LAMP - Major Project 2018

Waxy Laser Solutions
Max Wharton-Jones, Shourov Quazi, Jack Jiang

August
2018

Laser
Acrylic
Major
Project

Contents

1 Defining the Problem **2**
1.1 Client Details . . . 2
1.1.1 Clients . . . 2
1.1.2 Current System . . . 2
1.2 Client Needs Research . . . 3
1.2.1 Interview . . . 3
1.3 Feasibility Study . . . 7
1.3.1 Market feasibility . . . 7
1.3.2 Technical feasibility . . . 7
1.3.3 AutoCAD/Illustrator Interoperability . . . 8
1.3.4 Utility and calibration . . . 8
1.3.5 Laser cutter machine . . . 8
1.3.6 School Systems . . . 9
1.3.7 Financial feasibility . . . 9
1.3.8 Operational feasibility . . . 10
1.3.9 Social and ethical feasibility . . . 11
1.3.10 Conclusion . . . 11
1.3.11 Overall Feasibility . . . 12
1.3.12 Possible Solutions . . . 12
1.4 Rights Research . . . 13
1.4.1 IP rights . . . 15
1.4.2 Contract . . . 15

2 Planning and Designing **18**
2.1 Context Data flow diagrams . . . 18
2.2 System Flowchart . . . 20
2.3 IPO Chart . . . 21
2.3.1 Select Template . . . 21
2.3.2 Design Template . . . 21
2.3.3 Queue Job . . . 22
2.3.4 Approve Job . . . 22
2.3.5 Set Up Laser Cutter . . . 23
2.4 Gantt Chart . . . 23
2.5 Data Dictionary . . . 24
2.6 Algorithms . . . 25
2.7 Screen Design Principals . . . 37

Chapter 1

Defining the Problem

1.1 Client Details[1]

1.1.1 Clients

Sydney Boys High School is an academically selective high school conducted by the NSW Department of Education. The school is led by the senior executive team, comprising the Principal, Dr Kim Jaggar, and Deputy Principals Ms Rachel Powell and Mr Robert Dowdell. The executive staff of nine Head Teachers and the twelve School Administration Officers led by Senior Administrative Manager Ms Sharon Kearns, support the senior executive. The school has three main offices - in the Main Building, in the Ken Andrews Library and in McDonald Wing. Finance, purchasing, enrolment and general inquiries are handled in the main building. Secretarial and network services are the responsibility of the McDonald Wing office. The clients for this project will be Ms Dam, Mr Comben and Dr Jaggar. Ms Dam and Mr Comben moderate the use of the laser cutter machine and are both teachers in the Industrial Arts department with Ms Dam being the head teacher. They are seeking a better system regarding the use of the laser cutter, especially with cutting trophies for their respective sports. Dr Jaggar is the principal of SBHS, who will be financing the project. Additional users include all other staff at SBHS and all current students of SBHS, however only Staff that are MIC of sports and other extracurricular activities will have access to the creation of trophies. Other staff and students will have access to the template creator only.

1.1.2 Current System

In the current system, all laser cutting requests are handled by the IA staff over email or in person. Individuals send their design files into to the IA staff or go in person to cut their objects with supervision. Only limited number of students and teachers have access to the system, and each job must be signed off by someone from the IA staff. Before laser cutting, the file must be checked manually by staff to ensure for correctness, and an often trial-and-error approach is used to ensure the correct settings are applied to each line type. Material is then aligned in the laser cutter, and the piece is cut, a process that can take between 5 minutes to several hours, depending on the complexity and size of the job. Only one job can be cut at once. There is no tracking of jobs, relying on email and paper to track jobs in progress and in queue. Currently, they use a program to help generate one type of template, the school trophy, although this program lacks several features, explored in the interview.

[1] Client Details by Jack

1.2 Client Needs Research[2]

1.2.1 Interview

Interview with Mr Comben (MIC, IA Teacher)

Q: How many physical Awards are given out each year
A: The number varies through the years, but for rifle shooting in particular, there are mainly perpetual awards. For these, we would be looking for brass laser cut plaques.

Q: How much area in the budget is there for extra trophies?
A: Keeping in mind of the cost factor, including the cost for a teacher to operate the laser cutter, as well as the material, there is most likely not that much money available for many of the sports. I know rifle shooting has no area in the budget for extra trophies.

Q: How can we make the laser cutting process more efficient?
A: Look for vector-based fonts, this would allow faster cutting. You want faster cutting speeds to minimise the time spent waiting at the cutter. Also, try to make a reference-based system, this way the amount of time spent setting up the cutting process.

Q: What would be some improvements from the previous system that you would recommend?
A: The old system was very good, although due to the time constraint had many flaws. I would recommend an easier GUI. The old GUI was hard to navigate, and I believe there were some areas that were not functional. Also, the system of setting up the laser cutter took up a large bulk of time. If you can find a way to align the cutter easier, that would be good. Also, the workflow from AutoCAD is very unreliable, I would recommend exporting in illustrator

Q: What were some advantages in terms of resources from the previous system?
A: I can't say for sure about the budgetary resources saved, but I can say for sure that in the long term, the money saved would easily pay off the laser cutter. Also, the program has inspired a great deal of the industrial tech classes, we now have year 8's playing around with the potential of the cutter which is great.

Q: Do you have anything you want to see from our program?
A: I would like to see a function that would allow the user to nominate a folder of files that are ready to laser cut, and give the user detailed feedback on the user. Such as, this user has 90% black line in his work, and would take a long time to finish. This would allow us to check the jobs, and let it be easier to pass works.

Interview with Ms Dam. (MIC, Supervisor, Head of Industrial Arts)

Q: What are the costings of using the laser cutter in reference to the program used last year to create the trophies?
A: Last year, the trophies printed was a great success, especially considering the fact that it was the first year to use the laser cutter. The cost was of course the cost of the trophies from its original source. The profit came from the comparison of the cost of a teacher to be printing

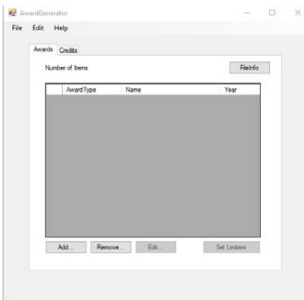
[2] Interview by Shourov

and the cost of printing the trophies elsewhere. The cost of a teacher is around $400 a day whereas each trophy would cost $10 elsewhere. Hence, if we could print 40 trophies a day, we would be making a profit, which we easily reached.

Q: What would you want to see from our program?
A: From your program I'd like to see it be a lot less time consuming. I would like to spend less time setting up the cutter and more time watching it cut. This would be done as Mr Comben said, to use a reference point system. Also, I would like to see a cleaner GUI.

Needs

Needs	Objectives
Must store > 50 different records	• Store > 50 different templates in a database • Store > 50 users, clients • Store fonts and different shapes
An editor to create templates	A drag and drop interface supporting • text with different fonts • circles • rectangles • lines
Use a variety of different materials	Program must indicate the material and settings to use on the laser cutter OR setting these values automatically before a job.
Be cost effective (manpower, trophy cost)	The program must be efficient in the usage and process of its materials. For example, when engraving school trophies for Sydney Boys High School. Assuming that: • Each trophy laser cut saves $10 • A working day is 5 hours long. • The cost of a Teacher is $400 a day Then, the program must be optimised such at LEAST 8 trophies can be cut per hour.
Minimum Manual work to save time	Program must assist the user in aligning the job in the laser cutter. Previously this was done manually with callipers.

Improvement on printing capabilities of the last product	To increase efficiency of the cutter: • Reduce Raster lines • Increase Vector Lines • No double lines • Vector-based fonts
Improved GUI usability	Current GUI has many issues, that must be rectified in the solution • Hard to navigate • Limited Usage • No tutorial • Some functions don't work like keybindings • Hard to line up 
Export/Import to Illustrator and Autocad	Ability to export/import files compatible with illustrator Illustrator AND/OR AutoCAD or other popular cad programs
Check Illustrator files for efficiency	A process that reads in an Illustrator file and • Reads all linetypes in the file • Compiles linetypes in numerical data • Quantitatively assesses linetype data • Provides estimate on print time • Displays linetype data in readable format

Ability to search and sort templates	Ability to attach a number of tags to each template, which can be filtered by the user • Template name • Template creator • Date created/Approved • Item ID • Template material • Purpose, e.g. athletics, rifle shooting
Different levels of access	3 levels of access, with each level having all the permissions of the levels below • Student (Lowest): can submit templates to be approved then used by teachers/administrators • Teachers: can also submit jobs to administrators for approval, who run the laser cutter. Can also approve templates. • Administrators: have all permissions, can view all information and approve all jobs/templates, to be cut by the laser cutter.

Specifications

The school computers are mostly Dell ThinkCentre machines, with the following specifications

CPU	Intel® Pentium® CPU G3220 @ 3.00 (GHz)
RAM	8192MB (8GB)
Graphics Adapter	Intel® HD Graphics
Operating System	Windows 10 Pro

Therefore, program requirements should include:

	Minimum Requirements	**Recommended Requirements**
CPU	Dual Core 2 GHz+	Quad Core 3 GHz+
RAM	512 MB	4GB
Operating System (Client)	Windows 7	Windows 7, 8, 10
Dependencies (Client)	NET Framework 4.0	NET Framework 4.0
Operating System (Servier)	Windows 7, MacOSX 4.0, Ubuntu 16.04+	Windows 7+, Ubuntu 16.04+

- We chose to limit the Client Program to the Windows operating system, all the school computers run windows exclusively. However, the server software must be cross platform, as servers are often run from a variety of OS's (Linux, windows etc.)
- The program should run well on school computers, as they meet both the minimum and recommended requirements.

1.3 Feasibility Study [3]

1.3.1 Market feasibility

The proposed plan is to design a software product that will use the laser cutting machine to produce physical products. The project involves several industries: software design, industrial production / manufacturing and product design.

The low volume and DIY (Do it yourself) manufacturing market has flourished, due to the higher availability and the diminishing costs associated with manufacturing. Flexible Manufacturing Systems (FMS), like the laser cutter or 3D printer allow for arbitrary objects to be created from computer-aided designer (CAD) files.

