Department of Computer Science and Engineering  
The University of Texas at Arlington

Detail Design Specification

Team: Ink3D

Project: 3-D Printer Fabrication System

Team Members:   
*Daniel Lain*

*Timothy Edmonson*

*Shawn Simonson*

*Jesse Bowles*

# Table of Contents

[Table of Contents 2](#_Toc379703396)

[1 Introduction 4](#_Toc379703397)

[2 Architecture Overview 4](#_Toc379703398)

[2.1 Module Decomposition Table 4](#_Toc379703399)

[2.2 Module Decomposition Diagram 4](#_Toc379703400)

[2.3 Module Data Flows Table 4](#_Toc379703401)

[2.4 Producer Consumer Table 4](#_Toc379703402)

[2.5 Module Functional Descriptions 4](#_Toc379703403)

[2.5.1 Use Interface Layer Modules 4](#_Toc379703404)

[2.5.2 Preprocessing Layer Modules 4](#_Toc379703405)

[2.5.3 Processing Layer Modules 4](#_Toc379703406)

[2.5.4 Post Processing Layer Modules 4](#_Toc379703407)

[2.5.5 Physical Layer Modules 4](#_Toc379703408)

[2.5.6 Printer Feedback Layer Modules 5](#_Toc379703409)

[2.5.7 Communication Layer Modules 5](#_Toc379703410)

[3 User Interface Layer 5](#_Toc379703411)

[3.1 GUI Subsystem Modules 5](#_Toc379703412)

[3.1.1 Import GUI Module 5](#_Toc379703413)

[3.1.2 Material GUI Module 5](#_Toc379703414)

[3.1.3 Configuration GUI Module 6](#_Toc379703415)

[3.1.4 Print GUI Module 6](#_Toc379703416)

[3.1.5 Status GUI Module 7](#_Toc379703417)

[3.2 Controller Subsystem Modules 7](#_Toc379703418)

[3.2.1 Import Controller 7](#_Toc379703419)

[3.2.2 Material Controller 8](#_Toc379703420)

[3.2.3 Configuration Controller 8](#_Toc379703421)

[3.2.4 Print Controller 9](#_Toc379703422)

[3.2.5 Status Controller 9](#_Toc379703423)

[3.3 Database Subsystem Modules 10](#_Toc379703424)

[3.3.1 Persistence Framework 10](#_Toc379703425)

[3.3.2 Command Structure 10](#_Toc379703426)

[4 Preprocessing Layer 11](#_Toc379703427)

[4.1 Normalization Subsystem Modules 11](#_Toc379703428)

[4.1.1 File Translation Module 11](#_Toc379703429)

[5 processing Layer 12](#_Toc379703430)

[5.1 Slicing Subsystem Modules 12](#_Toc379703431)

[5.1.1 Slicing Engine Wrapper 12](#_Toc379703432)

[5.1.2 Slicing Engine 12](#_Toc379703433)

[6 Post processing Layer 13](#_Toc379703434)

[6.1 G-Code Preparation Subsystem Modules 13](#_Toc379703435)

[6.1.1 Parser Module 13](#_Toc379703436)

[6.1.2 Unification Module 14](#_Toc379703437)

[7 Physical Layer 14](#_Toc379703438)

[7.1 [Subsystem] Modules 14](#_Toc379703439)

[7.1.1 [Module Name] 14](#_Toc379703440)

[8 Printer Feedback Layer 15](#_Toc379703441)

[8.1 [Subsystem] Modules 15](#_Toc379703442)

[8.1.1 [Module Name] 15](#_Toc379703443)

[9 Communication Layer 15](#_Toc379703444)

[9.1 [Subsystem] Modules 16](#_Toc379703445)

[9.1.1 [Module Name] 16](#_Toc379703446)

[10 Quality Assurance 16](#_Toc379703447)

[10.1 Unit Testing 16](#_Toc379703448)

[10.1.1 [Layer Name] 16](#_Toc379703449)

[10.2 Component Testing 16](#_Toc379703450)

[10.3 Integration Testing 16](#_Toc379703451)

[10.4 System Verification Testing 16](#_Toc379703452)

[10.5 Test Cases 17](#_Toc379703453)

[11 Requirements Traceability Matrix 17](#_Toc379703454)

[12 Acceptance Plan 17](#_Toc379703455)

[12.1 Package and Installation 17](#_Toc379703456)

[12.2 Acceptance Testing 17](#_Toc379703457)

[12.3 Acceptance Criteria 17](#_Toc379703458)

[13 Appendices 17](#_Toc379703459)

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision Number | Revision Date | Description | Rationale |
| 1.0 | 2/20/14 | Gate Review Version | Final changes prior to gate review |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

List of Figures

Figure # Title Page #

1-1 System Concept ??

