EBM Installation Guide

Introduction

This repository contains the programs required to set up a new Espresso Book Machine (EBM) for printing books. It also contains the entire source code for the EBM.

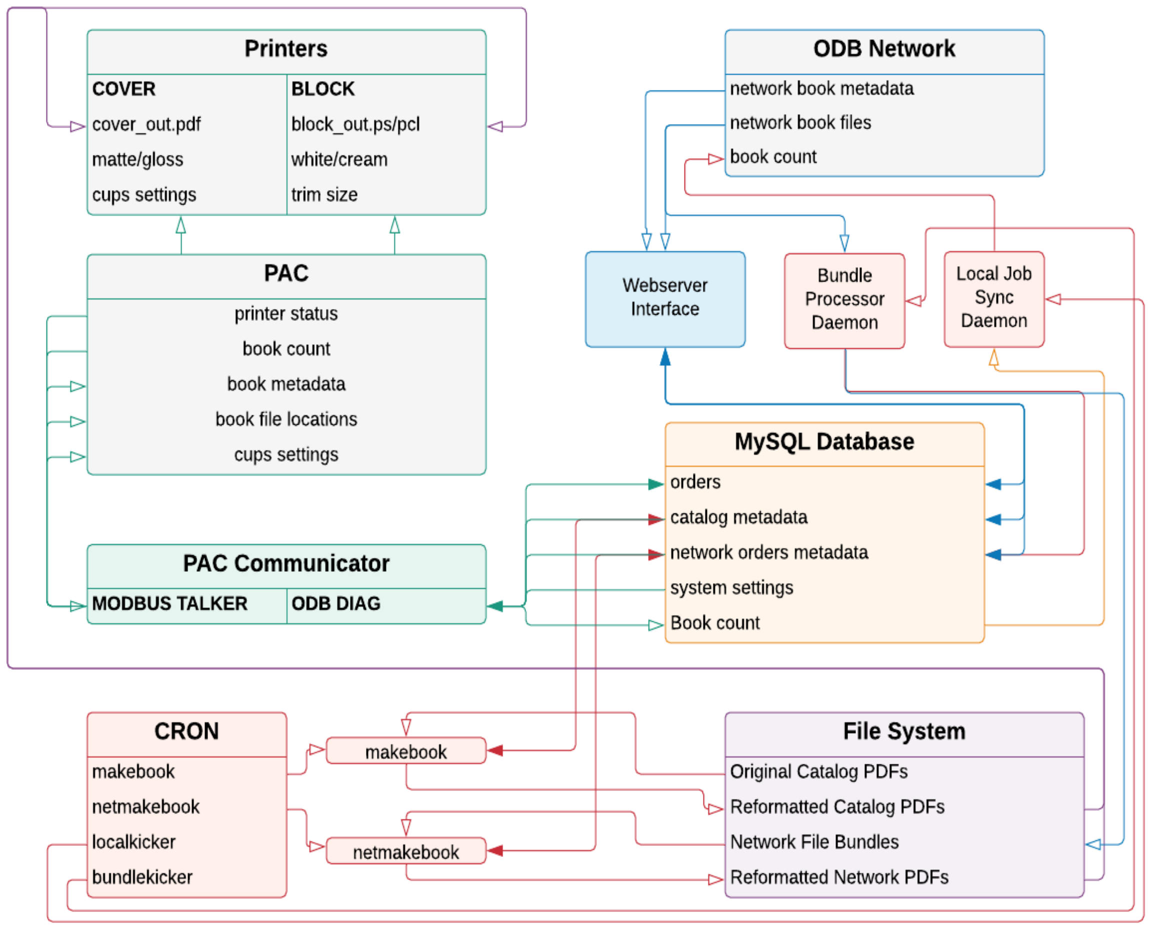
The EBM is a machine that turns PDF files into bound paperback books. It has access to a large online catalog of books and a smaller offline catalog of books, and can additionally be given new PDF files, which are processed into a print-ready format and added to the offline catalog. The hardware of an EBM consists of a block printer, a cover printer, and a Mac Mini to run the software.