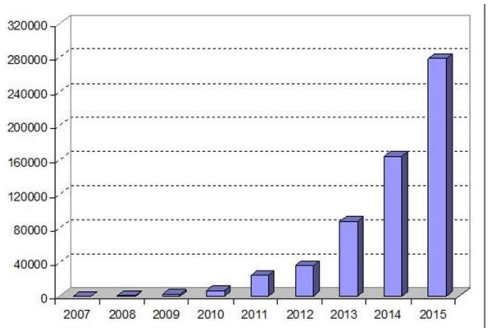


Figure 1.1: Desktop 3D printers sold. Source: Wohlers Report 2016

Currently, solutions on the marketplace include: 2D CAD programs, which although work with the laser cutter, becomes inconvenient and inefficient when producing a large quantity of objects with variations, and complex templating solutions which require a significant amount of programming and technical expertise. The project involves using a program to aid in the generation and production of files that are sent to the laser cutter. There are few or no off-the-shelf type programs that achieve this, as laser cutting is often a specialised and expensive industrial process. It is often cumbersome for students and teachers to create multiple copies of a single object, which the program seeks to automate Conclusion: LAMP focuses on the niche in the market. There are few commercial programs that help with low volume production, a market that has boomed in the last five years. Since the demand for such a software is increasing and there are few competitors, this program will have a well-defined target market, and is market feasible.

1.3.2 Technical feasibility

There are many technical components that need to be researched and experimented on beforehand to ensure that the project's success. Fortunately, the laser cutting project developed last year can be used as a proof of concept, with many technical barriers solved which can be ported over to the new program (existing software). Designer Technology: LAMP will include a designer that allows editing of templates dynamically inside the program. It will essentially be a 2D only CAD screen, with options for lines, circles, boxes, shapes and text. Text may be in different sizes and fonts. The designer will allow the three different cutting types to be specified for individual elements (vector cut, vector engrave, raster engrave). This will then be saved in a custom format (.spiff file), which contains all the data required to store the required

[3] Feasibility Study by Jack

template. Placeholder elements can be specified in the .spiff file, so that the template can be filled in automatically through the program with a list of names, or years.

- Basic 2D CAD interface
  - rendering of elements onto screen
  - optimization to run on lower-powered computers
  - real dimensions (cm/mm)
  - zoom
  - different modes (cutting, engraving)
- custom file type
  - .SPF file contains data on placeholder elements (text that needs to be replaced) and also line/box data
  - read and write .SPF files

1.3.3 AutoCAD/Illustrator Interoperability

The .SPF file will need to be exported to illustrator (.ai) and/or AutocAD (.dxf) files to be used by the laser cutter. The program will also need to be able to take a list of names/years from a document file, using this data to replace the placeholder elements on a template, and layout the template a variable amount of time in the output file in an optimum matter, taking into account the total space in the laser cutter. Manual alignment will also be possible.

- export to vector format (ai/dxf)
  - writing/reading design files
  - generating different vector lines
  - vector fonts
  - vector lines and curves
- read/write document files (.docx, .xlsx, .csv)
- layout of multiple copies of template

1.3.4 Utility and calibration

The program will have error-checking algorithms on .AI files, to ensure it has lines that are compatible with the laser-cutter.

- read/write illustrator files
  - file checking algorithm for illustrator files

1.3.5 Laser cutter machine

The laser cutter machine is a complex piece of hardware, capable of cutting thin material via vector lines, and engraving with both vector and raster modes available. The laser bed or cutting space is 450 x 600mm, which will limit the maximum number of trophies cut at once. It uses a 50 watt, 10—TODO—m laser, and the materials it can cut are thin woods and plastics, but it can engrave a variety of materials, including plastics, woods and glass.

- Capabilities of the laser cutter
  - engraves plastics, woods and glass
  - cuts thin woods and plastics
- Size of the laser cutter (450 x 600mm)
- Safety of operation

1.3.6 School Systems

LAMP stores the approved templates which are available to anyone using the program and the jobs in queue in a server. This may be run locally on the school server, or in a cloud server hosted on SaaS platforms. The files and credentials of users will need to be encrypted, to increase the safety of the system, and some basic measures need to be taken to stop hacking or denial of service (DOS) attempts. This server will come as a separate executable to the client system used by the end user. The client software will need to be able to run on school computers, which mostly have dual-core intel processors with between 8 to 16 gigabytes of ram. Approval from the school's IT staff may be required to install LAMP's client software.

- Server Software (if required)
  - file and credential encryption
  - serve requests to list all approved templates
  - unblocked from school internet
  - needs to be secure and reliable
- Client Software
  - may need administrator permissions to install on school systems (ask IT).
  - will connect to the server through the internet, or the schools internal intranet.
  - optimization to run on school's computers

Conclusion

There are a large number of technical challenges to solve in order for lamp to succeed. Fortunately, several of these have already been addressed in the previous program, and the Industrial Arts staff at school understand the laser cutting system, providing enough information to explain many of the laser-cutting related problems. Over the holidays and throughout the year, research will need to be done on the schools systems, the designer interface and our custom file type. Thanks to the previous program and our teams previous experience with working on the laser cutter, we will be able to focus on these issues instead, reducing the amount of work needed. Overall, the program will be technically feasible.

1.3.7 Financial feasibility

A middle-end laser cutting machine is an expensive instrument, coming around between $20,000 - $100,000. There are cheaper alternatives, but they are slower and/or less precise. However, as the school already has bought a laser cutter, the costs are mostly maintenance and teacher time. The original trophy system can be used as a proof of concept, and has proven to be cost effective. Expanding this system to other awards may allow for even more cost saving. Awards for different sports account for a significant savings. Costs can be further reduced

by bulk buying many trophies from overseas. Take for example the previous program, which focused on a particular trophy, the School Trophy. Initially it had cost the school 135$ per trophy, including raw material and engraving costs. However, a blank trophy can be sourced for 15$, and engraved on the school's cutter. Given that the hourly rate for teachers is 80$ per hour, with each school trophy taking approximately 15 minutes of time, and producing a trophy in-house would costs 35$, or a 75% decrease in cost. LAMP will decrease the time required to setup and cut the trophy, and also allow for other types of trophies to be cut through its templating system, which the old system could not do.

We will require some material and test awards to experiment with the abilities of the laser cutter, which needs to be accounted into our budget. Other costs may include licensing libraries, distribution costs and/or server maintenance. However, even with all these costs factored in, the in-house production of trophies along with other objects will still be cost-effective, saving the school thousands of dollars per annum.

The system will have some server setup and maintenance costs - however, this will be low, as it can be hosted on the school's existing servers or through inexpensive cloud providers. The program will not need much processing power, and will not handle a large amount of data, further reducing the server costs. Other setup costs include install time and storage space for LAMP's client software. A developer may be hired to continue to maintain the program after its release, and to fix any bugs discovered after.

LAMP will require a significant amount of developer time, and falling into the medium range in terms of software, probably costing between $20,000 - $60,000, based off several other custom software projects. Developer time costs between 75$ to 200$ per hour, and the project overall will take around 200-300 hours. The software will be licensed to the school, and may be licensed to multiple schools and businesses. A fee will be charged per user per year, with business subscriptions including priority tech support and user management features. There will be 3 tiers: individual licence will be between 10-30$ per year for 1 individual, small business (10-1000 users) for 20-40$ per year per user, and large businesses negotiated separately. Small and large business are also given a copy of the server software that can be setup on their own servers to serve their organisation's users. This copy will be completely separate from the systems of other businesses, allowing queued jobs and credentials to be kept separately.

Conclusion

The project's main expenses are developer time, and will cost approximately $40,000. This will be recouped by licensing the software to multiple business and individuals, and charging a yearly fee, which will also help pay the maintenance developers. For the business, LAMP will decrease operational costs by automating parts of the laser cutting process. Therefore, LAMP is financially feasible.

1.3.8 Operational feasibility

Users

Users may include teachers and some select students. Teachers will be able to use the program to quickly and easily submit a set of awards for some students. The process should not be changed significantly: the list of students will be sent to the program instead of emailed to an awards seller. The user interface will need to be reliable, consistent and uncomplicated, especially the template designer, which will allow both students and teachers to create the shapes and text without beforehand knowledge of the intricacies of CAD programs and the laser cutter. Little training will be required to use the system for users - pick a template, give some names, submit

job. This information will be provided by video tutorials, reference manuals and online help. On-call tech support may be available for business users. Users are able to design templates and/or submit jobs, depending on the permissions given to the user by an administrator.

Administrators

Administrators/IA staff will manage and approve request from the users. This will take a significant amount of time, that would otherwise be spent on teaching. To reduce the impact this has, the program will attempt to automate many of the intermediary steps in setting up the laser cutter, and allow for less time calibrating the machine, a process that takes 10-20 minutes each set of trophies. It will also use more efficient processes to reduce the production time. The program will first be tested on only the industrial arts staff, to ensure the time required to process these trophies will not be an overwhelming amount of work. Administrators have the highest level of access to the program, with permissions to approve both submitted jobs and submitted templates, create and manage users and other administrators, reset passwords etc. This will require a significant amount of training, through video tutorials, reference manuals and online help, with tech support available for businesses. An administrator will require experience with the laser cutter, as they must physically set up the material on the laser cutter in accordance to the current job.

Conclusion

For users, the program will require very little training. This means that it will be easy to introduce new users to the system. Administrators will require significant training; however, the industrial arts staff are already accustomed to the laser cutter, easing this process. Support will be given in the form of video tutorials, reference manuals, online support, with on-call technical support for businesses. Overall, this system will not require much training.

1.3.9 Social and ethical feasibility

The program will handle some sensitive data from users - their full name, email, passwords, secret questions and/or contact information will be required by the program in order to operate. This will be stored on the client's computers, and on LAMP's server software if applicable. Security will need to be kept in mind to prevent unauthorized access to this sensitive data, through extra security measures taken on the client's computers, e.g. anti-virus software, correct user privileges, and in the program, e.g. encryption of database, authentication through passwords. Additional server software provides another vector for attackers, and appropriate security measures, like https will need to be used to ensure communications between the program and the server are safe, and to prevent access from unauthorized users. Care will need to be taken to ensure the privacy of the information given to the program by preventing unauthorized access. The program should be as inclusive as possible: this would require the program work on lower-powered machines, have an easy and under stable user interface, and users with disabilities considered. Copyright and intellectual property is another possible issue, with appropriate credit given and/or licenses obtained from code included from another source. The program may reduce the amount of time spent laser cutting, but since laser cutting is a secondary job to teaching at Sydney Boys High School, its effect will be negligible on the workforce.