2-1 Architecture Diagram ??

2-2 Decomposition Tree ??

2-3 Detailed Design Diagram ??

List of Tables

Figure # Title Page #

2.2 External Inputs and Outputs 11

11.1 Cost Analysis Table 35

11.2 Function Points Analysis 36

11.3 Influence Multipliers 38

11.4 Jones First Order Estimation 39

11.5 Size Estimate – Lines of Code 39

11.6 CoCoMo Estimate 40

11.7 Estimate Comparison 40

# Introduction

## Overview

The Detailed Design Specification (DDS) provides a detailed specification of the subsystem and module breakdown of the 3-D Printer Fabrication System. The document is intended to use detailed constructs to represent implementation. The document focuses on low level concepts such as algorithms, module definitions, and data flows in such a way that the system can be implemented without requiring any further design work.

## Project Concept

Mainstream 3-D printers and their respective tool chains are suited for printing a 3-D model from an abstract model developed from Computer Aided Drafting (CAD) software. The software tool chain is capable of issuing instructions to the printer in the form of G-Codes. The printer then prints the object layer by layer as an additive process by depositing a heated polymer or plastic onto the print bed iteratively until the final object is realized. The limitation of this style of tool chain, however, is that it is not suited for depositing multiple materials interleaved with each other on a single print run.

The 3-D Printer Fabrication System aims to remove this limitation by providing a software tool chain that will read and interpret Stereo Lithography (STL) files with varying material parameters into a single stream of G-Codes that will be interpreted and executed by a device to be implemented by a Mechanical Engineering team at the University of Texas at Arlington. This device will be capable of depositing multiple materials within a single print run.

## Project Scope

The scope of the 3-D Printer Fabrication System is to develop software that will produce suitable machine code for a 3-D printer head that is capable of depositing multiple materials within a single print run. The system will present the user with an interface that will allow them to specify which STL files are to be loaded and specify the material properties of the respective STL files. The system will then use this information to process the geometry such that a suitable set of G-Codes can be issued to the device. The system will also provide a method for streaming the information to the printer control hardware via a serial interface. The system is intended to be used by 3-D printer operators, CNC operators, Dr. Shiakolas, and other experienced operators in the research field. The system is not intended for the consumer market.



Figure 1-1: System Concept

# Architecture Overview

The 3-D Printer Fabrication System is intended to be a software system that initially provides the core functionality necessary to print multi-material objects on a multi-extruder 3-D printer, but can be easily expanded in the future to become a feature rich multi-material 3-D printing software suite. This goal dives the need for a modular, configurable, extensible, and portable system. To meet these needs the architecture has been broken in to seven layers with the design constraint that each layer should be replaceable and modular for future expansion. The User Interface Layer is responsible to providing a GUI for the user to enter configuration date, select print options, load and save configuration, load object files, and start the print process. The Preprocessing Layer takes the configuration and object files provided by the User Interface Layer and translate the files into a correct form for the processing layer. The Processing Layer takes the configuration and translated object files and produces an instruction set in the form of G-Codes for the Post Processing Layer. The Post Processing Layer uses the instruction sets created by the Processing layer and the configuration data to build the complete series of G-Codes in the proper format for the Printer Control Layer to use. The Printer Control Layer uses the G-Codes and configuration to stream the instructions one by one to the Communications Layer. The Printer Control Layer also uses a pause, stop, resume command from the User Interface Layer based on the user input. The Printer Control Layer must also provide printer state monitoring and maintain bounds control to ensure safe and correct printer operation. The Communications Layer is responsible for serialization, deserialization, and direct serial communication with the printer. The Communications Layer uses the instructions issued by the Printer Control Layer to serialize those instructions then send them to the printer through the serial interface. The Communications Layer poles the printer for state information then de-serializes this information and sends it to the Printer Feedback Layer. The Printer Feedback Layer is responsible for sending printer state information to the Printer Control Layer and the User Interface Layer.