The EBM software is broken up into several parts: the Printer Access Control (PAC) Communicator, the On-Demand Book (ODB) Network, the Webserver Interface, and the various processing cron jobs and daemons. These parts do not directly interact with one another; instead, metadata is stored within the MySQL database, and PDF files are saved to a shared location in the file system. Each part of the system gets its information from, and directly updates, the database.

The PAC Communicator is an I/O interface for the printers' PAC. It pulls print orders from the orders queue in the database, and sends them to the PAC to be printed and bound. Once the PAC indicates the order is finished, the PAC Communicator updates the database to mark the order complete. It also displays information about the next book in the queue and provides options for the EBM operator, such as the ability to change the paper type. The PAC Communicator is a C++ program that runs as an executable file.

The ODB Network is external to this repository. It lives on its own server and houses the main book catalog. The Webserver Interface is local, and provides the primary GUI for EBM operators, allowing them to access, search, and print from both the ODB Network's catalog and the machine's local, offline catalog. EBM operators also use the Webserver Interface to see the entire queue and order history, as well as edit book metadata. The Webserver Interface is a PHP application that runs locally on an Apache server and is accessed via an internet browser.

Lastly, there are 4 cron jobs and 2 daemons which process PDFs both from the local system and from the ODB Network. The Bundle Processor Daemon pulls requested files from the ODB Network. The "makebook" cron checks for unprocessed local PDFs and reformats their text and covers to fit within the final book's stated measurements. The "netmakebook" cron does the same for PDFs pulled from the ODB Network. The "localkicker" and "bundlekicker" crons update the Local Job Sync Daemon about fulfilled, errored, and canceled orders, and the Local Job Sync Daemon sends those updates to the ODB Network.



(White arrows signify a one-way dataflow, where data is received but not modified at its source. Grey boxes indicate items that are external to this repository.)

Requirements

Materials

In order to set up an EBM machine, you will need the following components:

- A Mac mini (any version) running Mac OSX
- An EBM Block Printer
- An additional large-format color printer, positioned in the slot for the cover printer on the EBM Block Printer
- 2 USB-A to USB-A cables
- Credentials for the EBM network
- Two GPG passkeys, one public and one private, plus a passphrase

Software

Before installing the EBM software, create a new account on the Mac mini named ‘ebm’ with full administrator privileges. Sign into this account and install the following programs:

Program Name	**Download/Installation Instructions**
XCode	https://apps.apple.com/us/app/xcode/id497799835
MySQL Community Server 8	https://dev.mysql.com/downloads/mysql/
Open Source Qt 5 Source Code	https://wiki.qt.io/Building_Qt_5_from_Git#Getting_the_source_code
Homebrew	https://brew.sh/
GnuPG	https://www.gnupg.org/download/

Installation

The following steps will convert a Mac Mini into a working EBM machine, ready to be hooked up to the printer hardware or to serve as a work environment for engineers who wish to improve the code.

Make sure you have set up an administrator account named ‘ebm’ and downloaded all of the software under ‘Requirements’ before starting.

The installation has been broken down into nine steps. It is very important to **perform these steps in order**, as many steps rely on setup from previous steps. Be sure to check the troubleshooting doc if you run into an error.

1. Additional Downloads

Before installing the EBM software directly, it is important to make sure everything the EBM requires has been downloaded and installed properly.

a) Download this repository

The repository should be created in root for user *ebm* (/Users/ebm or ~).

Open a Command terminal with Administrative privileges, and run the following:

```
cd ~
git clone https://[USERNAME]@bitbucket.org/teamodb/ebm-code.git
```

(where [USERNAME] is your bitbucket username)

Enter your bitbucket password when prompted.

If the clone was successful, you will now see a new directory named 'ebm-code' in your user root directory.