1.3.10 Conclusion

The privacy of users will need to be considered when developing the solution. Appropriate measures need to be taken to prevent unauthorized access. The software should be inclusive to

all users, by altering the user interface to suit the needs of individuals. Intellectual property rights of others need to be considered in the program.

1.3.11 Overall Feasibility

the issues mentioned with this feasibility study will need to be resolved, but all essential components of the program will be met in the time period set in the Gantt chart

1.3.12 Possible Solutions

Aim: To better use the laser cutting system by simplifying the process required to design and cut objects, and allowing teachers and students limited access into the system.

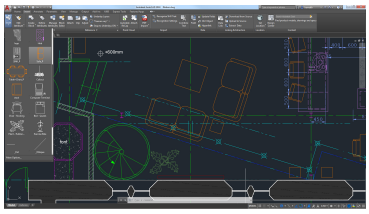
1. "Upgrade" Approach

This approach would see the original trophy generation revisited and revamped to fix issues outlined by staff. In addition, new features could be added, like support for different shapes or awards, and improvements could be made to the GUI to increase its useabilility. This program would only be accessible and usable by IA staff to generate shapes, which can then be rendered by Illustrator a format the laser cutter can use.

- Low cost, complexity and maintance required
- Code reuse, which decreases development time and cost
- Similar to system already in use, reduceds training required

2. "Designer" Approach

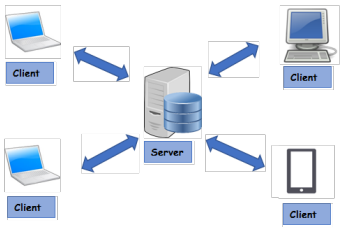
This approach requires a custom piece of software to be created, and is similar to the system already in use in the school. Using a program to generate a known-good file will allow IA staff to skip several steps in the laser cutting process. However, this incurs additional cost, both in development/setup time and cost. Some basic training and documentation will also need to be provided to the industrial arts teachers using the program. This solution does not attempt to track the created files, relying on email or another form of communication to send and receive files.



- Design tool that loads much faster than AutoCAD and uses less resources
- Specialised user interface that contains only the tools required for laser cutting templates
- File checking features to ensure linetypes are correct
- Automates some printing settings to the laser cutter

3. "Server-Client" Approach

This approach uses a server to store templates and jobs that users can edit and use. Using a server allows easier communication between the end-users and the IA staff, but incurs additional cost and setup complexity. Using a server also entails an additional, recurring cost of hosting the server. Security over the net could also be an issue, and care will need to be taken to avoid access by unauthorized individuals over the internet. On the other hand, the server-client approach will allow for better tracking of individual jobs, centralization of users to ensure only the correct people have access to the program through user login and server-side checking of files. This system also may allow users/administrators to use the system from multiple places, as long as there is a working internet connection Features:



- Loads much faster than AutoCAD and uses less resources
- Specialised user interface that contains only the tools required for laser cutting templates
- File checking features to ensure linetypes are correct
- Automates some printing settings to the laser cutter
- Templates on server can be accessed from anywhere with an internet connection
- Tracking of jobs that are in queue or complete
- User and Administrator management

The client has chosen option 3 The client would like to choose Option 3. Option 1 is far too basic and not really an the current system. Option 2 again is not an improvement on the current system, as we already use a server. Option 3 contains many new features that could help improve the efficiency of laser cutting, despite the extra cost.

1.4 Rights Research [4]

A software licence determines the use and redistribution of software. It determines how the software can be used by the purchaser of the software, often called the licensee, and may protect the developer legally from damage caused by the software.

[4] Section by Max

There are several types of licences:

- Public domain
- Open source (FOSS) licenses
- Freeware / Shareware
- Proprietary

Software licenses and rights granted in context of the copyright according to Mark Webbink.[1] Expanded by freeware and sublicensing.

Rights granted	Public domain	Permissive FOSS license (e.g. BSD license)	Copyleft FOSS license (e.g. GPL)	Freeware/Shareware/ Freemium	Proprietary license	Trade secret
Copyright retained	No	Yes	Yes	Yes	Yes	Yes
Right to perform	Yes	Yes	Yes	Yes	Yes	No
Right to display	Yes	Yes	Yes	Yes	Yes	No
Right to copy	Yes	Yes	Yes	Often	No	No
Right to modify	Yes	Yes	Yes	No	No	No
Right to distribute	Yes	Yes, under same license	Yes, under same license	Often	No	No
Right to sublicense	Yes	Yes	No	No	No	No
Example software	SQLite, ImageJ	Apache web server, ToyBox	Linux kernel, GIMP	Irfanview, Winamp	Windows, Half-Life 2	Server-side World of Warcraft

Open source licenses, like the GNU GPLv3 licence are for collaborative projects, where developers create code, often for free for their own use. Any developer can download and alter the source code of a GPL project, but they must provide the altered source code to end-users for their derivative work, display a notice on the program, crediting the original developers of the source code and license their work under the GPL. Many open source projects use this licence, as it ensures that their work will be credited and improvements to the software carried out will be made free to the public. The MIT licence is another software licence. It is a short, simple licence, that allows the alteration of source code with no other conditions. Both these open source licences disclaim any warranties or responsibilities of the original developer in the quality and usability of the code.

Copyright <2017-2018> <Max Wharton-Jones>
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The MIT Licence

Freeware licences may use ads or donations in order to make a profit. However, it often lacks enterprise support. Shareware uses locked features or a trial period, allowing users to try out the software before committing to a purchase.

Open source licences are unsuitable for our project, as they require the distribution of source code. In addition, there is no need for an open source licence as the project will only be created and maintained by our team. A freeware / Shareware licence is unsuitable, as our program will

mostly target large organisations, who are willing to pay extra in return for support. Therefore, the proprietary licence will be the most suitable. The software will be maintained by our team, allowing for more features and bug fixes when discovered, funded by the licensing fee. The safety, reliability and usability of the program is essential, as it involves the laser cutter, an expensive and potentially dangerous machine.

1.4.1 IP rights

Waxy LASER Solutions retains all intellectual property rights to the software. This is necessary so that the program can be licensed to other businesses, and to allow the program to be maintained by our team in the future.

1.4.2 Contract

L.A.M.P - Terms and conditions - Waxy LASER Solutions

1. **Preamble:** This Agreement, signed on Dec 6, 2017 (hereinafter: Effective Date) governs the relationship between Sydney Boys High School, a School Entity, (hereinafter: Licensee) and L.A.M.P, a partnership under the laws of whose principal place of School is 556 Cleveland St, Moore Park NSW 2021 (hereinafter: Licensor). This Agreement sets the terms, rights, restrictions and obligations on using L.A.M.P (hereinafter: The Software) created and owned by Licensor, as detailed herein

2. **License Grant:** Licensor hereby grants Licensee a Personal, non-assignable and non-transferable, commercial, royalty free, non-exclusive license, all with accordance with the terms set forth and other legal restrictions set forth in 3rd party software used while running Software.

1. Limited: Licensee may use Software for the purpose of:
   (a) Running Software on Licensee's Website[s] and Server[s];
   (b) Allowing 3rd Parties to run Software on Licensee's Website[s] and Server[s];
   (c) Publishing Software output to Licensee and 3rd Parties;
   (d) Distribute verbatim copies of Software's output (including compiled binaries);
2. This license is granted perpetually, as long as it is not materially breached.
3. **Binary Restricted:** Licensee may sublicense Software as a part of a larger work containing more than Software, distributed solely in Object or Binary form under a personal, non-sublicensable, limited license. Such redistribution shall be limited to 1600 codebases.
4. **Non-Assignable and Non-Transferable:** Licensee may not assign or transfer his rights and duties under this license.
5. **Commercial, Royalty Free:** Licensee may use Software for any purpose, including paid-services, without any royalties

3. **Term & Termination:** The Term of this license shall be until terminated. Licensor may terminate this Agreement, including Licensee's license in the case where Licensee:

1. became insolvent or otherwise entered into any liquidation process; or
2. exported The Software to any jurisdiction where licensor may not enforce his rights under this agreements in; or
3. Licensee was in breach of any of this license's terms and conditions and such breach was not cured, immediately upon notification; or

4. Licensee in breach of any of the terms of clause 2 to this license; or
5. Licensee otherwise entered into any arrangement which caused Licensor to be unable to enforce his rights under this License.

---

4. **Payment:** In consideration of the License granted under clause 2, Licensee shall pay Licensor a fee, via Credit-Card, PayPal or any other mean which Licensor may deem adequate. Failure to perform payment shall construe as material breach of this Agreement.

---

5. **Upgrades, Updates and Fixes:** Licensor may provide Licensee, from time to time, with Upgrades, Updates or Fixes, as detailed herein and according to his sole discretion. Licensee hereby warrants to keep The Software up-to-date and install all relevant updates and fixes, and may, at his sole discretion, purchase upgrades, according to the rates set by Licensor. Licensor shall provide any update or Fix free of charge; however, nothing in this Agreement shall require Licensor to provide Updates or Fixes.

1. Upgrades: for the purpose of this license, an Upgrade shall be a material amendment in The Software, which contains new features and or major performance improvements and shall be marked as a new version number. For example, should Licensee purchase The Software under version 1.X.X, an upgrade shall commence under number 2.0.0.
2. Updates: for the purpose of this license, an update shall be a minor amendment in The Software, which may contain new features or minor improvements and shall be marked as a new sub-version number. For example, should Licensee purchase The Software under version 1.1.X, an upgrade shall commence under number 1.2.0.
3. Fix: for the purpose of this license, a fix shall be a minor amendment in The Software, intended to remove bugs or alter minor features which impair the The Software's functionality. A fix shall be marked as a new sub-sub-version number. For example, should Licensee purchase Software under version 1.1.1, a fix shall commence under number 1.1.2.

---

6. **Support:** Software is provided under an AS-IS basis and without any guarantees of updates or maintenance. Nothing in this Agreement shall require Licensor to provide Licensee with fixes to any bug, failure, mis-performance or other defect in The Software.