Figure 2-1: Architecture Diagram

## Module Decomposition Tree

The 3-D Printer Fabrication System illustrates its modular design. The system is decomposed into seven layers. Each layer is decomposed to one or more subsystems that have related functionality. Each subsystem is then decomposed into one or more modules that support that subsystems function.



Figure 2.2: Decomposition Tree

## Module Decomposition Diagram



Figure 2.3: Module Design Diagram

## Module Data Flows Table

[Description]

[Module Data Flows Table]

## Producer Consumer Table

[Description]

[Producer Consumer Table]

## Module Functional Descriptions

### Use Interface Layer Modules

[Module Functional Descriptions]

### Preprocessing Layer Modules

[Module Functional Descriptions]

### Processing Layer Modules

[Module Functional Descriptions]

### Post Processing Layer Modules

[Module Functional Descriptions]

### Physical Layer Modules

[Module Functional Descriptions]

### Printer Feedback Layer Modules

[Module Functional Descriptions]

### Communication Layer Modules

[Module Functional Descriptions]

# User Interface Layer

The Interface Layer’s purpose is to collect user information and serve it to the processing layers. There are three subsystems providing data to lower levels. The GUI subsystem provides the user the ability to interact with the system and provides the ability to import files, create material profiles, set general configuration, and print specific configuration. The database subsystem provides a persistence framework for the other subsystems in this layer for storage and retrieval of configuration and material information. The controller subsystem decouples the user interface from the program logic. The controller provides the GUI with the information to present and takes in the user actions allowing the initiation of a print run, then packing the collected configuration, object files, and material data in to a print request object for the preprocessing layer.

## GUI Subsystem Modules

The GUI Subsystem is responsible for providing an interface to the user so that he or she can import model files and manage print input configuration information related to materials and printing hardware. The GUI Subsystem is also responsible for providing an interface from which the user can set configuration options for a specific print and initiate a print. During an ongoing print, GUI Subsystem is responsible for displaying the printer and current print status to the user.

### Import GUI Module

#### Prologue

Import GUI Module is a button or menu option that begins the event to import an object file

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

### Material GUI Module

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

### Configuration GUI Module

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

### Print GUI Module

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

### Status GUI Module

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

## Controller Subsystem Modules

The Controller Subsystem is responsible for collecting all the material and printer configuration information required for the print, bundling that information with the object file information received from the Import Subsystem, and sending that bundle to the Pre Processing Layer to begin the printing process. The Controller Subsystem is also responsible for telling the Printer State Controller to pause, resume, or stop a print job based on user input.

### Import Controller

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

### Material Controller

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

### Configuration Controller

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

### Print Controller

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

### Status Controller

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

## Database Subsystem Modules

The Database Interface subsystem is responsible for providing an abstract interface between the database of the system and any subsystems that need to store or retrieve information from the database. As such, the Database Interface Subsystem is responsible for exposing all the methods necessary for the other subsystems to communicate with the database in an abstract manner.

### Persistence Framework

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

### Command Structure

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

[Pseudo-code]

# Preprocessing Layer

The Preprocessing Layer provides an abstract interface between the User Interface and the Processing Layer. This layer‘s purpose is to translate and repackage the print request object in to the format that the processing layer needs. The Preprocessing Layer starts by receiving all of the necessary configuration and object data from the User Interface’s Print Subsystem then converts it into a unified format that the Processing Layer understands. In the current iteration of this project, the Preprocessing Layer only has one subsystem, the Normalization Subsystem and will be converting STL files it receives from the object sent to it into an AMF file. The Preprocessing Layer then packages the configuration, object definition, and material data into the correct format for the Processing Layer.

## Normalization Subsystem Modules

The Normalization Subsystem is responsible for receiving the bundle containing the object data and printer configurations from the User Interface’s Print Subsystem and converting the object data into a format the Processing Layer can understand. After the object file(s) is converted, this subsystem sends a modified bundle of the object file information and the printer configuration to the Processing Layer so it can calculate a printing path.