Leave the Command terminal open. You will use it for most of the remaining steps.

b) Set up Python packages

Python 2.7 comes with Mac OSX by default. However, some additional Python packages need to be installed: MySQL-python (to connect to the database), Paramiko (to connect to the ODB Network), and CherryPy (to connect to the Webserver Interface).

To install the necessary packages, run in the Command terminal:

```
sudo easy_install pip
sudo pip install MySQL-python
sudo pip install paramiko
sudo pip install cherrypy --upgrade --ignore-installed six
```

c) Install printer drivers

The block and cover printers have their own proprietary drivers.

Plug the printers in, then connect them to the Mac Mini using the USB cables. You should be prompted to install the printer drivers automatically once they are connected.

This step *must* be performed before running the PAC Communicator.

2. Configure MySQL Database

The MySQL database must be created and set up with the prerequisite tables before the EBM can run.

a) Add MySQL to PATH

MySQL needs to be accessible to the EBM programs which access it. This step will add MySQL as an environment variable to the PATH.

Run the following in the Command terminal:

```
echo 'export PATH="/usr/local/mysql/bin:$PATH"' >>
~/.bash_profile
```

b) Create database

Installing MySQL will not create a database; you must create it manually. The EBM expects the database to be named `odb` .

Pull up the MySQL monitor terminal by running the following in the Command terminal:

```
mysql -u root -p
```

Enter the root password when prompted.

Then, run the following commands in the MySQL monitor terminal to set up the database and default user:

```
CREATE DATABASE IF NOT EXISTS odb;
CREATE USER 'odb'@'localhost' IDENTIFIED WITH
mysql_native_password BY '[PASSWORD]';
GRANT ALL ON odb.* TO  'odb'@'localhost';
```

(where [PASSWORD] is the password for the root user)

Close the MySQL terminal by typing `quit` or `\q` . This will bring you back to the regular Command terminal.

c) Create tables

Included in this repository under `setup/command_line` is a script named `setup_mysql_tables.sh` , which will run each file in the sql directory in order.

Run it via the Command terminal by typing the following line:

```
sh ~/ebm-code/setup/command_line/setup_mysql_tables.sh
```

Enter the MySQL root password when prompted.

Alternatively, you can create each table individually by reopening the MySQL monitor terminal and typing this command:

```
source /Users/ebm/ebm-code/sql/[TABLENAME].sql
```

(where [TABLENAME] is the name of the table, for example available_block_printers)

Caution: Running these scripts will drop any existing tables. Be sure to back up the database if it has already been used.

3. PAC Communicator Configuration

The PAC Communicator cannot be run out of the box. The Qt library which it relies on needs to be set up to handle MySQL connections, and then the code for the PAC Communicator must be compiled to create an executable file.

a) Add Qt to PATH

Add Qt as an environment variable to the PATH by running the following in the Command terminal:

```
echo 'export PATH="[PATH/TO/QT]:$PATH"' >> ~/.bash_profile
```

(where [PATH/TO/QT] points to the bin directory for Qt, for example /Users/ebm/Qt/5.12.3/clang_64/bin)

b) Set up the QMySQL Driver

Some versions of Qt do not come with the QMYSQL Driver enabled. To check if this is the case with your installation, go to

/Users/ebm/Qt/[VERSION]/clang_64/plugins/sqldrivers

where [VERSION] is the version of Qt (ex. 5.12.3), and look for a file named **exactly** libqsqlmysql.dylib .

If there is no file with that name in that folder, perform step i. If this file exists, skip to step ii.

i) Compile QMYSQL to create dylib files

Recompile the source via the 'configure' script.

```
cd ~/Qt/[VERSION]/Src
./configure -sql-mysql
make
```

where [VERSION] is the version of Qt (ex. 5.12.3).

NOTE: This step can take several hours. **Do not continue with installation until the compiler is finished.**

Once the build completes, go into the source for MySQL and make it:

```
cd
/Users/ebm/Qt/[VERSION]/Src/qtbase/src/plugins/sqldrivers/mysql
qmake
make install
qmake -- MYSQL_PREFIX=/usr/local
make sub-mysql
```

where [VERSION] is the version of Qt (ex. 5.12.3).