1. Bug Notification: Licensee may provide Licensor of details regarding any bug, defect or failure in The Software promptly and with no delay from such event; Licensee shall comply with Licensor's request for information regarding bugs, defects or failures and furnish him with information, screenshots and try to reproduce such bugs, defects or failures.
2. Feature Request: Licensee may request additional features in Software, provided, however, that
   (a) Licensee shall waive any claim or right in such feature should feature be developed by Licensor;
   (b) Licensee shall be prohibited from developing the feature, or disclose such feature request, or feature, to any 3rd party directly competing with Licensor or any 3rd party which may be, following the development of such feature, in direct competition with Licensor;
   (c) Licensee warrants that feature does not infringe any 3rd party patent, trademark, trade-secret or any other intellectual property right; and
   (d) Licensee developed, envisioned or created the feature solely by himself.

---

7. **Liability:** To the extent permitted under Law, The Software is provided under an AS-IS basis. Licensor shall never, and without any limit, be liable for any damage, cost, expense or any other payment incurred by Licensee as a result of Software's actions, failure, bugs and/or any other interaction between The Software and Licensee's end-equipment, computers, other software or any 3rd party, end-equipment, computer or services. Moreover, Licensor shall never be liable for any defect in source code written by Licensee when relying on The Software or using The Software's source code.

8. **Warranty:**

   1. Intellectual Property: Licensor hereby warrants that the Software does not violate or infringe any 3rd party claims in regards to intellectual property, patents and/or trademarks and that to the best of its knowledge no legal action has been taken against it for any infringement or violation of any 3rd party intellectual property rights.
   2. No-Warranty: The Software is provided without any warranty; Licensor hereby disclaims any warranty that The Software shall be error free, without defects or code which may cause damage to Licensee's computers or to Licensee, and that Software shall be functional. Licensee shall be solely liable to any damage, defect or loss incurred as a result of operating software and undertake the risks contained in running The Software on Licensee's Server[s] and Website[s].
   3. Prior Inspection: Licensee hereby states that he inspected The Software thoroughly and found it satisfactory and adequate to his needs, that it does not interfere with his regular operation and that it does meet the standards and scope of his computer systems and architecture. Licensee found that The Software interacts with his development, website and server environment and that it does not infringe any of End User License Agreement of any software Licensee may use in performing his services. Licensee hereby waives any claims regarding The Software's incompatibility, performance, results and features, and warrants that he inspected the The Software.

9. **No Refunds:** Licensee warrants that he inspected The Software according to clause 7 and that it is adequate to his needs. Accordingly, as The Software is intangible goods, Licensee shall not be, ever, entitled to any refund, rebate, compensation or restitution for any reason whatsoever, even if The Software contains material flaws.

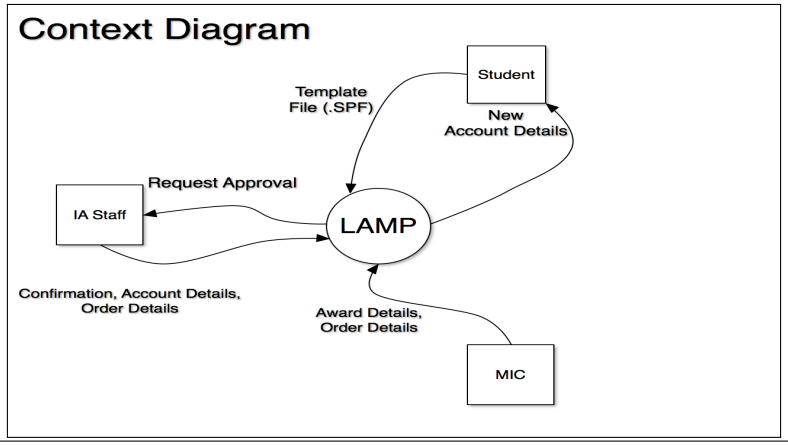
10. **Indemnification:** Licensee hereby warrants to hold Licensor harmless and indemnify Licensor for any lawsuit brought against it in regards to Licensee's use of The Software in means that violate, breach or otherwise circumvent this license, Licensor's intellectual property rights or Licensor's title in The Software. Licensor shall promptly notify Licensee in case of such legal action and request Licensee's consent prior to any settlement in relation to such lawsuit or claim.

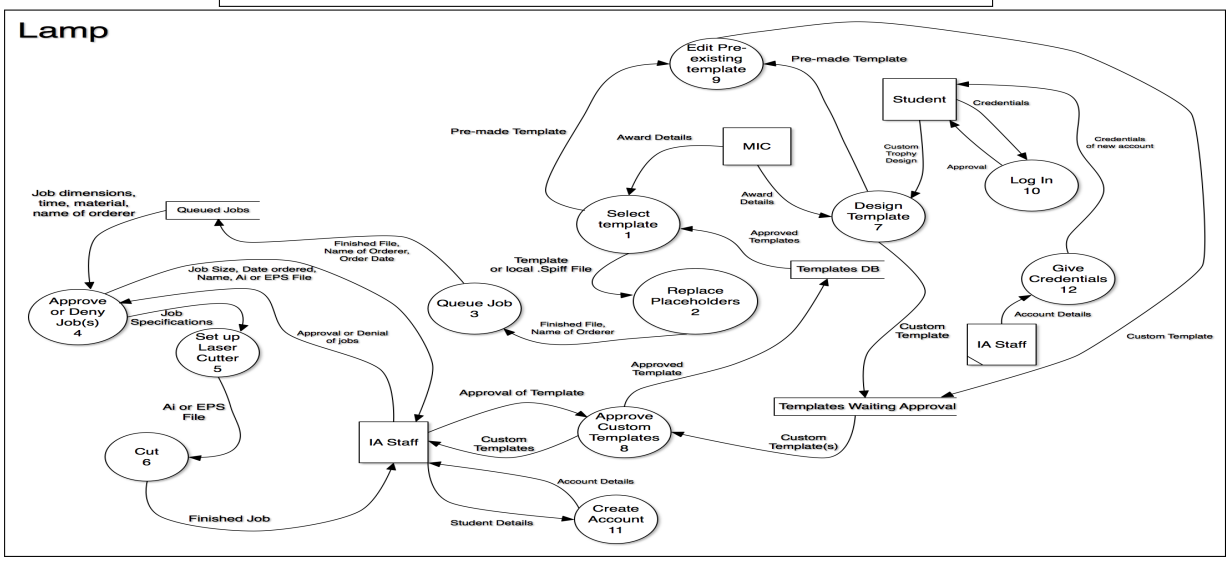
11. **Governing Law, Jurisdiction:** Licensee hereby agrees not to initiate class-action lawsuits against Licensor in relation to this license and to compensate Licensor for any legal fees, cost or attorney fees should any claim brought by Licensee against Licensor be denied, in part or in full