### Object Subsection Module

#### Prologue

The Object Subsection Module splits objects into multiple subsections about the z axis and produces a new STL file to represent each subsection.

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
| Subsection | STL File, bottom z, top z | Subsection STL File |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

subsectionConfigurationList = printJobConfiguration.getSubsetConfigurationList()

for each subsectionConfiguration in subsectionConfigurationList:

stlFile = subsectionConfiguration.getParentStlFile()

bottomZ = subsectionConfiguration.getBottomZ()

topZ = subsectionConfiguration.getTopZ()

subsectionStlFile = createSubsection(stlFile, bottomZ, topZ)

subsectionConfiguration.setStlFile(subsectionStlFile)

### File Translation Module

#### Prologue

The File Translation Module translates multiple STL files, each describing a part of the object that is a different material, into a single AMF file. The produced AMF file describes the object as a whole while also containing information mapping parts of the objects to the material that the part consists of.

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
| Translate | Subsection Configuration Object | Modified Subsection Configuration Object |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

create new materialFile

create new vertexFile

sort file configuration list by extruder number

for each file configuration:

write material definitions to materialFile

for each file configuration:

create new volumeFile

add material attribute to volumeFile

stlFile = get subset stl file from file configuration

for each face in stlFile:

for each vertex in face:

write vertex to amf file

write face to volume file

amfFile = concatenate(materialFile, vertexFile, volumeFile)

store AMF file in SubsetConfiguration

# processing Layer

The Processing Layer takes the formatted package and translates it into G-codes. The slicing engine is the only subsystem in this layer. The slicing engine needs to be able to be replaced based on the growing needs of future development. The Processing Layer is designed to be expandable. The initial implementation will use Slic3r as its engine, but the normalized data from the Preprocessing layer will need to change if the slicing implementation changes. This allows the slicing engine to be easily replaced with a different implementation and provides the future possibility of allowing the user to select which slicing engine they wish to use in any given print. One the Processing Layer processes the normalized data from the Preprocessing Layer, the Processing Layer passes a packet of G-Codes and printer configuration information to the Post-Processing Layer.

## Slicing Subsystem Modules

The responsibility of the Slicing Engine is to read the object file and divide the geometry of the object into appropriate layers. Then for each material object of each layer, the subsystem will draw out a printing path for the head to follow. Once the object has been drawn out, the printing instructions will be converted to G-Codes and passed to the Post-Processing Layer for final changes to the instructions.

### Slicing Engine Wrapper

#### Prologue

The Slicing Engine Wrapper is responsible for wrapping an abstract API around the slicing engine of the system. The wrapper takes a normalized Print Job Configuration Object, writes the properties from that object to a configuration file readable by the slicing engine, then uses the configuration file as a parameter to generate G-Codes for each subsection found in the Print Job Configuration Object.

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
| GenerateGCode | Print Job Configuration Object | Modified Print Job Configuration Object |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS]

#### Process

configFile = new File().openForWriting()

printerConfiguration = PrintJobConfiguration.getPrinterConfiguration()

for each property in printerConfiguration:

write property to configFile

subsectionConfigurationList = printJobConfiguration.getSubsetConfigurationList()

zOffset = 0

for each subsectionConfiguration in subset:

write zOffset to configFile

printConfiguration = subsetConfiguration.getPrintConfiguration()

for each property in printConfiguration:

write property to configFile

amfFile = subsetConfiguration.getAMF()

gCode = run slicing engine with amfFile and configFile as parameters

subsectionConfiguration.setGCode(gCode)

zOffset += subsetConfiguration.getEndZ()

### Slicing Engine

#### Prologue

The Slicing Engine is responsible for slicing 3-D objects into printable layers based on provided configuration data.

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS

#### Process

[Pseudo-code]

# Post processing Layer

The Post Processing Layer receives the G-Codes from the Processing Layer and modifies the instructions to match special considerations the printer may need. G-Codes are somewhat standard, but some printers have instructions that are specific to that printer only. It is the job of the Post Processing Layer to correct the G-Code received from the Processing Layer to accommodate for those special instructions. Once finished, the Post Processing Layer will output G-Codes and the configuration to the physical layer. Similar to the Preprocessing and Processing Layers, the Post-Processing layer is designed to be easily expanded in future iterations. Initially, this layer will only support the printer supplied by the Mechanical Engineering Team.

