Chemical Synthesis Automation

by

Michael Wenning

A Major Qualifying Project

Submitted to the Faculty of

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Bachelor of Science in

Computer Science

by

May 2022

Approved:

Smith, Therise

This report represents work of WPI undergraduate students submitted to the faculty as evidence of a degree requirement. WPI routinely publishes these reports on its web site without editorial or peer review.

Abstract

The goal for this project was to further build on the existing work and understanding of a chemical synthesis machine for AbbVie and leave the project in a good state for continued work. The purpose of this machine is to automize chemical synthesis given some input from lab workers. It will do so through the use of several devices and softwares that can communicate with each other using commands for desired actions.

Table of Contents

List of Figures

CHAPTER 1

Introduction

1.1 Problem Description

Chemical synthesis experiments can become tedious and dangerous in some cases for humans, and can become quite time consuming in the lab. This is why automating the process through machines can be a worthwhile project that can also improve safety and efficiency. Chemists should be able to give the machine some input about what kind of chemical reactions it should perform, and it can safely handle the tasks with little human interaction.

After the input is collected from the chemists through the interface, the machine can make use of natural language processing to convert the input into a set of instructions that include some nouns and verbs. That information will then be used to administer commands to several machines that can independently control certain jobs of the process. For example, one machine will take commands regarding the syringe pump, such as the position it needs to move the pump to, whether to dispense or draw the liquid, or even turn on and off. It will be the job of the conductor machine to take the commands generated from the interface and send them to the correct machine for the action.

1.2 Existing Work

Research into similar machines has been a growing field recently, especially in machines that can handle more generalized chemical processes. In some cases people even look into using machine learning to make new discoveries [ML], especially with the growing resources that can be used to train machines. However there seems to be very little that attempts to tackle the same goal as this project, that is they are very specific about their purpose while this machine takes a more generalized approach so that it can have uses in all kinds of chemical reactions.

There is still a lot more work and research to be done until chemical synthesis can be industrialized but we are making good progress, and the combination of machine learning and automizing lab equipment seem like the right direction.

CHAPTER 2

Overall Design

In this project there are several tools / instruments that will be used for the chemical reaction. These instruments will each be paired with a controller, and those controllers will take commands from a main conductor component. Each of the controllers and the conductor are Raspberry Pis like the one shown in Figure 2.1 below, and depending on the situation, may need a HAT attached above it.

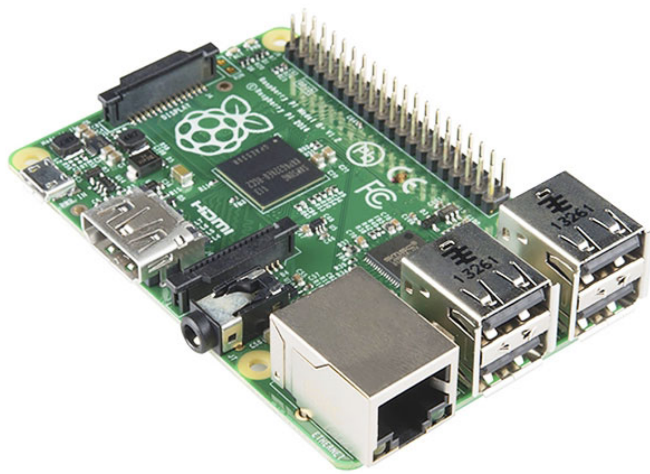

Figure 2.1: Image of Raspberry Pi used for devices

2.1 Conductor

The conductor's purpose is to receive the input from the chemists, translate it using natural language processing, and provide actions for the controllers that follow the inputs that were processed. It looks at evaluated verbs to associate the data and inputs with specific commands related to those actions. To prevent confusion in the device, the chemists will have a specific set of verbs to choose from when inputting instructions into the machine.

It can communicate with the controllers, also Raspberry Pi's, through the client server connection where messages can be sent and received through shared memory between the servers on each controller and the client on the conductor, as shown below in Figure 2.2.

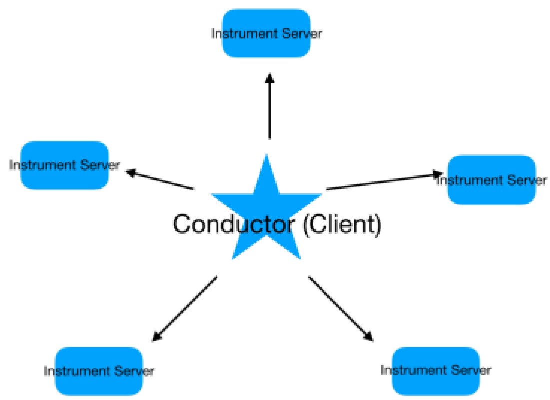

Figure 2.2: Diagram of the client / server connection between the conductor and controllers

Using the shared memory the conductor can also read the state of each machine and if any controllers or sensors report any serious numbers or errors, the conductor is also capable of shutting the machine down. All the controllers need to do is read the shared memory to take each of their commands and perform that action using their own logic between their instruments. This approach can make modifying and solving specific issues with instruments and controllers much easier, along with introducing additional functionalities and instruments.