Chapter 2

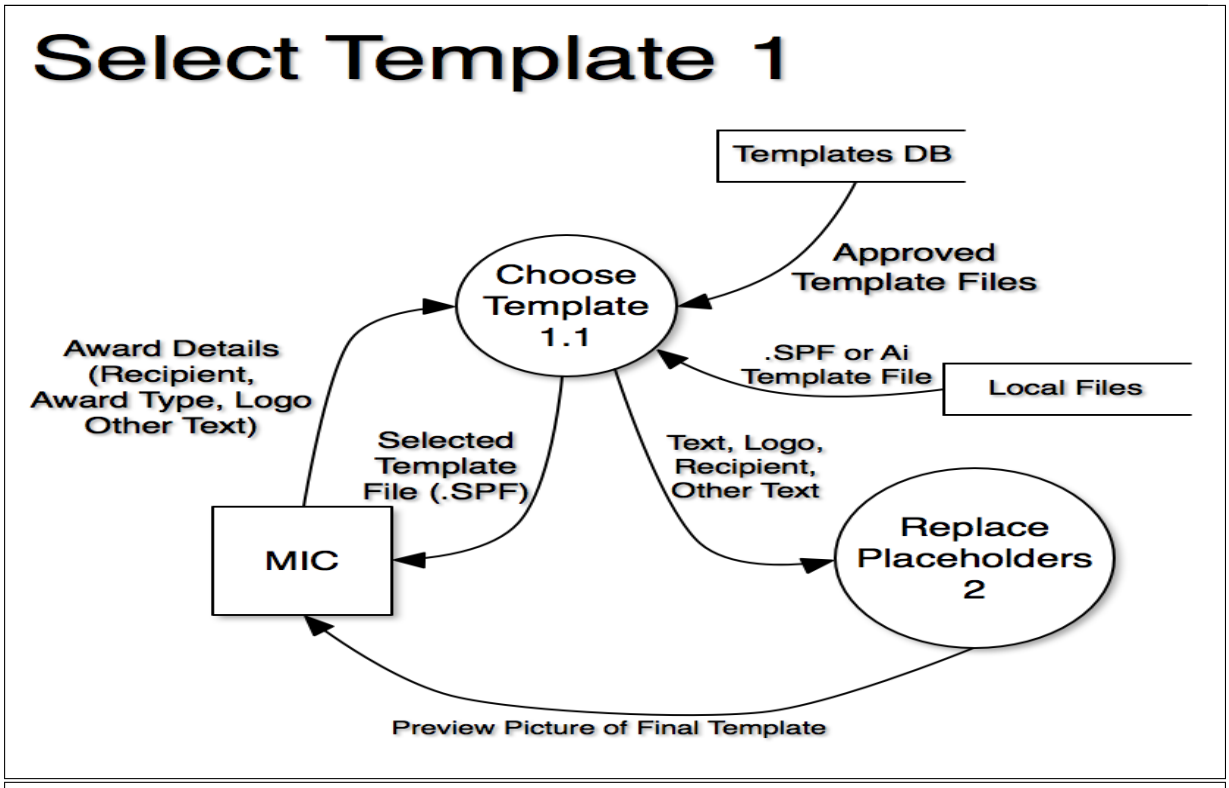
Planning and Designing

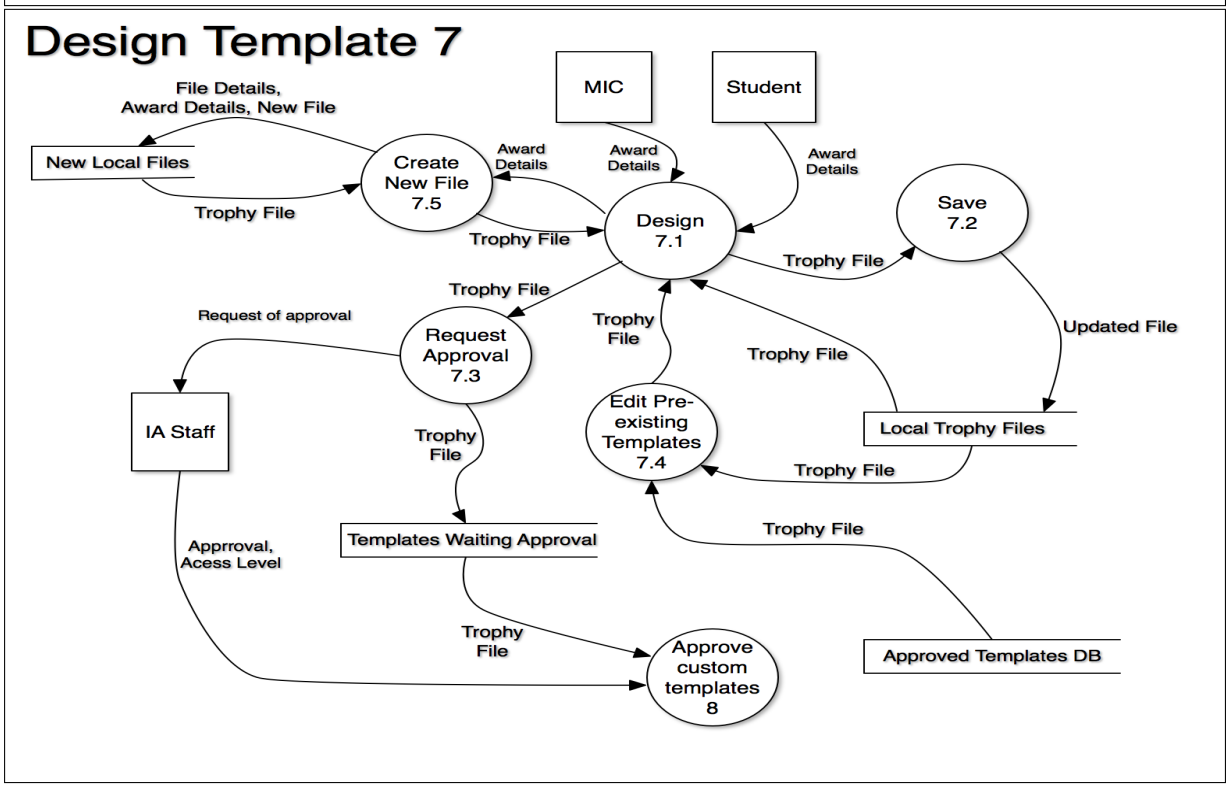
2.1 Context Diagram and Data flow diagrams [1]



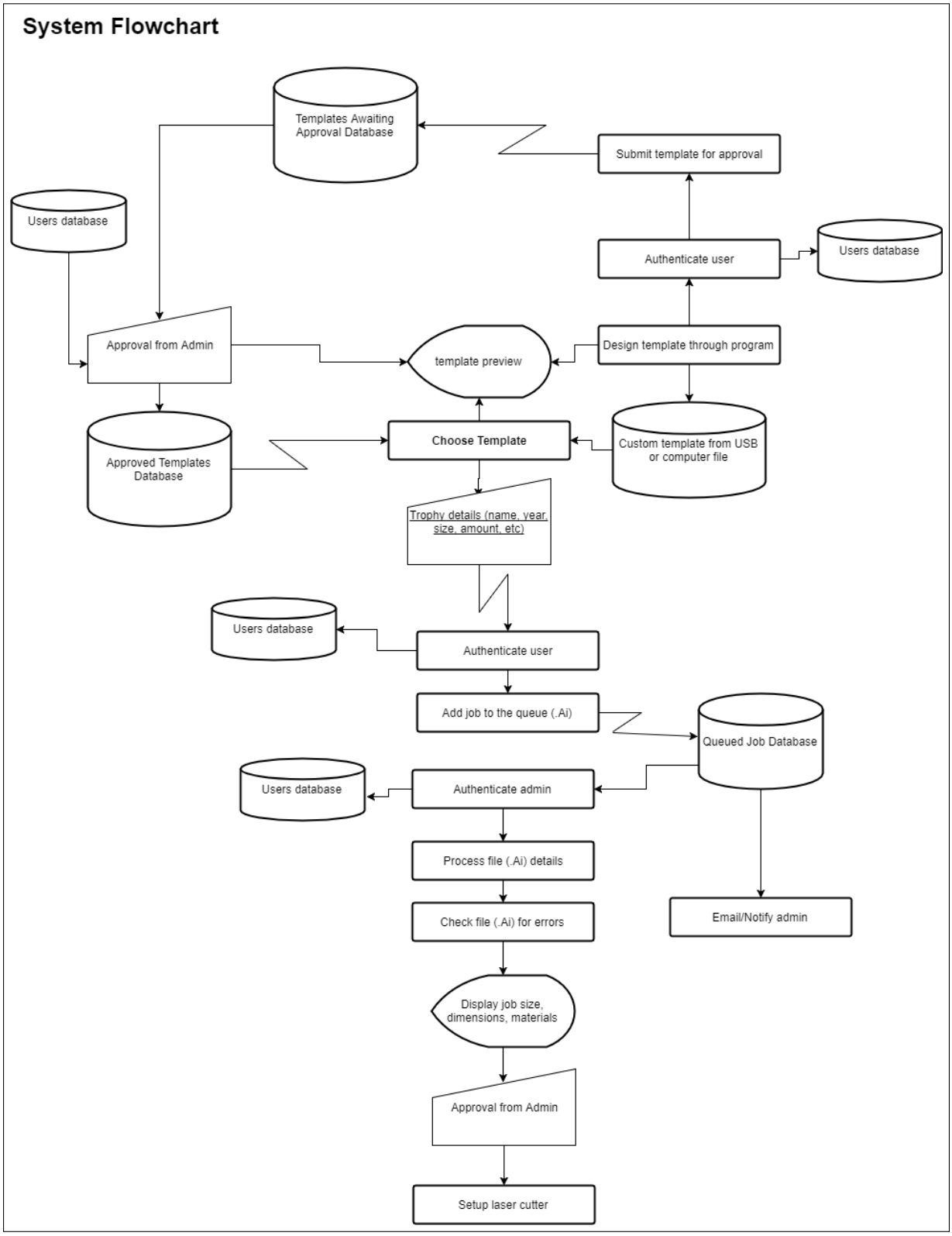


[1] Context and DFD by Max


Select Template 1
Templates DB
Approved Template Files
Choose Template 1.1
Award Details (Recipient, Award Type, Logo Other Text)
.SPF or Ai Template File
Local Files
Selected Template File (.SPF)
Text, Logo, Recipient, Other Text
MIC
Replace Placeholders 2
Preview Picture of Final Template


Design Template 7
File Details, Award Details, New File
MIC
Student
New Local Files
Create New File 7.5
Award Details
Award Details
Award Details
Trophy File
Trophy File
Design 7.1
Save 7.2
Trophy File
Trophy File
Request of approval
Request Approval 7.3
Trophy File
Trophy File
Updated File
IA Staff
Edit Pre-existing Templates 7.4
Local Trophy Files
Trophy File
Trophy File
Templates Waiting Approval
Apprroval, Acess Level
Trophy File
Approve custom templates 8
Approved Templates DB

2.2 System Flowchart [2]



[2] System Flowchart By Jack

2.3 IPO Chart [3]

2.3.1 Select Template

Input	Process	Output
Array of SPF files	Obtains Template From Database Displays all template designs in the template database Client selects appropriate template from the database	Partial SPF File containing data for the template selected
Names (string) **Number of Awards (int)** **Args*(vary)**	Adds Data to the SPF file containing the type of template already chosen * *based on the template chosen, the user will be prompted for certain data. An example is for a trophy that requires both a name and a score. The user will be prompted to enter the name as well as the score. If left blank, the input will be considered invalid unless otherwise specified.*	Complete SPF File containing both the selected template data as well as the variation data specified by the user

2.3.2 Design Template

Input	Process	Output
Array of SPF files	Creates Template from Selection Selection From Client • Presented as a selection GUI • Displays all template designs in the template data base • Client selects appropriate template from the database	Partial SPF File containing data for the template selected

[3] IPO Chart by Shourov

Raw AI / EPS Template File	Edits Template with Graphical Editor Graphical Editor • Editor Based off of CAD software • Tools include Line creation • Ability to add images • Ability to add dynamic text [4] • Ability to add static text [5] • Ability to create CUT lines and ENGRAVE lines	AI / EPS Template File
Login / Password (string)	Authenticates User	Authentication Level (Admin / Teacher / Student)
AI / EPS Template File Authentication level of Client(integer)	Sends to get verified	None

2.3.3 Queue Job

Input	Process	Output
Login / Password (String)	Authentication	Authentication Level (Admin / Teacher / Student)
SPF File containing • **Template Selected from Template Database** • **Variation Data unique to the job required to client** • **Authentication level of Client** • **Client Details**	Add to Job Queue Database	Reponse code

2.3.4 Approve Job

Input	Process	Output
Login / Password (String)	Authentication	IF Authentication level is not Admin deny Approval ability

[4]Dynamic text is text which is different for each job from the template file. An example of Dynamic text is the name on a trophy

[5]Static text is text which appears on every job from the template file. An example of Static text is the year on a trophy

SPF File from Job Queue DB	Approval Process Admin level client will see • Preview of SPF File • Line weight detail of the SPF file • Authentication level of the user • Client Details	IF Approved the job is sent to a folder to be laser cut The SPF File is converted to AI / EPS files to match the format of the Laser Cutter. If required, the job is split into multiple files IF Not Approved the job is deleted