Once this is done, check to see if it was successful by going to the directory `/Users/ebm/Qt/[VERSION]/clang_64/plugins/sqldrivers` and looking for `libqsqlmysql.dylib` .

NOTE: There are two directories named sqldrivers, one under `Src/qtbase/src` and one under `clang_64` . The directory you need to check is the one under clang_64.

ii) Load QMYSQL driver

Qt needs to be told where to find MySQL. Until you pinpoint the MySQL Client Library, setup scripts and the PAC Communicator will fail with the error "QMYSQL driver not loaded."

Check what version(s) of the MySQL Client Library you have. This is a Dynamic Library (dylib) file named "libmysqlclient" that lives within mysql's lib directory.

Run in the Command terminal:

```
cd /usr/local/mysql/lib
ls -a
```

You will get a list of all of the files in that directory. Check if you have a file with this **exact** name: libmysqlclient.dylib .

If you do not have this file, there should be a file named libmysqlclient.[X].dylib

(where [X] is a version number). If this is the case, take note of the number.

Once you have found the libmysqlclient.dylib file, check to see what library Qt is expecting. Go to the SQL Drivers directory under Plugins for the compiler (clang) you are using. In the Command terminal, run:

```
cd
~/Qt/[VERSION]/clang_64/plugins/sqldrivers
otool -L libqsqlmysql.dylib
```

(where [VERSION] is the version of Qt you installed, ex 5.12.3 (make sure to type the filename correctly: it's Lib Q SQL MySQL))

This command will list the libraries that Qt's SQL Driver is looking for. The important one is libmysqlclient. Make sure that libmysqlclient points to the client file.

If the client version is wrong, then while within the same directory, run in the Command terminal:

```
install_name_tool -change [OLD PATH] [NEW
PATH] libqsqlmysql.dylib
```

(where *[OLD PATH]* is the directory currently listed by otool, and *[NEW PATH]* is the actual location of your client file)

For example, your command may look like this:

```
install_name_tool -change
@rpath/libmysqlclient.20.dylib
/usr/local/mysql/lib/libmysqlclient.dylib
libqsqlmysql.dylib
```

c) Run *Printer Settings*

The *Printer Settings* app is a small GUI program that sets the default printer drivers, as well as some other settings such as which paper tray to use. These settings must be set before the PAC Communicator can run.

Run the *Printer Settings* app inside the printer_settings directory, and choose the settings that match your current setup. Note that the only printer drivers listed by this app are ones that already exist on the machine. If drivers are added later, re-run the app.

d) Compile the PAC Communicator

In the Command terminal, change directories to the PAC_Communicator directory, then run QMake and Make to compile the PAC Communicator.

```
cd ~/ebm-code/PAC_Communicator
qmake
make
```

This will create the PAC Communicator app under /Users/ebm.

Note: The PAC Communicator will crash on opening if you have not saved settings via the printer settings app (previous step, 3.C) or if you have not installed printer drivers (step 1.G).

You can test the PAC Communicator in Island Mode (disconnected from PAC) via the following command:

```
./PAC_Communicator.app/Contents/MacOS/PAC_Communicator
--island
```

If everything is set up properly, the PAC Communicator should run in Island Mode without crashing or producing errors. If you do not run in Island Mode it will hang but should not crash.

4. Webserver Configuration

The Webserver runs locally on an Apache server and is accessed by going to localhost in an internet browser.

a) Set up Apache Server

Apache should be included with the OS; all you need to do is to change the configuration and start it.

Replace the file in `/etc/apache2/httpd.conf` with the httpd.conf file in this repository.

To start the server, run

```
sudo apachectl start
```

b) PHP Setup

Copy the included PHP.ini file to `/etc/php.ini` , then test by opening Safari and going to "localhost" (do not add http:// or .com).

