Software Engineering Laboratory
section 2
Project: Logic circuits

# Final Report

**1. Task**

Out task is to modernize an obsolete program so that it compiles on Visual Studio 2015 while introducing minimal changes to the original code.

**2. Section members and team organization**

Hasan Gündüz – code developer
Aleksandra Kotula – code developer, leader of Windows API migration
Mateusz Korusiewicz – QA tester
Michał Krypczyk – code developer, leader of cursor handling
Michał Męcik – QA tester, leader of QA testers
Burak Özkan – QA tester
Zbigniew Pamuła – code developer
Robert Pietrzyński – code developer
Mikołaj Stankiewicz – section leader, documentation master

Github: https://github.com/gilotyna815/SE-section-2---Logic-circuits
Trello: https://trello.com/b/rEfyeORV/se-section-2-logic-circuits

**3. Work plan progress**

Our work is divided into two stages: code analysis and code implementation and QA testing. These stages are described in Development Methodology document. Initially, we misunderstood the requirements of our task and begun rewriting the entire program despite one of the requirements being minimal changes to the code. In the end we returned to code analysis stage with better understanding of requirements. Currently the program is in final stage of code development – it compiles, but some errors were introduced to it while implementing Windows API and that causes the program to be non-functioning.

**4. Code analysis**

Code analysis has been performed and its effects have been described in Analysis and Requirements document. As a result of code analysis a set of requirements has been created to be fulfilled by code implementation and QA testing stage.

**5. Code implementation progress**

Functional requirements fulfilled:

1. Implement minor fixes – virtually all minor errors in the code stemming from platform and language change, as well as regular mistakes on the part of the original authors, has been already fixed.
2. Replacing obsolete graphics API
    2.1. No changes to GUI design itself
    2.2. Migration from Borland Graphics Interface to Windows API – however it should be noted that mistakes were made during migrations which cause program to be unuseable and should be fixed.
3. Remove obsolete cursor handling.

Non-functional requirements fulfilled:

1. Compatible with Visual Studio 2015
3. Minimal changes of the code
    3.1. All errors or suboptimal solutions in code which were not errors of compilation has been left unchanged.

Despite program not working properly, all requirements has been defined in such a way that they can be considered completed anyway. This stems from certain vagueness in our requirements formulation caused by inexperience in project management and highlights the importance of precise requirements' language.

The only unfulfilled requirement is separation of model from view. This is something we planned to implement once the code is running.

**6. Testing**

As of yet no testing have been performed, no part of code has reached that stage.

## 7. Rescue

As of now, the program is unusable because of mistakes introduced to it during API migration. These mistakes were caused by our inexperience with Windows API and not understanding its structure fully. This problem could have been avoided by designating a person to researching and understanding Windows API without any coding responsibilities while other developers would focus on practical side of coding.

As it stands right now, the Windows API message loop is implemented incorrectly and there is no controller for handling click events, as well as some non-essential functions which handle displaying program's logo are still not rewritten. These changes would require some more research into Windows API. Fixing those mistakes would not be a big task and the project could move onto QA testing fairly quickly.

## 8. Mistakes and conclusions

During our work on the project many mistakes were made on all stages and levels of development.

Initially we misunderstood the specifications of the tasks and formulated the requirements incorrectly, leading to wasting large amounts of time. This mistake could have been avoided by more careful analysis of task and perhaps engaging more people to team management, like separating roles of section leader and documentation master, which would introduce additional person to the process of task analysis.

The decisions of team organization were not ideal. QA team was chosen from the beginning and over the course of the project they didn't perform much tasks because no code entered QA. Also division of code development tasks into cursor handling and Windows API implementation was incorrect as the tasks were far more closely related and also, Windows API implementation was by far a bigger task. This caused extreme task imbalance which led to one API implementer actually fulfilling cursor handling task while working on API and cursor team was merged with API team very shortly into its existence when team API already worked out coordination and communication between each other. This lead to a small chaos and inefficiency, all while QA testers still didn't have anything to do. A good move was to include everyone in code analysis, which made analysis very efficient. In the end the categories were chosen not fittingly for the task at hand.

Also, work imbalance was increased by the fact that Trello usage was not equal among section members. Some people included everything they did as a Trello task, but some solved

problems as they were told without worrying about Trello tasks. As a result it was often a case that task was left on Trello for days after being completed or a new task needing completion was not added to Trello board. As a result, communication between members about what has to be done and by whom was made difficult especially since on Facebook not everyone is online at the same time so very short and simple communication can take several hours or maybe even a whole day, and not everyone attended Skype conference calls. Trello task imbalance also resulted in some people having much more tasks than others, although generally section leader tried to retroactively add tasks for work done if section members neglected it.

Another practice that was neglected by section members was self-documentation. GitHub commits often contained vague entries which gave little informations about changes made to code and despite self-documentation being demanded of sections members there was little documentation which renders understanding some changes impossible without their author's presence, if authorship can even be determined at all.

In the end the program doesn't even work as intended. As it was discussed above, it was because no one on the team was actually sufficiently familiar with Windows API. This situation could have been easily fixed with addition of Windows API expert who would focus on researching and understanding minutia of Windows API and giving tasks to code developers.

In summary, the project was plagued by many mistakes stemming from inexperience with project management, difficulty in communication and lack of skill. These mistakes highlighted the importance of many practices in software engineering which may be dismissed by programmers who never had real experiences with group projects. Even in relatively small team of nine programmers with regularly scheduled weekly physical meeting and Skype conferences communication was not always ideal and large amounts of problems abounded.

---

Version 1.0 – 27.11.2017
Version 2.0 – 04.12.2017
Version 3.0 – 19.12.2017
Version 3.1 – 22.12.2017