2.3.5 Set Up Laser Cutter

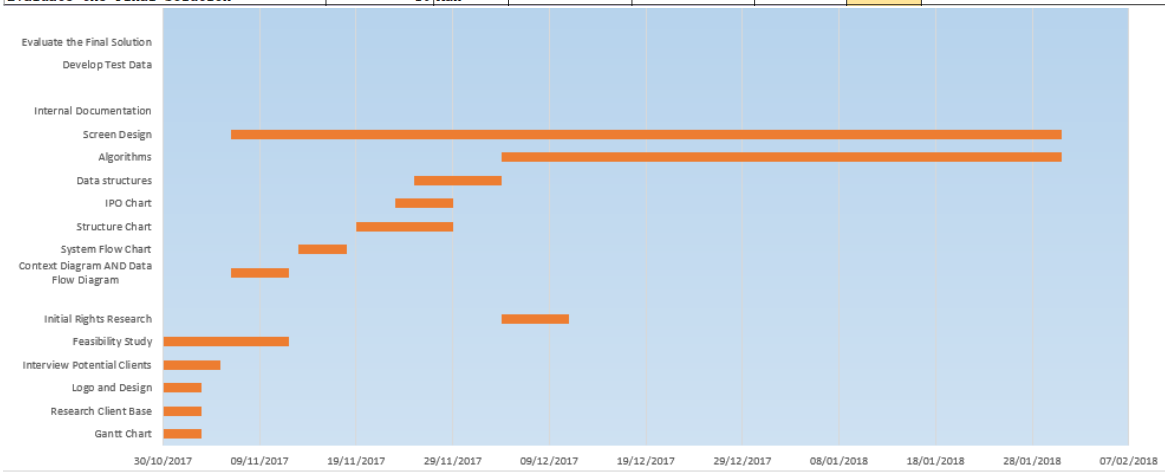
Input	Process	Output
Login / Password (String)	Authentication	IF Authentication level is not Admin deny Approval ability
AI / EPS File	Laser Cutter Process • An Admin must place the resource within the Laser Cutter • Open the files on the Laser Cutter computer • Print via Laser Cutter printer to the Laser Cutter • Remain by the Laser Cutter until the job is finished • Remove finished job and replace material if need be	Final Job

2.4 Gantt Chart [6]

[6] Gantt Chart by Shourov

L.A.M.P
Waxy LASER Solutions
Max Wharton-Jones, Shourov Quazi, Jack Jiang

Task Name	Marks	Member	Start Date	End Date	Duration	Complete	Notes
Gantt Chart	5	Shourov	30/10/2017	03/11/2017	4	✓	
Research Client Base	10	Jack	30/10/2017	03/11/2017	4	✓	
Logo and Design	-	Max	30/10/2017	03/11/2017	4	✓	
Interview Potential Clients	10	Shourov	30/10/2017	05/11/2017	6	✓	
Feasibility Study	10	Jack	30/10/2017	12/11/2017	13	✓	
Initial Rights Research	10	Jack	04/12/2017	11/12/2017	7	✓	
Context Diagram AND Data Flow Diagram	10	Max	06/11/2017	12/11/2017	6	✓	
System Flow Chart	5	Jack	13/11/2017	18/11/2017	5	✓	
Structure Chart	10	Shourov	19/11/2017	29/11/2017	10	✓	
IPO Chart	10	Shourov	23/11/2017	29/11/2017	6	✓	Redoing for End Date : 9/02/2018
Data structures	15	Jack	25/11/2017	04/12/2017	9	✓	Redoing for End Date : 9/02/2018
Algorithms	20	Everyone	04/12/2017	31/01/2018	58	✓	Redoing for End Date : 9/02/2018
Screen Design	15	Max	06/11/2017	31/01/2018	86	✓	
Internal Documentation	15	Everyone					
Develop Test Data	10	Everyone					
Evaluate the Final Solution	10	Max					



2.5 Data Dictionary [7]

Data Structure	Data Type	Length [8]	Description
tblTemplates Array (TemplateRecord)	Array(Of Record)		Contains details about approved templates created using the program. can be saved/loaded as .SPF files
	Auto_ID (Auto Number)	32	Index 0-2147,483,647 (for database). Primary Key
	DXF (String)	10000	Line and text data encoded using the DXF format
	tags (Array(Of String))	20x20	the criteria that the template meets
	material (String)	10	the material that the template uses
	length (real)	8	the length of each piece, in millimeters
	height (real)	8	the height of each piece, in millimeters
	thickness (real)	8	the thickness of each piece, in millimeters
	preview (Array(Of Image))	3x4MB	preview images for each template
	creatorId (Integer)	4	id of the creator, joined to tblUser.Auto_ID
	approverId (Integer)	4	id of the approver, joined to tblUser.Auto_ID
	dynamicTextList(Array(Of String))	10x10	Dynamic texts are text labels that are filled in by the user. They describe the name of the label (e.g. year)
	textLocation(Array(Of Point))	10x16	the location where the dynamic text will be filled
tblSumittedJobs Array (JobRecord)	Array(Of Record)		Contains information about jobs submitted to the IA staff for cutting.
	Auto_ID (Auto Number)	32	Index 0-2147,483,647 (for database). Primary Key
	template_ID	32	the id of the spf file to cut. Joined to tblTemplates.Auto_ID
	submitterId (Integer)	4	the id of submitter. Joined to tblUser._ID
	approverId (Integer)	4	the id of approver. Joined to tblUser._ID
	approved (boolean)	1	whether or not the job is approved
	submitDate (date)	32	the date in which the job is submitted
tblUsers Array (UserRecord)	Array(Of Record)		Contains information about all the users that are stored in the database
	Auto_ID (Auto Number)	32	Index 0-2147,483,647 (for database). Primary Key
	email (string)	20	the email used to sign up
	password (string)	20	password used to sign in
	accesslevel (integer)	4	1=student, 2=teacher, 3=admin. Determines if the user can do certain actions
	name (string)	20	full name of the user
lines Array (newline)	Array (Of Record)		Stores data about a shape or line drawn using the template designer. Lines is the array containing all the currently drawn lines
	point1x (real)	8	start point x co-ordinate
	point1y (real)	8	start point y co-ordinate

[7] Data Dictionary by Jack

	point2x (real) point2y(real) lineType (string) color (string) lineNumber (integer)	8 8 20 20 4	end point x co-ordinate end point y co-ordinate the type of shape drawn color of line the index of the line in the array
Point	Record x1 (real) y1 (real)	 8 8	Stores data of 1 point in 2d space location on x axis location on y axis
Unapproved Templates	Array (Of JobRecord)	20 x 10MB	Stores all the unapproved Templates

SCOPE	Identifier	Data Type	Length [9]	Description
Global	running userTags CurrentUser loggedIn dataConnection dataAdapter tool linesCollection USER TEACHER ADMIN sortBy	boolean Array (string) UserRecord boolean OleDbConnection OleDbDataAdapter string Array (Line) constant integer constant integer constant integer string	1 20 1 1 1 1 1 10000 1 1 1 1	whether the program is running or not Shows the tags that the user currently want to look at Stores the currently logged on user shows if te user is logged in or not Used to connect to the database used to read and write data to the database currently selected tool stores all the lines on the screen accesslevel of user=1 accesslevel of teacher=2 accesslevel of admin=3 how to sort the database, either by date or material
FilterTemplate	MatchingTemplate	Array (Template)	100	An array containing the templates that match a user's preferences
Global	prefCount tagCount	integer integer	1 1	Counter for looping through userpreferences Counter for looping through the tags in a template
SortByDate SortByMaterial SortByName	sortedTillElement largestpos upto nextUnsorted	integer integer integer integer	1 1 1 1	The index that the array is sorted up to. Starts at 0 (before the first element) the position of the largest element in the sub-array after index sortedTillElement the index of the element that the algorithm is currently checking the index of the next unsorted element in the array
checkValidSPF	file lineData	file (SPF) Array (Line)	1 10000	the spf file to check the lines that the spf file contains

2.6 Algorithms

```
BEGIN MAIN — Jack —
    running is a boolean = true

    InitalizeDatastructures
    Login
    LoadTemplates

    WHILE running = true
        selectModule(Max)
    END WHILE
END MAIN

BEGIN SUB selectModule — Max —
    Get user selection
    CASEWHERE user selection
        Select Template : selectTemplate
        Design Template : designTemplate
        Queue Job : queueJob
        Submit Template : submitTemplate
        Approve Template : approveTemplate
        Approve Job(s) : approveJob
        Set up Laser Cutter : setupLaser
        Change Settings : changeSetting
    END CASEWHERE
END SUB

BEGIN SUB passwordIsStrong(password) — Shourov —
    strong (boolean) = false
    IF password contains numbers AND password.length >= 8 THEN
        strong = true
```

[8] Length in bytes
[9] Length in elements

```
    END IF
    RETURN strong
END SUB

BEGIN SUB validEmail(email) — Shourov —
    IF email contains characters before a "@" symbol AND ends with "sbhs.nsw.edu.au" THEN
        RETURN true
    ELSE
        RETURN false
    END IF
END SUB

BEGIN SUB IsTeacherEmail(email) — Shourov —
    IF first character of email is NOT a number THEN
        RETURN true
    ELSE
        RETURN false
    END IF
END SUB

BEGIN SUB Login — Max —
    loggedIn = false
    email = "" (string)
    password = "" (string)
    found = false (boolean)

    get email, password
    IF email = "" AND password = "" THEN
        print "Please enter a Username and Password."
        succeeded = false
    ELSE
        index = 1
        WHILE index <= tblUsers.length AND found <> true
            IF tblUsers(index).email = email THEN
                found = true
                IF tblUsers(index).password = password THEN
                    loggedIn = true
                    print "login successful"
                ELSE
                    loggedIn = false // wrong password!
                    print "Wrong password!"
                END IF
            END IF
            index = index + 1
        END WHILE
        IF found <> true THEN
            print "email not found, please sign up"
        END IF
    END IF

    RETURN loggedIn
END SUB

BEGIN SUB SortTemplates(type) — Shourov —
    CASEWHERE type
        'date': SortByDate
        'material': SortByMaterial
        'name': SortByName
        otherwise : print "Invalid type supplied for sort!"
    END CASE
END SUB

BEGIN SUB SignUp — Max —
    failed = false (boolean)
    email = "" (string)
    password = "" (string)
    name = "" (string)

    Get email, password, name from user input
```

```
    IF email = "" OR password= "" OR name = "" THEN
        display "Enter an email, password and name"
        failed = true
    END IF

    IF failed <> true THEN
        IF validEmail(email) = true THEN
            IF IsTeacherEmail(email) = true THEN
                accesslevel = 1
            ELSE
                accesslevel = 0
            END IF
        ELSE
            failed = true
            display "Invalid email address!"
        END IF
    END IF

    IF failed <> true THEN
        IF passwordIsStrong(password) = true THEN
            IF CreateUser(email, password, name, accesslevel) = true THEN
                Login
            ELSE
                failed = true
                print "error creating account"
            END IF
        ELSE
            print "password must be 8 characters or longer, and contain numbers and text"
            failed = true
        END IF
    END IF

    succeed (Boolean) = true
    IF failed = true THEN
        succeed = false
    ELSE
        RefreshDatabases
        // Refresh database so new user is added to tblUsers
    END IF

    RETURN succeed
END SUB


BEGIN SUB CreateUser(email, password, name, accesslevel) — Jack —
    succeeded = true (boolean)
    IF SearchForUser(email) = false THEN
        // check that the user is not in the database already
        Open UsersDatabase for relative access
        Write email to UsersDatabase.email

        Write password UsersDatabase.password
        Write name to UsersDatabase.name
        Write accesslevel to UsersDatabase.accesslevel

        Close UsersDatabase
        RefreshDatabases
    ELSE
        succeeded = false
        print "email already used!"
    END IF
    RETURN succeeded
END SUB


BEGIN SUB SearchForUser(email) — Jack —
    index = 1
    found (boolean) = false