## G-Code Preparation Subsystem Modules

The responsibility of the G-Code Preparation Subsystem is to modify the G-Code produced by the Processing Layer to be 100% compatible with the printer being used. This subsystem therefore must accommodate for any special instructions that are unique to that printer.

### Parser Module

#### Prologue

The Parser Module is responsible for parsing G-Code data it receives and modifying G-Codes to ensure that they are compliant with the G-Code standard required by the printing hardware.

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS

#### Process

[Pseudo-code]

### Unification Module

#### Prologue

The Unification Module is responsible for combining separated G-Codes into a single string of G-Codes that can be streamed to the printer. When parts of an object are sliced separately, the Unification Module must be used to combine the separated G-Codes together.

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
| combineSubsections | Print Job Configuration Object | Modified Print Job Configuration Object |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS

#### Process

subsetConfigurationList = printJobConfiguration.getSubsetConfigurationList()

unifiedGCodeFile = new File().openForWriting()

for each subsectionConfiguration in subsectionConfigurationList:

gCode = subsectionConfiguration.getGCode()

write gCode to unifiedGCodeFile

printJobConfiguration.setUnifiedGCode(unifiedGCodeFile)

# Physical Layer

The physical layer receives inbound G-Codes and printer configurations from the post processing layer. The physical layer will then assemble and serialize the data received in preparation to be sent to the communications layer. The G-Codes may be modified contingent on data received from the printer feedback and user interface layers. If either the printer feedback or user interface layers indicate that the print must be stopped then the physical layer must insert halt commands into the G-Code command stream such that the machine terminates the print in a timely manner.

## [Subsystem] Modules

[Description]

### [Module Name]

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS

#### Process

[Pseudo-code]

# Printer Feedback Layer

The printer feedback layer is responsible for receiving streamed input from the printer hardware, interpreting the data, then formatting and dispatching the information to the physical and user interface layers. The information received from the printer will mainly consist of printer state such as extruder temperature, arm position, and other operating parameters. The printer feedback layer will transform the data received from the printer into useful and readable data that can be passed to the user interface and physical layers.

## [Subsystem] Modules

[Description]

### [Module Name]

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS

#### Process

[Pseudo-code]

# Communication Layer

The purpose of the communication layer is to serve as an interface between the client software (The 3-D Printer Fabrication System) and the printer firmware itself. The communication layer shall be able to implement most RS-232 compliant serial communications medium such that data can be transferred from the client software to the printer firmware reliably. The communications layer serves both the printer feedback and physical layers. The physical layer represents outbound communication (from client software to printer) while the printer feedback layer represents inbound communication (from printer to client software).

## [Subsystem] Modules

[Description]

### [Module Name]

[Description]

#### Prologue

[Name/Description/Function]

#### Interfaces

|  |  |  |
| --- | --- | --- |
| Interface | Information Required | Information Returned |
|  |  |  |
|  |  |  |
|  |  |  |

#### External Data Dependencies

|  |  |
| --- | --- |
| Data | Source |
|  |  |
|  |  |

#### Internal Data Descriptors

[NOT EXACTLY SURE WHAT THIS IS

#### Process

[Pseudo-code]

# Quality Assurance

[Description of test plans and procedures]

## Unit Testing

### [Layer Name]

#### [Subsystem Name]

##### [Module Name]

[Description of tests on the module]

## Component Testing

[NOT SURE WHAT EXACTLY THIS IS YET]

## Integration Testing

[Description of integration testing]

## System Verification Testing

[Description of system verification testing (possibly take from SRS)]

## Test Cases

[Description]

|  |  |
| --- | --- |
| Test Case | Expected Result |
|  |  |
|  |  |

# 

# Requirements Traceability Matrix

[Description]

[Requirements traceability matrix (probably want to break this into layers)]

# Acceptance Plan

[Description/Overview]

## Package and Installation

[Probably use requirements from SRS]

## Acceptance Testing

[Description]

## Acceptance Criteria

[Reiterate acceptance criteria for requirements]

# Appendices