The TODO Desktop Application

An Introduction to an Application to Help Struggling Programmers

Ryan Szilvasi
Computer Science
Virginia Tech
Blacksburg, Virginia, USA
rszilvasi@vt.edu

Tyler Streator
Computer Science
Virginia Tech
Blacksburg, Virginia, USA
tys305@vt.edu

Junxiang Feng
Computer Science
Virginia Tech
Blacksburg, Virginia, USA
junxiang@vt.edu

Xiaolong Xuan
Computer Science
Virginia Tech
Blacksburg, Virginia, USA
xxiaolong@vt.edu

Kechen Yu
Computer Science
Virginia Tech
Blacksburg, Virginia,USA
kechen21@vt.edu

ABSTRACT – Ryan Szilvasi

Every day, Computer Scientists are constantly worrying about their deadlines and work with changing requirements. It doesn't help that many industries forget to remind their workers of important deadlines for their programs. In an example provided by "Impact of time pressure on software quality: A laboratory experiment on a game-theoretical model," the authors explain that when delivering underdeveloped software solutions near a deadline, the developer must choose between a high-quality but late application or a low-quality application that meets the deadline requirements [1]. This TODO system that we are proposing will be able to help developers achieve their high-quality products without sacrificing their deadlines. This proposed solution will be able to notify individuals of the code they need to complete by the end of the day. The view of the application will have a high contrast so that important information can stick out to the developer. In addition, programmers can prioritize/rank their assignments in order of importance, or another order of their choice. Between Team Managers and Team Members, this application will be able to share and assign tasks to individuals on their team to ensure that everyone is meeting their deadlines.

CCS CONCEPTS

• Java • Windows • Linux • OS •

KEYWORDS

TODO Application, Deadlines, Programmers, Developers, Software, Computer Science

ACM Reference format:

Ryan Szilvasi, Tyler Streator, Junxiang Feng, Xiaolong Xuan and Kechen Yu. 2024. The TODO Desktop Application: An Introduction to an Application to Help Struggling Programmers

1 Introduction - Tyler Streator

As part of their work, programmers must regularly keep track of many different tasks at the same time. Any major programming project can be broken down into many subtasks that must be completed in order or in parallel, as well as meetings and other important events. Keeping track of these is complicated further by the possibility of changes in the requirements for the program partway through the project. The number of independent moving parts in the process can easily become too large to effectively keep track of, and developers could benefit from the ability to more easily see which tasks they need to accomplish within a given timeframe. Our proposed solution for this problem is an application to help programmers track tasks and events. The system would allow users to set daily or weekly goals for themselves or their teams, remind them of important deadlines, and sort their tasks based on importance or a variety of keywords. This would aid both groups and individuals in meeting their targets and help increase overall productivity.

2 Relevant software engineering tools -Xiaolong Xuan

Git: Version Control System

Git is an excellent and relevant software tool with powerful capabilities for managing source code changes, facilitating collaboration, and maintaining a history of project evolution. With its branch and merge capabilities, developers can develop distinct functions simultaneously without affecting the stability of the main codebase.

Branching capabilities enable developers to create separate lines of development for new features or bug fixes, which is critical to experimenting with and implementing new features without affecting stable versions of the application.

The merge capability enables changes in different branches to be consolidated back into the main branch, consolidating development efforts, and seamlessly integrating new functionality.[2]

Kanban: Enhancing Project Management

Task visualization: Kanban visualizes tasks at different stages of the development process. This visualization helps the team understand the workload and progress in detail, allowing for better planning and prioritization.

Workflow management: Kanban can organize tasks into columns and use cards to represent individual work items. The team can identify the process and measure the workload to ensure the sustainability and stability of the task in development.

Git integration: Kanban can be integrated with Git repositories to automatically update task status based on Git actions. For example, when the pull request is open, the task is moved to the Review bar, and when the pull request is merged, the task is moved to the Complete bar. This automation reduces manual updates and enables Kanban to accurately reflect project status. [3]

3 Software Engineering Process – Junxiang Feng and Kechen Yu

In this project, we chose the Prototyping Process. We're going to use a modified Prototyping Model for our software development process. The requirements for the software may evolve based on the problems and ideas we found in the project.

Analyze the project: We begin by collectively defining the initial project requirements. This stage includes discussion and agreement on the basic functionalities and goals of our software.

Building the Initial Prototype with Java and VS Code: For compilation, we use Visual Studio Code (VS Code), which is the best platform for our students who are new to software development.

Iterative Development with Git and Kanban: Our development process is basically an iterative process. We use Git to manage our code base. It allows us to update our progress in real-time and easily revert to a previous version when needed. In addition to Git, we also use Kanban for task management, giving us a clear visualization of project progress and facilitating seamless collaboration among team members.

Prototype Refinement through Team Feedback: Each prototype iteration is experienced and tested by our team. We give each other feedback and discuss and synthesize each member's point of view. This process helps us identify areas for improvement and improve the prototype accordingly.

Final Development and Integration: After a few iterations and improvements, our final version is here. This release passes all our proven features and functionality and gives users a great experience.

REFERENCES

[1] Basten, D., Müller, M., Ott, M., Pankratz, O., & Rosenkranz, C. (2021). Impact of time pressure on software quality: A laboratory experiment on a game-theoretical model. *PloS one*, *16*(1), e0245599. https://doi.org/10.1371/journal.pone.0245599

[2] Chacon, S., & Straub, B. (2014). Pro Git. Apress.

[3] Anderson, D. J. (2010). Kanban: Successful Evolutionary Change for Your Technology Business.