```

```
    WHILE index <= tblUsers.length AND found <> true
        IF tblUsers(index).email = email THEN
            found = true
        END IF
        index = index + 1
    END WHILE
    RETURN found
END SUB

BEGIN SUB LoadTemplates — Jack —
    IF loggedIn <> true THEN
        Login
    END IF

    IF loggedIn = true THEN //loggedIn will be set to true of Login is successful
        RefreshDatabases
    ELSE
        Print "You must login to access the templates"
    END IF
END SUB

BEGIN SUB InitalizeDatastructures — Jack —
    TemplateRecord is a record containing
        Auto_id (integer)
        DXF (string) // this contains all the line data on the template
        tags (Array(string)) //list of categories that apply to the temp
        material (string) // the material the template is to be cut on
        length (Integer)
        height (Integer)
        thickness (Integer)
        preview (Array(Image)) // preview images of the template. Optional
        creatorName (string)
        creator_id (Integer)
        approverName (string)
        approver_id (Integer)
        complete (boolean)
        dynamicTextLabel (Array(Of String))
        textLocation (Array(Of Point))
    END TemplateRecord

    JobRecord is a record containing
        Auto_id (integer)
        template (TemplateRecord)
        submitterName (string)
        submitterID (Integer)
        approverName (string)
        approverID    (Integer)
        approved (boolean)
        submitDate (date)
    END JobRecord

    UserRecord is a record containing
        Auto_id (integer)
        email     (string)
        password (string)
        accesslevel (string)
        name (string)
    END UserRecord
    // database tables
    tblTemplates is an array of TemplateRecord indexed from 1 to EOF
    tblSubmittedJobs is an array of JobRecord indexed from 1 to EOF
    tblUsers is an array of UserRecord indexed from 1 to EOF
    tblUnApprovedTemplates is an array of TemplateRecord indexed from 1 to EOF
    RefreshDatabases

    UserTags is an array of string indexed from 1 to EOF
    CurrentUser (UserRecord) =  null
    Tool = ""  (string) // currently selected tool
    loggedIn = false (boolean)
    USER = 0 (constant integer)
```

```
    TEACHER = 1 (constant integer)
    ADMIN = 2 (constant integer
    sortBy = "date" (string)
END SUB

BEGIN SUB RefreshDatabases — Jack —
    Open AllDatabases for sequential reading
    Get tblTemplates, tblSubmittedJobs, tblUsers, tblUnApprovedTemplates
    FilterTemplates
    SortTemplates(sortBy)

    Close AllDatabases
END SUB

// Templates can be filtered according to criteria, stored in the tags
// on the template item
BEGIN SUB FilterTemplates — Jack —
    MatchingTemplates is an array of TemplateRecord indexed from 1 to EOF
    currentTemplate = null (TemplateRecord)
    index = 1 (integer)

    WHILE index <= tblTemplates.Length
        currentTemplate = tblTemplates(index)
        IF TagIsPresent(currentTemplate) = true THEN
            Append current to MatchingTemplates
        END IF
        index = index + 1
    END WHILE
    tblTemplates = MatchingTemplates
END SUB

BEGIN SUB TagIsPresent(template) — Jack —
    // check if one of the tags in UserPreferences
    prefCount = 1 (integer)
    tagCount = 1 (integer)
    found = false (boolean)
    usertag = "" (string)
    currentTag = "" (string)
    // loop through each element in preferences, and check it with each
    // tag in the template
    WHILE prefCount <= UserPreferences.length AND found <> true
        tagCount = 1
        userTag = UserPreferences(prefCount)
        WHILE tagCount <= template.tags.length AND found <> true
            currentTag = template.tags(tagCount)
            IF currentTag = userTag THEN
                found = true
            END IF
            tagCount  = tagCount + 1
            END WHILE
        END WHILE
    RETURN found
END SUB

BEGIN SUB SortByDate — Jack —
    // sorts tblTemplates by submitted date, from latest to oldest
    // (descending) how much of the list is sorted, so we don't resort it
    // as it is already in descending order

    sortedTillElement = 0 (Integer)

    temp = null (TemplateRecord)
    current = null (TemplateRecord)

    // the location of the largest item in the list
    largestpos = 1 (integer)

    // what element the sort is at
    upto = 1 (integer)

