Group Project Documentation

Final report

Faculty of Electronics, Telecommunications and Informatics

Gdansk University of Technology

{model document version: version 2/2023}

Project name and acronym: *{name of the project, e.g.: Port security system against terrorist threats - SZP}* Redundant coding visualization app	**Principal:** *{customer name}* *Bartosz Czaplewski, PhD*	
Order number: *{number of the project team within the Group Project according to the SPG system, e.g. 13@KSSR'2022}* *5@KSTI'2023/24*	**Project manager:** *{project team leader}* **Bartosz Kołakowski**	**Client:** *{Client's name}* **Bartosz Czaplewski, PhD**

Document name/code: **Final report – RK**	**Version No.:** *{document version e.g. 1.00}* **2.00**
Responsible for the document: *{surname, first name}* **Kołakowski Bartosz**	**Date of first drafting:** *{date of first version of the document}* **29.05.2024**
	Last updated date: *{date of execution of the current version of the document}* **09.06.2024**
	Semester of Group Project: 2 *{do not change}*

Historia dokumentu

Version	Description of the modification	Chapter / page	Author of the modification	Date
1.00	*Draft version*	*whole*	*Jastrzębski Paweł*	*29.05.2024*
2.00	*Supplement to the annexes*	*point 3*	*Jastrzębski Paweł*	*09.06.2024*

Table of contents

1 Introduction - About the document

1.1 Full Document

{do not change }

The purpose of the document is to collect relevant information about the entire completed project in one place and present it in a clear way. The document is primarily intended to facilitate the evaluation of the project, in particular in the case of submitting it for an award or distinction. The expected length of the document is 3-6 pages (substantive part).

It is also necessary to indicate the work performed with a brief description, indicate discrepancies in the work performed in relation to the planned ones, summarize the work with a demonstration of teamwork and specify a list of documents produced in the project (final versions – the list includes documents from both semesters in the final versions) that have been placed and approved by the supervisor on the SPG website.

1.2 Document scope

{specifying what is included in the scope of the document and what is not, or indicating related documents}

The document describes our results of creating the project and changes with the original plan. In addition, the attachments are described in point 3.

1.3 Customer

{specification of the addressees of the document, it can be the type of recipient; here: the contractor (Department), members of the project team and named persons to whom the document is to reach}

The contractor and the client (Bartosz Czaplewski, PhD, Eng.) and members of the project team; Bartosz Kołakowski, Michał Mróz, Paweł Jastrzębski, Maksym Nowak, Piotr Noga

1.4 Terminology

{explanation of terms and abbreviations used in the document, designations used inside the document, e.g. requirements designations}

Abbreviations of people in the table:
Bartosz Kołakowski - BK
Michał Mróz - MM
Paweł Jastrzębski - PJ
Maksym Novak - MN
Piotr Noga – PN

2 Project results

2.1 Introduction – general description of the project

{In projects where part of a larger system or device is being implemented, the introduction must include a description of the entire system or device in its final form and a clear explanation of which parts of the system or device were available at the start of the project and which parts the project team has (or was expected to build) on its own. The introduction must also include a description of the parts of the system or equipment to be implemented as part of the project.
For projects that are a closed whole (device, layout, application), the introduction must include the general characteristics of the solution (device, layout, application) being created and its working environment.}

The project consisted in creating a computer application that aims to visualize the operation of redundant codes. The codes shown are the Hamming code and the Reed-Solomon code. The application is a whole, it was implemented from scratch. First of all, it must be available on Windows. No specific programming language was required, we decided to use C++.

The application was to show the process of encoding messages and decoding with permission to change the encoded message in order to simulate transmission errors. The final product is an executable file in Windows (.exe).

2.2 Purpose of the project and planned scope of implementation

{formulated project goal, possibly modifications of the goal during implementation, presentation of the initial project requirements set in the first and second semester, discussion of the planned method of implementation}

The aim of the project was to write a tool to support the teaching process. This tool is an application that visualizes issues related to redundant coding.

It was required to visualize 2 codes – Hamming in the first semester, Reed-Solomon in the second. The application also had to be available on Windows.

When choosing a programming language, we thought about C#, Java and C++. Some people were interested in using C#, some wanted to use Java because they knew it well and it was used in their work, but after discussions we came to the conclusion that it would be best to use C++, because it is the language that everyone knows best.

When it comes to creating a graphical interface, you can choose between Qt, FLTK, GTK and wxWidgets. After reading the opinions and the amount of help for given frameworks, we decided to choose Qt.

The planned method of implementation – C++ language, we divide tasks taking into account our strengths and weaknesses, the code is divided using a repository on GitHub, to which each member has access.
Planned list of tasks:

- creating a project database – initializing the repository, empty Qt project
- creating a GUI database
- implementation of Hamming code
- implementation of Reed-Solomon code
- visualization of Hamming code
- Reed-Solomon code visualization
- writing tests for Hamming code
- writing tests for the Reed-Solomon code

2.3 The actual scope of the project and discrepancies and the scope of work performed

{presentation of the project implementation process: hardware and/or programming environment used in the implementation of the project, factual description of the main work done in the first and second semester, with a brief description of them; there may be a table, indication of discrepancies from the plan from point 2.2, additional tools used, such as libraries, APIs, development kits or ready-made modules, electronic circuits, etc. (in each case, it is necessary to indicate which elements were produced entirely by the company and which were available in advance); The description must include the specification of the design requirements and their modifications introduced during the implementation of the project.}

The application was created in C++, in the Qt framework. .cpp files are used, which contain the implementation of the codes with their tests, and .qml files that describe the appearance of the graphical interface. We managed to meet all the requirements and complete all the planned tasks.

