not being mandatory, it is still nice to have.

4.5.2 Stimulus/Response Sequences

The user needs to click on the Options tab, where he can find the Clean Database button, and click on it if they want to truly reset everything.

4.5.3 Functional Requirements

The tables will be automatically deleted from the database, so the user should not be concerned with anything related to that.

4.6 Search bar for looking for process names

4.6.1 Description and Priority

In the search bar, the user can look up different fragments of process names. It has a low priority since it doesn't necessarily impact the application thoroughly, but it definitely makes the experience easier for users.

4.6.2 Stimulus/Response Sequences

The user writes in the Search Bar and receives only the processes whose names contain the Search written.

4.6.3 Functional Requirements

The processes shown on the interface will only be those that contain the characters included in the Search bar. There are no functional requirements for that feature.

4.7 Sort Ascending/Descending buttons

4.7.1 Description and Priority

The user can sort the processes. It has a low priority since it doesn't necessarily impact the application thoroughly, but it definitely makes the experience easier for users.

4.7.2 Stimulus/Response Sequences

The user clicks on these buttons and the application automatically sorts the lists shown in the interface.

4.7.3 Functional Requirements

The processes shown on the interface will be sorted, and nothing more, hence no functional requirements are needed.

4.8 Process tab

4.8.1 Description and Priority

The user can view details regarding a process where they should see the name, a bar, and the total time since it's been used. It has a high priority.

4.8.2 Stimulus/Response Sequences

The user clicks on the name of a process from the list and they are welcomed by the Details Window for that process, which contains the details mentioned earlier.

4.8.3 Functional Requirements

It is possible for the database to not be stored in the folder, but, the team assured it doesn't break when it doesn't exist and makes it on the spot when starting the program.

4.9 Displayed time Combobox

4.9.1 Description and Priority

The user can choose how to view the timestamps of a process. His options are 24 hours, 1 hour, 30, 5, and 1 minute. It has a low priority, but it is definitely nice to have.

4.9.2 Stimulus/Response Sequences

The user clicks on the Combobox called "Displayed time" and chooses his desired view of timestamps of the current process being looked on.

4.9.3 Functional Requirements

It is possible for the database to not be stored in the folder, but, the team assured it doesn't break when it doesn't exist and makes it on the spot when starting the program.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The program must run on at least the Windows 10 version and have at least 30 MB RAM and 30 MB Space.

5.2 Safety Requirements

People should note that the application is not dangerous in any physical way. It can be used by anyone safely without having put anything at risk regarding their physical health. Besides that, one possible risk of using it is when people decide to monitor other's people activity without their consent.

It is highly recommended to inform the people who will be using that program without knowing that their activity will be monitored and stored in the local machine. Because there might be people who believe their rights regarding their privacy might be violated, it is advised to let them know beforehand what comes next. However, if someone is employed in such an environment, they are assumed to have agreed on that rule whatsoever.

5.3 Security Requirements

This specific application is designed to be used by many different classes of users, so it has high usability with a steep learning curve. Anyone who tries the application should get along with it fairly easily. Its purpose is to monitor the timestamps of different applications, and the application should have high correctness to ensure accurate results. That is where Unit Testing comes into play and helps in testing edge cases.

5.4 Software Quality Attributes

This specific application is designed to be used by many different classes of users, so it has a high usability with a steep learning curve. Anyone who tries the application should get along with it fairly easy. Its purpose being to monitor the timestamps of different applications, the application should have a high correctness to ensure accurate results. That is where Unit Testing comes into play and helps at testing edge cases.

5.5 Business Rules

The team needed to follow the code of conduct imposed, and a set of rules, in order to make the application a minimum viable product:

- Making the SRS document following the IEE model
- Having a Help associated
- UML Diagrams for Use Case, Classes, Activities, Sequence
- Using Unit Testing
- Using a Design Pattern
- Having the Code commented
- Respecting the specifications for documentation
- General impression (with functional complexity, presentation etc.)
- Graphical User Interface
- Exception Handling
- Following C# Coding and Naming Conventions
- Using a separated DLL for each module
- Each file to have a Header with information regarding the author and functionality
- List of tasks made by each member of the team, without having the "Worked Together"

6. Other Requirements

Appendix A: Glossary

We don't use any glossary.

Appendix B: Analysis Models

The diagrams are included in the folder of the project.