```

```
    // the next element that is unsorted
    nextUnsorted = 1 (integer)

    // selection sort : will continually find the largest item in the list
    // and move it to the right position till the entire list is sorted
    WHILE sortedTillElement <= tblTemplates.length
        // set upto to the first element that is not sorted yet
        upto = sortedTillElement + 1

        // assume largest item is on the first element to be tested
        largestpos = upto

        WHILE upto <= tblTemplates.length
            current = tblTemplates(upto)

            IF current.date > tblTemplates(largestpos).date THEN
                // found newer largest item!
                largestpos = upto
            END IF
            upto = upto + 1
        END WHILE

        // largestpos is now the index of the largest element
        // swap it with the next unsorted item
        nextUnsorted = sortedTillElement + 1

        temp = tblTemplates(next)
        tblTemplates(next) = tblTemplates(largest)
        tblTemplates(largest) = temp
        sortedTillElement = sortedTillElement + 1
    END WHILE
END SUB

BEGIN SUB SortByMaterial — Shourov —
    // sorts tblTemplates by submitted material
    // (descending) how much of the list is sorted, so we don't resort it
    // as it is already in descending order

    sortedTillElement = 0 (Integer)

    temp = null (TemplateRecord)
    current = null (TemplateRecord)

// the location of the largest item in the list
    largestpos = 1 (integer)

    // what element the sort is at
    upto = 1 (integer)

    // the next element that is unsorted
    nextUnsorted = 1 (integer)

    // selection sort : will continually find the largest item in the list
    // and move it to the right position till the entire list is sorted
    WHILE sortedTillElement <= tblTemplates.length
        // set upto to the first element that is not sorted yet
        upto = sortedTillElement + 1

        // assume largest item is on the first element to be tested
        largestpos = upto

        WHILE upto <= tblTemplates.length
            current = tblTemplates(upto)

            IF current.material > tblTemplates(largestpos).material THEN
                // found newer largest item!
                largestpos = upto
            END IF
            upto = upto + 1
```

```
        END WHILE

        // largestpos is now the index of the largest element
        // swap it with the next unsorted item
        nextUnsorted = sortedTillElement + 1

        temp = tblTemplates(next)
        tblTemplates(next) = tblTemplates(largest)
        tblTemplates(largest) = temp
        sortedTillElement = sortedTillElement + 1
    END WHILE
END SUB

BEGIN SUB SortByName — Shourov —
    // sorts tblTemplates by submitted tags
    // (descending) how much of the list is sorted, so we don't resort it
    // as it is already in descending order
    sortedTillElement = 0 (Integer)

    temp = null (TemplateRecord)
    current = null (TemplateRecord)

    // the location of the largest item in the list
    largestpos = 1 (integer)

    // what element the sort is at
    upto = 1 (integer)

    // the next element that is unsorted
    nextUnsorted = 1 (integer)

    // selection sort : will continually find the largest item in the list
    // and move it to the right position till the entire list is sorted
    WHILE sortedTillElement <= tblTemplates.length
        // set upto to the first element that is not sorted yet
        upto = sortedTillElement + 1

        // assume largest item is on the first element to be tested
        largestpos = upto

        WHILE upto <= tblTemplates.length
            current = tblTemplates(upto)

            IF current.tags(1) > tblTemplates(largestpos).tags(1) THEN
                // found newer largest item!
                largestpos = upto
            END IF
            upto = upto + 1
        END WHILE

        // largestpos is now the index of the largest element
        // swap it with the next unsorted item
        nextUnsorted = sortedTillElement + 1

        temp = tblTemplates(next)
        tblTemplates(next) = tblTemplates(largest)
        tblTemplates(largest) = temp
        sortedTillElement = sortedTillElement + 1
    END WHILE
END SUB

BEGIN SUB PrintJobDetails(job) — Shourov —
    Initialize Printer Driver
    print "Job ID: ", job.id
    print "Date, time", job.time
    print "Submitter: ", job.submitter, "Approver:", job.approver
    print "Dimensions: " job.jobWidth, "x", job.jobHeight
    Close Printer Driver
END SUB
```

```
BEGIN SUB SelectTemplate — Shourov —
    LoadTemplates
    Display tblTemplates on GUI
    selectedTemplate = null (TemplateRecord)
    get selectedTemplate
    Display inputForm(size of selectedTemplate.dynamicTextLabel)
    dynamicText = empty (array(string))
    dynamicText = data from inputForm
    selectedTemplate.dynamicTextLabel = dynamicText
    Send selectedTemplate to queueJob
END SUB

BEGIN SUB inputForm(rows) — Shourov —
    Display input table with num Rows
END SUB

BEGIN SUB scrollC — Max —
    Tool = "Move Screen"
    //Moves the Field of the designer
    REPEAT
        CASEWHERE mouse Moves:
            Increased x: Design Field moves down by the same amount
            Decreased x: Design Field moves up by the same amount
            Increased y: Design Field moves left by the same amount
            Decreased y: Design Field moves right by the same amount
        END CASE
    UNTIL user releases middle click
END SUB

BEGIN SUB rightC — Max —
    IF user clicks Line THEN
        Ask user for linetype
        //(Cut, Raster, Vector, Includes descriptions of each)
        CASEWHERE user selection is
            Cut: linesCollection(lineNumber).lines.lineType = "Cut"
            Vector: linesCollection(lineNumber).lines.lineType = "Vector"
            Raster: linesCollection(lineNumber).lines.lineType = "Raster"
        END CASEWHERE
    END IF
END SUB

BEGIN SUB designTemplate — Max —
    lineIndex is an integer = 1
    REPEAT
        lines is a Record containing:
            x1 (Integer)
            y1 (Integer)
            x2 (Integer)
            y2 (Integer)
            lineType (string)
            color (string)
            lineNumber (Integer)
        END lines
        linesCollection is an Array of lines indexed from 1 to EOF
        CASEWHERE user selection is
            Straight line: drawStr
            Free line: drawFree
            Edit Logo: editLogo
            Edit Text: editText
            New Object: newObj
            Select Parts: partSelect
            Line Type: lineType
            Move: move
            Left Click: leftC
            Right Click: rightC
            Middle Mouse: scrollC
        END CASEWHERE
        Tool = ""
    UNTIL Done = True
    genImg
```

```
END SUB


BEGIN SUB moveScreen — Max —
    Tool = "Move"
    //Moves the Field of the designer
    IF left click is pressed THEN
        REPEAT
            CASEWHERE mouse Moves:
                Increased x:Design Field moves down by the same amount
                Decreased x: Design Field moves up by the same amount
                Increased y:Design Field moves left by the same amount
                Decreased y:Design Field moves right by the same amount
            END CASEWHERE
        UNTIL user releases left click
    END IF
END SUB

BEGIN SUB partSelect — Max —
    Tool = "Select Part"
    leftC
END SUB

BEGIN SUB leftC — Max —
    Highlight ends of clicked line
    WHILE mouse is hovering over line
        Display co-ordinates of Line ends next to mouse
    END WHILE
    IF left click is pressed AND a line is selected THEN
    // Moves the line
        REPEAT
            CASEWHERE mouse Moves:
                Increased x: Line moves up by the same amount
                Decreased x: Line moves down by the same amount
                Increased y: Line moves right by the same amount
                Decreased y: Line moves left by the same amount
            END CASEWHERE
        UNTIL Left click is released
    END IF

    IF left click is pressed AND a logo is selected THEN
        // move the logo
        REPEAT
            CASEWHERE mouse Moves:
                Increased x: Logo moves up by the same amount
                Decreased x: Logo moves down by the same amount
                Increased y: Logo moves right by the same amount
                Decreased y : Logo moves left by the same amount
            END CASEWHERE
        UNTIL Left click is released
    END IF

    IF left click is pressed AND a TextBox is selected THEN
        // move the logo
        REPEAT
            CASEWHERE mouse Moves:
                Increased x: TextBox moves up by the same amount
                Decreased x: TextBox moves down by the same amount
                Increased y: TextBox moves right by the same amount
                Decreased y: TextBox moves left by the same amount
            END CASEWHERE
        UNTIL Left click is released
    END IF
END SUB

BEGIN SUB lineType — Max —
    Tool = "Line Type"
    IF user left Clicks on a Line Segment THEN
        CASEWHERE user selection is
            Cut: linesCollection(lineNumber).lines.lineType = "Cut"
```

```
            Vector: linesCollection(lineNumber).lines.lineType = "Vector"
            Raster: linesCollection(lineNumber).lines.lineType = "Raster"
        END CASEWHERE
    END IF
END SUB

BEGIN SUB newObj — Max —
    Get user input
    // Logo, text Box, Line
    CASEWHERE User selection is
        Logo: Create a logo in Design Field with center at Mouse coordinates
        Text Box: Create a Text Box in Design Field with center at mouse coordinates
        Line: drawStr
    END CASEWHERE
END SUB

BEGIN SUB editText — Max —
    IF User left clicks Text box THEN
        Get user input (String)
    END IF
    Textbox.Text = user input
END SUB

BEGIN SUB editLogo — Max —
    REPEAT
        Access = "Denied" (String)
        Ask user for a logo
        Prompt File viewer
        IF the File is a PNG OR the file is a JPEG OR the File is a JPG THEN
            File = "Approved"
        ELSE
            Deny
        END IF
    UNTIL File = "Approved"
END SUB

BEGIN SUB drawFree — Max —
    Ask user for linetype
    //(Cut, Raster, Vector, Includes descriptions of each)
    Tool = "Free Line"
    freeTimer = 0 (integer)
    IF user clicks Left Click THEN
        REPEAT
            linesCollection(lineIndex).lines.x1 = mouse position x in Design Field
            linesCollection(lineIndex).lines.y1 = mouse position y in  Design Field
            freeTimer = freeTimer + 1
            Wait 0.01 Seconds
            linesCollection(lineIndex).lines.x2 = mouse position x in Design Field
            linesCollection(lineIndex).lines.y2 = mouse position y in Design Field
            lineIndex = lineIndex + 1
        UNTIL Left Click is released
    END IF
END SUB

BEGIN SUB drawStr — Max —
    Ask user for linetype
    //(Cut, Raster, Vector, Includes descriptions of each)
    Tool = "straightLine"
    IF User clicks left mouse AND Tool = straightLine THEN
        linesCollection(lineIndex).lines.x1 = mouse position x in Design Field
        linesCollection(lineIndex).lines.y1 = mouse position y in Design Field
        IF user clicks left mouse THEN
            linesCollection(lineIndex).lines.x2 = mouse position x in Design Field
            linesCollection(lineIndex).lines.y2 = mouse position y in Design Field
        ELSE
            linesCollection(lineIndex).lines.x1
            linesCollection(lineIndex).lines.y1
            lineIndex = lineIndex - 1
        END IF
    END IF
```

```
    lineIndex = lineIndex + 1
END SUB

BEGIN SUB genIMG — Max —
    Prompt File viewer
    Load Ai File
    Take image from Ai Logo
    Use image as icon for file in file viewer
END SUB

BEGIN SUB checkValidSPF(file) — Shourov —
    Open file for sequential reading
    lineData = null (Array of line, from 0 to EOF)
    get lineData from file

    valid = true
    index = 1
    WHILE lineData.Length <= index AND valid <> false
        IF lineData(index).color <> "RED" AND lineData(index).color <> "BLUE" AND lineData(index).color
            valid = false
            print "Lines must be red, black or blue"
        END IF
        index = index + 1
    END WHILE
    close file
    RETURN valid
END SUB

BEGIN SUB queueJob — Jack —
    IF loggedIn = false THEN
        Login
    END IF
    currentDate = null (date)
    get currentDate
    IF loggedIn = true AND CurrentUser.accesslevel >= TEACHER THEN
        // teacher/admin, and logged in
        Get SPF file
        IF file.complete = true AND checkValidSPF(file) = true THEN
            // send it
            Open JobsDatabase for relative access
            write file to JobsDatabase.template
            write CurrentUser.name to JobsDatabase.submitterName
            write CurrentUser.Auto_ID to JobsDatabase.submitterID
            write "" to JobsDatabase.approverName
            write -1 to JobsDatabase.approverID
            write false to JobsDatabase.approved
            write currentDate to JobsDatabase.submitDate
            Close JobsDatabase
            RefreshDatabases
        ELSE
            print "File is incomplete, or contains errors"
        END IF
    ELSE
        print "Only teachers/administrators can submit jobs"
    END IF
END SUB

BEGIN SUB submitTemplate — Jack —
    LoadTemplates\\
    templatefile = null (SPF file)
    get templateFile
    IF loggedIn <> true THEN
        Login
    END IF
    IF loggedIn = true AND CurrentUser.accesslevel >= TEACHER THEN
        Open unapprovedTemplateDatabase
        write templateFile to unapprovedTemplateDatabase
        Close unapprovedTemplateDatabase
    ELSE
        print "please login to submit templates"
```

```
    END IF
END SUB

BEGIN SUB approveJob — Jack —
    IF loggedIn = false THEN
        Login
    END IF

    selectedJob = null (JobRecord)
    confirmation = false (boolean)

    IF loggedIn = true AND CurrentUser.accesslevel >= ADMIN THEN
        // admin, and logged in
        display tblSubmittedJobs
        get selectedJob
        print "Approve job?"
        get confirmation
        IF confirmation = true THEN
            print "Do you want to print job details to paper?"
            get confirmation
            IF confirmation = true THEN
                PrintJobDetails(selectedJob)
            END IF
            selectedJob.approverName = CurrentUser.name
            selectedJob.approverID = CurrentUser.Auto_ID
            selectedJob.approved = true
        ELSE
            print "Cancelled"
        END IF
    ELSE
        print "Only administrators can approve jobs"
    END IF
END SUB

BEGIN SUB setupLaser — Shourov —
    Initialize Laser Driver
    Process SPF file as DXF file in lasercutter printer
    Close Laser Driver
END SUB

BEGIN SUB ChangeSort(criteria) — Shourov —
    CASEWHERE criteria
        date: sortBy = "date"
        material: sortBy = "material"
        name: sortBy = "name"
        otherwise: print "Invalid criteria"
    END CASE
END SUB

BEGIN SUB changeSettings — Shourov —
    CASEWHERE userinput
        add category: AddTag(tag)
        remove category: RemoveTag(tag)
        change sort: ChangeSort(criteria)
        otherwise: print "invalid input"
    END CASE
END SUB


BEGIN SUB approveTemplate — Jack —
    IF loggedIn = false
        Login
    END IF
    selectedTemplate = null (JobRecord)
    confirmation = false (boolean)

    IF loggedIn = true AND CurrentUser.accesslevel >= ADMIN THEN
        // admin, and logged in
        display tblUnapprovedTemplates
```

```
        get selectedTemplate

        print "Approve template?"
        get confirmation
        IF confirmation = true THEN
            delete selectedTemplate from tblUnapprovedTemplates
            append selectedTemplate to tblTemplates
        ELSE
            print "Cancelling"
        END IF
    ELSE
        print "Only administrators can approve templates"
    END IF
END SUB

BEGIN SUB AddTag(tag) — Shourov —
    // add a category to the user settings
    UserTags(UserTags.length+1) = tag
END SUB

BEGIN SUB RemoveTag(tag) — Shourov —
    // remove a category to the user settings
    index = 1
    found = false
    WHILE index <= UserTags.length AND found <> true THEN
        IF UserTags(count) = tag THEN
            delete index field from UserTags
            found = true
        END IF
        index = index + 1
    END WHILE
    IF found <> true THEN
        print "Cannot find tag"
    END IF
END SUB
```

2.7 Screen Design Principals [10]

hewwo

[10] Screen Design by Max