5. Network/Bundle Processor Credentials

For security reasons, the credentials for the network connection are not included in this repository. In order to connect to the network to pull the network catalog, you will need to add these in manually to the table "credentials". You will additionally need to provide credentials to the order_tosser script.

a) In the database

Each machine stores its network credentials in the database in a table called `credentials` with the primary key set to 1.

In the Command terminal, run:

```
mysql odb -uodb -p
INSERT INTO credentials VALUES id=1, type='active',
ready_url='https://web.odbnetwork.com/readyJobs',
update_url='https://web.odbnetwork.com/update',
ebm_id='[EBM ID]', ebm_passwd='[EBM PASSWORD]',
sftp_hostname='island.odbnetwork.com', sftp_port='22',
sftp_uname='[SFTP USERNAME]', sftp_passwd='[SFTP
PASSWORD]',
error_update_url='https://web.odbnetwork.com/errorUpdate'
,
local_job_updater_url='https://web.odbnetwork.com/localEb
mJobs';
```

(where:

[EBM ID] is the username for this particular EBM setup

[EBM PASSWORD] is the password for the EBM ID

[SFTP USERNAME] is the username for SFTP (separate from EBM ID)

[SFTP PASSWORD] is the password for SFTP (separate from EBM password))

b) In order_tosser

Create the file /bin/order_tosser.ini and type the following into it:

```
net_uname   = [EBM ID]
net_passwd  = [EBM PASSWORD]
net_authkey = [NETWORK AUTHORIZATION KEY]
net_ip      = [STORE IP ADDRESS]
```

Then save the file.

6. Set MySQLdb library paths

As is the case with Qt, Python needs to know where the MySQL library is, along with some other dependencies (SSL and Crypto).

Run in the Command terminal:

```
cd /Library/Python/[VERSION]/site-packages/MySQLdb/
otool -l _mysql.so
```

Otool will tell you where Python is looking for libmysqlclient, and what versions of libssl and libcrypto it expects. Until you set the paths, it will look in the wrong place.

Run in the Command terminal:

```
install_name_tool -change @rpath/libmysqlclient.[VERSION].dylib
/usr/local/mysql/lib/libmysqlclient.dylib _mysql.so

install_name_tool -change libssl.[VERSION].dylib
/usr/local/opt/openssl/lib/libssl.[VERSION].dylib _mysql.so

install_name_tool -change libcrypto.[VERSION].dylib
/usr/local/opt/openssl/lib/libcrypto.[VERSION].dylib _mysql.so
```

(where [VERSION] matches what is displayed by otool).

7. Set up GPG

GPG is necessary for connecting to the ODB Network, as it encrypts, decrypts, and signs data such as book requests and files that are downloaded from the server.

a) Set up keys

In order to zip/unzip data bundles from the network you need to set up the GPG keys and passphrase. You should have two of them, one public and one private. The keys look like this:

```
-----BEGIN PGP [PUBLIC/PRIVATE] KEY BLOCK-----
[hash]
-----END PGP [PUBLIC/PRIVATE] KEY BLOCK-----
```

You should have one private key and one public key, and additionally, have a short passphrase (approx 15 bytes). Create a file named:

`/opt/odb/gpg/passphrase.txt`

and paste the passphrase (no leading or trailing characters) inside. Additionally, save the keys in txt files (location does not matter).

For both the public and private key, run in the Command terminal:

```
gpg --import [PATH/TO/KEY.txt]
```

(where [PATH/TO/KEY.txt] is the full path to the key file)

For the secret key you will be prompted for the passphrase.

Now set the key to trusted Run in the Command terminal:

```
gpg --list-keys
gpg --edit-key [KEY] trust quit
```

(where [KEY] is the key returned from list-keys)

In this prompt, re-enter the passphrase, then choose option '5', then 'y' to confirm.

b) Connect to TTY

Set the GPG environment variable to use TTY correctly.

