Final Assembly Project Documentation

HYDAC International

Mirko Glavan

Summer 2024

# **Day 1 - 5/14/2024**

7:30AM - 4:00PM

HYDAC - Denver

Mirko Glavan

## Daily Project Agenda

1. *Synthesize* a problem statement
2. *Generate* project specifications and measures of success
3. *Utilize* project management tools
4. *Identify* and *Diagram* subprocesses of the project
   1. *Identify* applicable programming, data management, physical, or mathematical concepts
5. *Begin* brainstorming alternative and iterative solutions for step one
6. *Communicate* daily results to both upper management, IT, and employees on assembly floor

## I. Problem Description

The HYDAC [Cooling - Denver] manufacturing facility would like to implement digitalization in its final assembly process. During this process, employees working on the plant floor use, in cohesion with the SAP database, a combination of in-house manufactured heat exchangers, subcontracted parts, and purchased sub-assemblies to put together heat exchanger systems that meet a customer’s needs. Currently, methodical instructions are sparse, as employees’ primary form of assembly instructions are [paper-copy] exploded views of engineering drawings. After the conclusion of the Final Assembly Project, HYDAC visions the final assembly process can be guided by a digital form of instructions that autonomously observes prompts from the assembler to progress through a set of in-house generated, customized work instructions for each of the possible final assemblies in catalog.

The execution of the project has already been broken up into three broad sub-tasks:

Firstly, operators or final assembly leads will use handheld electronics to submit images at various points in the assembly process, where they can be [digitally] annotated and uploaded to a file management system. This file management system must organize the images in respective folders, by assembly number and sector (ex: JLG assembly no. 7702559 will have address J:\\Production\Work instructions\\JLG\\7702559).

Secondly, after the operator has submitted a sufficient amount of pictures to coherently break down the final assembly process, the images and annotations will be used to either manually, or with image processing and text identification, autonomously generate non-redundant, fluid work instructions. It is noted that the instructions cannot be oversimplified, excessive, or redundant, as the digital instructions pose the possibility of becoming an unnecessary burden if interrupting the natural flow of final assembly. These instructions must be then uploaded to the same folder (J:\\Production\Work instructions\JLG\7702559) to be accessed during final implementation in step three.

Thirdly, and finally, with either a barcode scan, or an interface interaction, the operators on the assembly floor will be able to pull up the generated work instructions on a digital format, such as a monitor, TV, or tablet. Depending on the scale of implementation, time remaining, the preferences of operators, and the requests of supervisors, the work instructions will be semi-interactive. Therefore, a relatively simplistic solution to the project will be to collaborate with operators to create [natural] step-by-step instructions for final assemblies by uploading images they capture, synthesizing them into a “slideshow”-esque series of work instructions, and finally return them to the assembly floor for access via TV. However, intermediate steps and progression within the steps can be controlled by a mixed use of human interactions with an interface (click, button, arrow key, etc) and sensor inputs (object detection, bluetooth tool responses, etc) the latter of which would require methods added to a central class for as many, or few, aspects of assembly that are to be controlled via programmed sensor responses.

## II. Problem Statement

*Create* a database of digital assembly instructions to *replace* the physical drawings operators use in the final assembly process. *Given* annotated images, *build* a network that can store the images, *synthesize* them into work instructions, *return* a new digital guide to the assembly floor, and *implement* semi-autonomous processes to navigate along steps in the new guide.

## III. Project Specifications

A needs finding table was generated to summarize the general requirements of the Final Assembly Project into measurable criteria. See figure 1 below.

Figure 1

| Project Requirement | Parameter | Technical Requirement | Target Value |
| --- | --- | --- | --- |
| Image Capturing is Accessible | Time from Annotation to Reception into Image Cloud Storage System (sec) | 10 | 5 |
| Organize Operator’s Annotated Images | Folders per Assembly (#) | 1 | 1 |
| Images are Synthesized into Digitized Instructions | Ratio of Images to Steps  (nImages : nSteps) | 1:1 | 10:9 |
| Instructions are Effective | Ratio of Assembly Time Before to new Assembly Time (tOld : tNew) | 1:1 | 10:9 |
| Instructions are Accessible | Time to Open Correct Instruction File (sec) | 30 | 10 |
| Implementation allows for Addition of Semi-Autotomized Additions to the Network | Number of [compatible] devices that instructions can call upon to give control to the digital prompter (#) | 1 | 3 |
| Implemented Reasonably | Total Cost of Startup and Sustenance (USD) | Less than $1500 | $0 |
| Implemented Completely | Deadline to Reach Final Step (Date MM/DD/YY) | 08/09/24 | 07/12/24 |
| Project is Able to be Continued after Summer Internship Timeline | First Step Needed Rework after Summer 2024 (#) | Step 2 | Step 3 |