The commands sent to each of the controllers are decided from the instructions put in from the chemists, which are translated into a more machine useful XML file, based on the verbs and nouns of the instructions. These vary from controller to controller since each one needs different kinds of numbers and information about their instrument.

2.2 Controllers

The controllers are the server Raspberry Pi's that communicate with the client conductor, and with their associated instrument in the machine at the same time. It manages this by running both of the processes independently by forking. This creates a parent and child process. The parent process works with the HAT on the Raspberry Pi and communicates with the attached instrument, while the child process communicates with the client on the conductor. These processes also make use of shared memory on the Raspberry Pi to exchange information.

By calling the function *fork()* on the controller software into a variable, there will be two instances of the program where that specific variable will have different values, but other variables can be shared allowing the transfer of data between them. This can also allow the program to know which process it is in through an if else statement and execute certain functions depending on the process. Splitting these processes can make the machine perform a lot smoother and allow for one process to loop and wait while the other performs other tasks.

2.3 Instruments

Currently for this project, the device has two syringe pumps, a thermocouple, and a stirrer / hotplate. However more instruments will potentially have a use with this machine as mentioned in the list of final instruments from our sponsor AbbVie:

1. Syringe Pumps
2. Stirrer
3. Hotplate
4. Bunsen Burner
5. Heating Mantle
6. Thermocouple
7. Conductance Sensor
8. Reactor Effluent Routing Valve

Other unnamed instruments may also be involved with the machine, such as devices for isolation of products or analytical equipment.

Each of the controllers for the current instruments also need to make use of a HAT so that a direct connection to their respective machines. This can allow software to send commands to the instruments and collect data from them as well.

That functionality is needed so the conductor can keep track of what is happening while it sends the commands to the controllers.

2.4 RS232 HAT

This device can be placed on top of the Raspberry Pi and introduces some more functionality to it, which is exactly what was needed to work with the instruments. It does so using the serial port seen on the lower portion of the HAT in Figure 2.3 below. Since most instruments are capable of connecting to serial ports, this makes introducing new instruments into the project much easier as well. The port contains 9 pins that are used in the transfer of electricity and data between the Raspberry Pi and instrument. However it still requires some additional softwares and libraries to make use of for this project.

Figure 2.3 RS232 HAT on top of a Raspberry Pi

CHAPTER 3

Commands

Each of the instruments requires a certain set of commands that can be given by the conductor through the instruments controller. These can vary from turning the machine on and off, giving more specific actions, or tuning it to specific settings.

3.1 Heat Commands

The stirrer / hotplate has a set of actions provided that we can make happen through the software on the controller when they are connected to each other through the serial port. This device is a combination of two essentially so it can fulfill the requirements for heating and stirring the mixture in one set of commands. There are plenty more commands possible for this device but as of now the commands it will make use of will simply be:

- Set the stirrer speed
- Turn the stirrer on
- Turn the stirrer off
- Set the hotplate temperature

- Turn the hotplate on
- Turn the hotplate off

This makes 6 commands but this data could be represented with just 4 parameters since whether the parts are on or off can be represented by a variable instead of two. These command fields can be represented in a structure so it is easier to contain the data and send it between machines. They are made up of numbers representing the speed and temperature in celsius, and some indicator value as to whether each component should be on or off.

3.2 Syringe Pump Commands

The syringe pump also has a set of commands we can send from the conductor to the controllers to perform some action. These commands can modify the syringe and valve positions. The data relevant to these actions are

- Valve output position
- Valve input position

- Syringe position
- Syringe plunger position
- Execute the command
- Motor speed

The execution command must come after the rest of the commands so the machine can function correctly. These commands can be represented as numbers representing the desired position for all the components and the speed at which to move them. This is needed since reactions can depend on how quickly or slowly a reagent is introduced into the mixture.

3.3 Command Buffer

The commands are given to the controllers by the conductor through the command buffer, essentially a list of values that could represent integers or characters that the controller can decode as long as the structure is as expected. Some additional data can also be appended onto the buffer that can be used to check any errors or restart the process. If the controller does find any errors in the command buffer it will know and throw an error and avoid those commands. For example it can check how many bits were sent or whether a number sent along with the commands matches what the specific controller expects. The controller can also send information about the command buffer back to the conductor such as there being an error or if it is still processing commands or if there are too many commands waiting to be processed.

CHAPTER 4

Conclusion

Overall, this machine seems extremely complex and sophisticated, but definitely worth the effort to get functioning for the chemists. I feel like the project went well for everyone and some great progress was made, although I wish I was able to be of more help and had more experience working with Raspberry Pi's on this kind of project going into it. I certainly learned a lot about them throughout this project and how multiple can work together to delegate functions with outside instruments. Hopefully the group that continues this work will be able to make even more progress with taking this machine to its final goal state. I will be looking forward to hearing about new discoveries from this project, and from teams all around the world investing their time into researching this field.

References

[ML] Machine Learning in Computer Aided Synthesis Planning

https://pubs.acs.org/doi/abs/10.1021/acs.accounts.8b00087

[ET] Enabling Technologies for the Future of Chemical Synthesis

https://pubs.acs.org/doi/abs/10.1021/acscentsci.6b00015