To add GPG to your PATH, run in the Command terminal:

```
echo 'export GPG_TTY=$(tty)' >> ~/.bash_profile
```

c) Allow automatic passphrase entry

Configure GPG-Agent to allow for automatic passphrase entry. In the file named `/Users/ebm/.gnupg/gpg-agent.conf` , add the line:

```
allow-loopback-pinentry
```

Note that gpg-agent.conf may not exist; if it does not, create it.

8. Makebook/Netmakebook

Makebook and netmakebook are cron jobs which run periodically to process book PDF files into files compatible with the printer. They need several libraries and directories before they can run properly.

a) Create necessary directories

The EBM expects a few extra directories that need full read, write, and execute permissions.

Run:

```
mkdir /opt/odb/catalog
mkdir /opt/odb/makebook
mkdir /opt/odb/netmakebook
mkdir /opt/odb/network

sudo chmod -R 777 /opt/odb/catalog
sudo chmod -R 777 /opt/odb/makebook
sudo chmod -R 777 /opt/odb/netmakebook
sudo chmod -R 777 /opt/odb/network
```

b) Install dependencies

Makebook and netmakebook rely on three Python libraries to process text and images.

To install these libraries, run in the Command terminal:

```
brew install poppler
brew install ghostscript
brew install imagemagick
```

c) Copy test book files into catalog

Move the five directories found under setup/catalog into /opt/odb/catalog.

d) Create cron jobs

Run:

```
crontab -e
```

Then paste the contents of setup/crontab.txt into the editor. To exit the editor, press the Esc key, then, :, then w, then q, then enter to save changes.

9. PAC Connection

The PAC communicates on `192.168.0.250` and must be opened in a Chrome browser window to operate. Open Chrome and navigate to the following page:

```
http://192.168.0.250/views/main_a.html
```

The PAC also requires a specific Chrome extension to allow its legacy Java applet to run. The extension has been included in this repository under setup/chrome_extension.

In chrome, go to

```
chrome://extensions/
```

and turn on Developer Mode (upper right-hand corner). Then click "Load Unpacked" (upper left-hand corner) and select the folder with the extension. When opening the PAC you will have to enable this extension.

You should now have a fully operational EBM machine.

Testing the EBM

To test that the install has run correctly, you can print a test book.

1. Open the executable file for the PAC Communicator, as well as a Chrome browser window.
2. Open two tabs in Chrome, and navigate one to localhost and the other to http://192.168.0.250/views/main_a.html .
3. In the first tab, navigate to Search, then enter “test” under Book Title and hit Enter. Several test books should appear in the search results.
4. Pick whichever test book you like and select “Add to Queue”.
5. Check the display for the PAC Communicator. The test book you selected should now be listed in the “next book in queue”.
6. Go back to localhost and navigate to the Print Queue.
7. Select “Print Next Book” and wait.
8. Within a few minutes, both printers should begin printing. The book will take some time to print.
9. Once the book is finished, it will drop into the chute on the block printer. Remove the book and check it to make sure the printers have printed it correctly.

Troubleshooting

Installing the EBM is a complicated process, and it is easy to make mistakes during the installation. Here are some common issues and their solutions:

MySQL

There is no libmysqlclient.dylib *or* libmysqlclient.[x].dylib file in /MySQL/lib (step 3.b.ii): It is possible that your MySQL download did not include the required libraries for connecting to other programs. You can get these libraries by installing Connector/C. This can be done using Homebrew. Run in the command line:

```
brew install mysql-connector-c
```

After installing Connector/C, check /MySQL/lib again. The file should now be there.

Printer Settings

The Printer Settings app does not have any options to select under Cover Printer:
If your cover printer does not show up as a printer option, this means its driver is not installed correctly. Usually, plugging the printer into the Mac mini is enough to start the installation, but if that does not work, go to the webpage for the manufacturer of the printer you are using and search for a download of the driver for your model of printer. Once the printer driver installation is complete, close and reopen the Printer Settings app to see your cover printer as an option.

