­­

Design Specification

For

DNA Data Extraction

Prepared by:

Group Name: Group 5

|  |  |
| --- | --- |
| Ben Sanders | sanders.199@wright.edu |
| Jordan Ellis | ellis.145@wright.edu |
| Caleb Smith | smith.2026@wright.edu |
| Frank Randazzo | randazzofr@gmail.com |
| April Stewart | stewart.300@wright.edu |

For:

Project Sponsor Applied Biosystems

|  |  |
| --- | --- |
| Nathaniel Adams | adams.201@wright.edu |

Document Tracking Information:

Document Version *1.0*

Date *02/21/17*

Final Approval Date *<xxxx.xx.xx>*

Final Appoval By: *<name> <e-mail>*

*<name> <e-mail>*

Table of Contents

1. Executive Summary 1

1.1 Purpose of this document 1

1.2 Design Scope 1

1.3 Intended Audience and Document Overview 1

1.4 Definitions, Acronymns, and Abbreviations 2

1.5 References and Acknowledgments 2

2 Problem Statement 2

2.1 Historical Introduction 2

2.2 Market Analysis and Relevant Art 2

2.3 Alternative Approaches 2

2.4 Impact of Success 3

3 Context of Design Solution 3

3.1 Design Objectives 3

3.2 Design Assumptions 4

3.3 Design Requirements 4

3.4 Design Functionality 5

3.5 User Characteristics 5

3.6 Operating Environment 6

3.7 User Documentation 6

4 Technical Approach 6

4.1 Hardware 6

4.2 Software 6

# 1. Executive Summary

This document outlines the proposed system design for DNA Data Extraction, the new system whose purpose is to extract data from several DNA data file types. It will be capable of reading binary data and transforming it into a human readable form.

## Purpose of this document

The purpose of this document is to describe in sufficient detail how the proposed system is to be developed. This document translates the requirements document into a guide from which the developers can create the actual system. It identifies top-level architecture and design schemes.

## Design Scope

The overall goal of this project is to construct a data extraction utility that includes the following capabilities:

1. Compatibility with .fsa and .hid files generated by the families of ABI 310, 31xx, and 35xx genetic analyzers and their according data collection programs.
2. Extracts raw, binary signal data from these files.
3. Extracts metadata from these files.
4. Preserves this data/metadata in a human- and machine-readable format.
5. Operated from a command line interface.

This utility will be beneficial to any user who wishes to quickly read files of the .fsa and .hid file type. The output that this utility will produce will allow for very quick searching for the necessary data the user requires.

Delivery of a product that implements the above capabilities as described in this and other documents shall constitute a complete deliverable.

## Intended Audience and Document Overview

This document’s intended audiences are project sponsors, program managers, and project reviewers (Professor Reisner). Project sponsors will refer to this document in order to understand how developers intend to implement the system. By doing this, project sponsors will be able to understand the direction developers are taking to provide a useful utility for the sponsors. Sponsors will then be able to adjust developers’ design in a way they believe to better suit the task at hand. Program managers will refer to this document to keep their developers in the correct scope during design. Project reviewers will use this document to understand the design approach the team has taken.

## Definitions, Acronymns, and Abbreviations

ABIF Applied Biosystems Inc. File

DNA Deoxyribonucleic Acid

JSON JavaScript Object Notation

## References and Acknowledgments

[1] Applied Biosystems. "ABIF File Format Specification and Sample File Schema." Applied Biosystems Genetic Analysis Data File Format (2009). Web. 21 Feb. 2017.

# Problem Statement

## Historical Introduction

Applied Biosystems historically has accessed .hid and .fsa files through a basic hexadecimal text reader. They would open the file with the corresponding information they wished to access. Once the file was open, they would use the find tool of the text reader to search for specific header tags of the data they wanted. Then they could read the data that they were looking for. This has been an issue due to the time required to find the information they required. This is the issue our utility is striving to solve.

## Market Analysis and Relevant Art

This system’s intended use is for internal use at Applied Biosystems only.

## Alternative Approaches

Since the system implements the extraction of several different file types, there are a few different ways to implement a satisfactory solution. One method of implementation is with rapid development in mind. The other is with extensibility in mind. The former method is a straightforward approach where we develop the solution to create the fastest solution according to exactly how the file types are created. This solution is acceptable and would be satisfactory. The other approach takes into account the fact that file types sometimes change. This would be more costly in regards to time. Even though it would take more time, the solution would require very little adjustments to adapt to file type changes. It would be a long-term solution as opposed to the quicker solution. The long-term solution is the ideal solution we intend to develop.

There are also multiple approaches in regards to the choice of technology used. We have chose between which programming language to use to develop the utility. One option was Java. This option would be a slower option, but it would have provided us with a highly object oriented programming language. This would allow for extensibility and flexibility within the design structure especially if the data file types were to ever change. The other option was Python. This option is much faster, however, it doesn’t allow for the same type of object oriented programming like Java. The issue we decided to be most concerned with though was speed. Since these files are quite large, we didn’t want the users to be waiting for long periods of time for the program to complete. We want a fast, robust solution, thus we chose Python as our programming language.

## Impact of Success

Upon successful completion of the product, users will have the ability to use the utility at any time they may need it. The utility will provide them will a rapid way of obtaining the information they need. Rather than having to search through a binary file for the single bit of information that they need, users will now be able to run our utility to obtain whatever pieces of information they may need in one quick process. Not only that, but they will also be able to know with certainty that the information that they are receiving is accurate on top of being quickly accessible. The information now required will be more available and more quickly accessible to the users.