Completed tasks:

Name	**Description**	**Additional tools**
Creating a graphical interface	Basic appearance of the application – main menu, encoding selection, message entry, language selection	Created independently using the Qt framework (.qml files)
Hamming code implementation	Implementation of encoding and decoding according to the Hamming algorithm	Created independently in C++
Testy kodu Hamminga	Automatic implementation tests – checking if messages are well encoded/decoded	Created on my own using the testing framework – Google Test and the boost library
Hamming code visualization	Showing the performed operations step by step, the ability to change the encoded message before decoding. Communication between the implementation and the graphical interface	Created independently using the Qt framework (.qml files, ready-made communication mechanism between .cpp and .qml)
Implementing Reed-Solomon code	Implementation of encoding and decoding according to the Reed-Solomon algorithm	Created independently in C++
Testi Kodu Reida-Solomon	Automatic implementation tests – checking if messages are well encoded/decoded	Created on my own using the testing framework – Google Test and the boost library
Reed-Solomon code visualization	Showing the performed operations step by step, the ability to change the encoded message before decoding. Communication	Created independently using the Qt framework (.qml files, ready-made communication mechanism between .cpp and .qml)

	between the implementation and the graphical interface	
Translation	Saving the selected language to the configuration file. Translation of the graphical interface based on translation files.	Self-created system for reading translations from files, independent translations. Configuration file created automatically by Qt

Initially, the implementation, visualization, and testing of the Reed-Solomon code was to be done in the first semester, but due to the focus on improving the functionality and graphical user interface in the program. In addition, while working on the program, we came up with the idea to introduce two language versions.

2.4 Results achieved

{description of the final result; short description of the function/technical parameters; may or may not include suggestions for improvement, directions for further work.}

In the main menu of the application, you can first select one of the codes (the substantive description of the encoding is also shown), then you can enter the message that will be encrypted. The process of encoding messages is shown, during which you can adjust the speed of calculations. The encoded message can then be changed and decoded – again animated step by step. At the end, it is shown what message has been successfully decoded.

The Hamming code supports typing messages of any length (the required number of parity bits is automatically selected). The Reed-Solomon code requires 5 blocks of 3-bit data (numbers 0-7), 2 additional blocks are created.

Potential directions for further work are, first of all, to be able to implement more codings. Potentially, translations into new languages could also be added.

2.5 Characteristics of teamwork

{here it should be factually indicated who did what, how the project was managed, how tasks were assigned, indicate the contribution of individual people to the process of creating the product, documenting, operating the SPG system, cooperation with the client, supervisor, etc.; it should be shown how teamwork was carried out; there may be a table}

Task	BK	MM	PJ	MN	PN	Inception	Conclusion
creating a project database		X				31.10.2023	5.11.2023
creating a GUI database		X			X	31.10.2023	15.11.2023
implementation of Hamming code		X				03.11.2023	01.12.2024
implementation of Reed-Solomon code			X			21.02.2024	30.04.2024
visualization of Hamming code		X	X		X	05.11.2023	10.01.2024
Reed-Solomon code visualization	X		X			01.03.2024	31.05.2024
writing tests for Hamming code		X				05.11.2023	22.11.2024
writing tests for the Reed-Solomon code	X	X				01.03.2024	31.05.2024
writing an English translation					X	01.04.2024	31.05.2024

Documentation	BK	MM	PJ	MINUTES	PN
Schedule	X		X		
Poster	X				
Presentation	X				
Semester report	X				
Final report	X		X		
Technical documentation	X		X		X

Before moving on to the implementation of tasks, we planned group work so that individual tasks were assigned according to the capabilities of the given people, but also taking into account that the work was distributed fairly evenly. The schedule of tasks was such that most of the tasks were carried out at the beginning of semesters, not at the end, because then it is the period of study for the session and it would be difficult for us to carry out the project then. The goal was not to put off work until the last minute.

Code sharing was carried out using the GitHub platform. The pull request had to be accepted by at least 1 person from the group (apart from the author), which allowed to ensure good code standards.

Efficient group work was also possible thanks to regular meetings to work on the project (we were motivated to do so by lectures from the group project) and a quick way of communication (via Messenger and at the university – as we had other classes).

In addition, thanks to regular meetings with the project supervisor (every 2-3 weeks) and efforts to write documentation on a regular basis, it was easier for us to prepare the project and presentation for the project seminar.

The whole team usually appeared at meetings with the supervisor, unless one of the people did not have time at the time. During these meetings, the supervisor gave us a lot of valuable comments and possible improvements to our program, which we tried to address first.

The SPG system was operated by a project manager – he kept track of the deadlines for uploading documents, prepared or checked each document and uploaded it to the system. After uploading the document, he let the guardian know (which the guardian asked us to do – due to the lack of notifications after uploading)

3 Attachments

{applies to all documents from SPG - full list of current-final versions of documents, developed during the project implementation over 2 semesters, together with the current version of the final report; in particular:
- we include here a list of documents from the 1st semester (in accordance with the content of the semester report), but in the latest versions (if they were subject to correction, extension, etc. and marked with new version numbers)
- we put here a list of documents from the 2nd semester in the latest versions, including the final report}

Table. 3.1. Specification of the developed documents

No.	Document name	Name of the file placed in SPG
1	**Semester 2 - Detailed schedule**	HarmonogramSzczegolowySemestr2.doc
2	**Semester 2 - Information Poster**	PlakatSemestr2.doc
3	**Semester 2 - Technical Product Documentation**	PG_WETI_DTP.docx
4	**Semester 2 - Presentation of results**	PrezentacjaWynikowProjektGrupowySemestr2.pptx
5	**Semester 2 - Final Report**	PG_WETI_RK.doc