You may also need to restart the Mac mini after the driver is installed.

PAC Communicator

The PAC Communicator instantly crashes when opened with no error message
If the PAC Communicator cannot be opened without immediately crashing, that means it cannot find the printer settings. Open the Printer Settings app, choose the correct cover and block printer drivers, and make sure to save your chosen options.

If this does not solve the problem, one of your printer drivers may be corrupted. Try printing a file directly, by opening a text file, typing some words, and pressing Command (⌘)+P to print it. If it fails to print, that means the printer driver did not install correctly and must be reinstalled.

If both printers are capable of printing, check to see that the database holds the correct printer settings. Open MySQL and run

```
SELECT * FROM printer_settings WHERE id = 1;
```

This should return one row of data. Check that the cover_printer and block_printer fields are both set to the correct printers. If not, you can manually edit the table entry.

The PAC Communicator displays the error “QMySQL not loaded”:
There is something wrong with your MySQL connection. Check and make sure you followed steps 3.a and 3.b exactly. To check if you performed everything correctly ***TODO something with otool, look this up***

The PAC Communicator opens, but then hangs/freezes when attempting to interact with it:
Is the Mac mini currently connected to the block printer? If not, the PAC Communicator will hang unless run in Island Mode. To run in Island Mode, run the following in the Command terminal:

```
./PAC_Communicator.app/Contents/MacOS/PAC_Communicator --island
```

The PAC Communicator does not show any books in the queue even though a book has been ordered:
Check that the book was added to the database correctly. Open MySQL and run:

```
SELECT * FROM orders ORDER BY id DESC LIMIT 1;
```

If it does not return the book that was ordered, then re-order the book.

Webserver Interface

Unable to connect to localhost:
Make sure you have started the Apache server and that it is running correctly. ***TODO get name of apache program etc

No results appear when searching "test" in the Webserver Interface:
The test books have not been placed in their proper directory. You can find the test books in this repository under `setup/catalog` , and they should be moved to the directory `/opt/odb/catalog` .

Only the test books show up when searching for books in the Webserver Interface:
There is an issue with your connection to the ODB Network. Make sure the credentials you entered in step 7 are correct.

Log Message	**Meaning**	**How to fix**
No books found, exiting.	Makebook did not find any unprocessed books.	There are two possibilities: 1) Makebook has already attempted to process the book. Check earlier in the logs to see if you can find where it processed the book. 2) There was an error adding the book to the queue. Try adding it again via the Webserver Interface.
Save for [book title] failed.	Makebook was unable to save the processed book to /opt/odb/catalog	Check that the folder /opt/odb/catalog exists and that its permissions are 777.
Unable to process [book title], wrong	The book file is not in PDF format, or has	See if you can open the book file as a PDF and that it displays correctly. If not, there is

format.	been corrupted	something wrong with that particular book file. Delete it via the Webserver Interface and try a different book.
Processing [different book title] successful	Makebook detected another book in the queue and processed that instead of the book you wanted.	Another book had gotten into the book queue. Did you click on the wrong book? Check that the correct book is in the queue and wait for makebook to run again.
Processing [correct book title] successful	Makebook processed the book for printing.	The issue is not with makebook. There is something wrong with the PAC Communicator. Make sure the PAC Communicator is open and running.

Printing Issues

Test book has been next in queue for over 10 minutes after clicking print:
The makebook cron checks every 5 minutes for books to process. It may still be processing the test book, or it may not be running properly. First see if makebook is running by checking the Activity Monitor. Press Command(⌘) + Alt + Escape to open Activity Monitor, and check the list of running processes for makebook. If it is running, wait until it finishes.
You can check the logs for makebook under `~/logs/makebook` . Open the logs and check the most recent message against the following table.

The book’s cover is upside-down and inside-out:
The cover printer is not mounted to the block printer correctly. Remove the cover printer, turn it 180 degrees, and remount it.