# Context of Design Solution

## Design Objectives

The product is designed to be a utility for more efficient workflow for employees at Applied Biosystems. The utility has two main functions, to extract data and to preserve the data in a human-readable and machine-readable form. The following objectives will be achieved:

1. The system will extract data and metadata from .fsa file types.
2. The system will preserve the data and metadata in a human-readable and machine-readable format.
3. The system will operate from the command line.

Objective one will be achieve by utilizing Python and the libraries available to add capabilities to Python. Python is capable of reading raw binary data and converting it to useful information that can be used by humans or machines when necessary.

Objective two will be achieved through Python and JSON. JSON is data-interchange format that is very lightweight and capable of being utilized by many different computer science technologies. Not only that, it is also a human-readable format that can be saved as a .json file for use in the future, thus satisfying and preserving both formats necessary in objective two.

Objective three will be achieved through the use of Python. Python programs or scripts can be run directly from the command line, and, in fact, the intent of the Python programming language is that they are supposed to be executed this way. The Python executable must simply be placed in the environment variables of the computer first, and then any Python script can be run directly from the command line in any directory on the computer.

## Design Assumptions

The following assumptions have been made in order to fulfill our design choices:

DA-1: The Python executable is readily available for use by Applied Biosystems.

DA-2: There are readily available .fsa files that can be used for testing purposes with our system.

## Design Requirements

1. DNA Data Extraction shall be compatible with files generated by the families of ABI 310, 31xx, and 35xx genetic analyzers.
2. DNA Data Extraction shall be compatible with files generated by all versions of ABI Data Collection Software published in 2016 and earlier.
3. DNA Data Extraction shall be compatible with both .fsa and .hid file types.
4. DNA Data Extraction shall translate and preserve all data specified by tags described in the ABIF file format specification.
5. DNA Data Extraction shall preserve all output data in a human- and machine-readable format.
6. DNA Data Extraction shall adhere to an internal schema for data preservation/output. Operating Environment
7. DNA Data Extraction shall be agnostic to hardware.
8. DNA Data Extraction shall operate with full functionality on Windows, Apple OSX, and Linux.
9. Linux testing shall be performed on Ubuntu.
10. A user manual shall be provided with each distributed copy of DNA Data Extraction or made available on a public-facing website linked to by the program.
11. The command line interface shall accept a file or folder as an input parameter.
12. The command line interface shall accept a file or folder as an output parameter.
13. The command line interface shall accept a specifier for recursion.
14. The command line interface shall require no more than one human interaction with the program via the command line interface. All parameters shall be provided to the program at once.
15. Aside from locking input files, DNA Data Extraction shall not affect the state or content of any file provided as input.
16. DNA Data Extraction shall accept as input a path pointing to one file OR one folder.
17. The input file shall be checked for an .fsa or .hid extension. The user shall be notified if the file is incompatible.
18. The contents of the folder shall be checked for .fsa or .hid extensions. Only .fsa and .hid files shall be operated on.
19. If an input file is specified, the output shall be written to a file at the provided output path. If a file of the specified name exists at that path, that file shall be overwritten with no warning to the user.
20. If an input folder is specified, the output shall be written in the same relative subdirectories with the same file names (except for file extension) as included in the input folder.
21. If a recursion specifier is provided, DNA Data Extraction shall recurse through the input directory’s subdirectories. If no recursion specifier is provided, DNA Data Extraction shall operate on only those files contained directly within the input directory.
22. A hierarchical schema shall be specifiable as a program configuration.
23. The schema shall be specified in a human-readable format.
24. The schema shall be specified in a key-value relationship where keys are the data tags specified in the ABIF file format specification and the values are a structure of number, type, and description.
25. The schema shall be specifiable in a list. E.g. one key may point to multiple subkeys in a list.
26. A data tag may exist only once within the schema, except under conditions of compatibility.
27. If necessary, tags that exist in multiple genetic analyzer or Data Collection Software file format specifications with different specifications shall be differentiated in the schema by machine model and/or Data Collection Software versions.
28. The schema shall support custom data tags not specified in the ABIF file format specification.
29. The output format shall be human- and machine-readable.
30. The order of tags in the output file shall match the order and hierarchy of the tags in the configuration schema.
31. All data in an input file matching a data tag specified in the ABIF file format specification shall be preserved in the output file.
32. All data for data tags specified in the ABIF file format specification shall be translated with full fidelity to the output file.
33. Data tags not specified in the ABIF file format specification have no guarantee to reliability or fidelity of translation.

## Design Functionality

* The user will navigate to the command line and type in a command to run our utility.
* The utility will extract data and metadata from files according to the given input from the user.
* The utility will manipulate the binary data into a human-readable and machine-readable format.
* The utility will store the results of the extraction in a .json file.
* The utility will display to the user feedback to let the user know that it has completed.

## User Characteristics

Users of our utility will be employees of Applied Biosystems. It is assumed that these employees have a basic knowledge of the JSON file format and are capable of using a JSON file to obtain the information they require.

## Operating Environment

The environment in which this utility will operate is in any location on any computer running a Windows, Linux, or OSX operating system. Any connection to Internet is unnecessary for our utility to work correctly.

## User Documentation

Users will receive a user manual detailing exactly how the program is to be run. It will cover what needs to be done to set up your computer so that it is capable of running the utility as well as all the possible input you could send our utility through the command line. By giving the user a user manual, they should be fully capable of understanding how to run our program without any further assistance.

# Technical Approach

## Hardware

This section is not applicable. Our system will be capable of running on any computer running a Windows, Linux, or Apple OSX operating system.

## Software

There is only one true use case for our software. This case would be that a user is in need of reading a binary file of file type .fsa or .hid. That user will then enter a command into the command line to run our utility giving the correct input of those files’ locations on the file system. From that point, our system will extract the data from those files and save them to JSON files and give feedback to the user to inform them of completion. By completing these tasks, our utility will have satisfied all requirements of the system.