*Needs Finding Table*

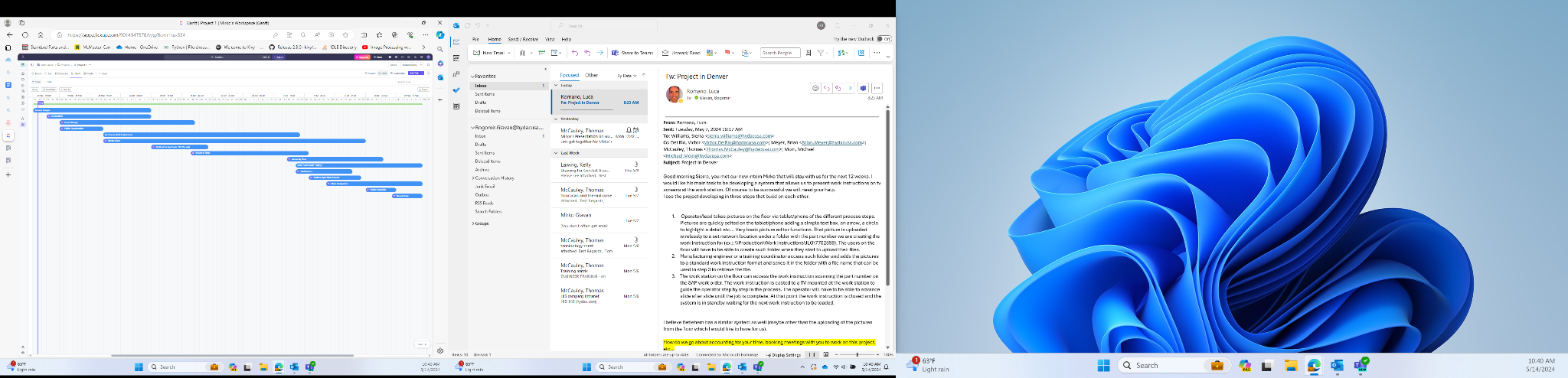
The needs finding table shown above in figure 1 represents the breakdown of each of the requirements the project must address. In the final column, the target values, estimated parameters for success were identified, which will be elaborated on in *Measures of Success*. This section will also discuss rationale for selecting these target values for success, such as for costs and measures of effectiveness without use of a budget.

## IV. Measures of Success

The general measures of success can be identified by meeting the target values in the final column of Figure 1. This ensures the basic components of the project are completed and are functional in a network with the infrastructure that exists. Additional measures of success, perhaps less quantifiable, such as “effectiveness” represents the goal of the project to not just work, but create some positive change within Final Assembly, measurably in reducing production time (which can be measured at conclusion). This was quantified by the ratio of assembly time from pre- to post- implementation (ideally less than 1), the amount of autonomous devices implemented, and a step rating of compatibility for future work. Lastly, by making use of open-source programming languages (Python) and pre-purchased devices, the project is aimed to be completed at a price of $0. While some larger scales of implementation will require additional funding, that will be addressed nearer the conclusion of the project.

## V. Gantt Chart

A Gantt Chart was generated for project management. See figure 2 on the next page.

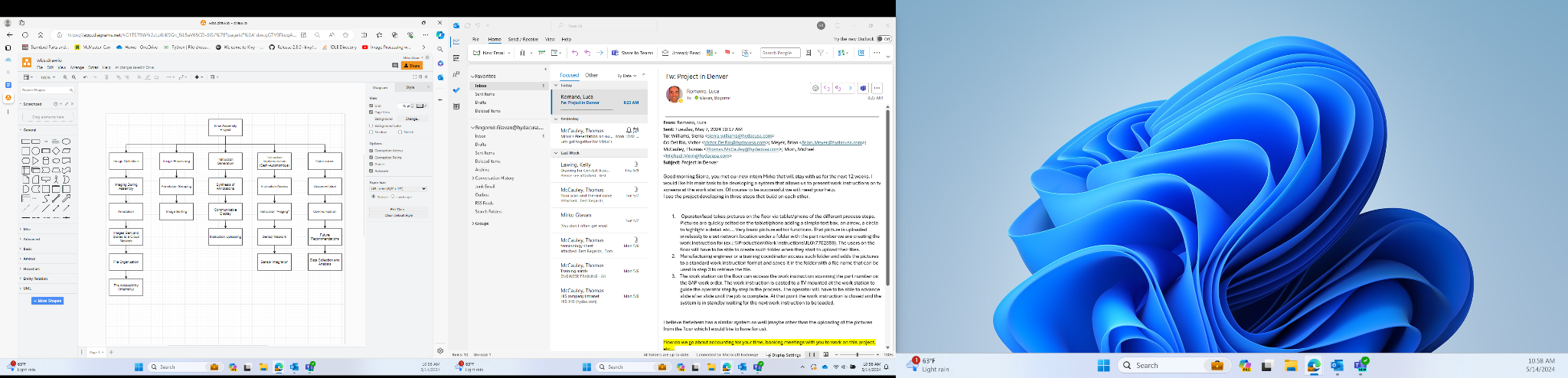
Figure 2

*Gantt Chart Accessible Also at:* [*Gantt Chart Link [External]*](https://app.clickup.com/9014347578/v/li/901403351120)

## VI. Work Breakdown Structure

A Work Breakdown Structure was generated for project management. See figure 3 below.

Figure 3

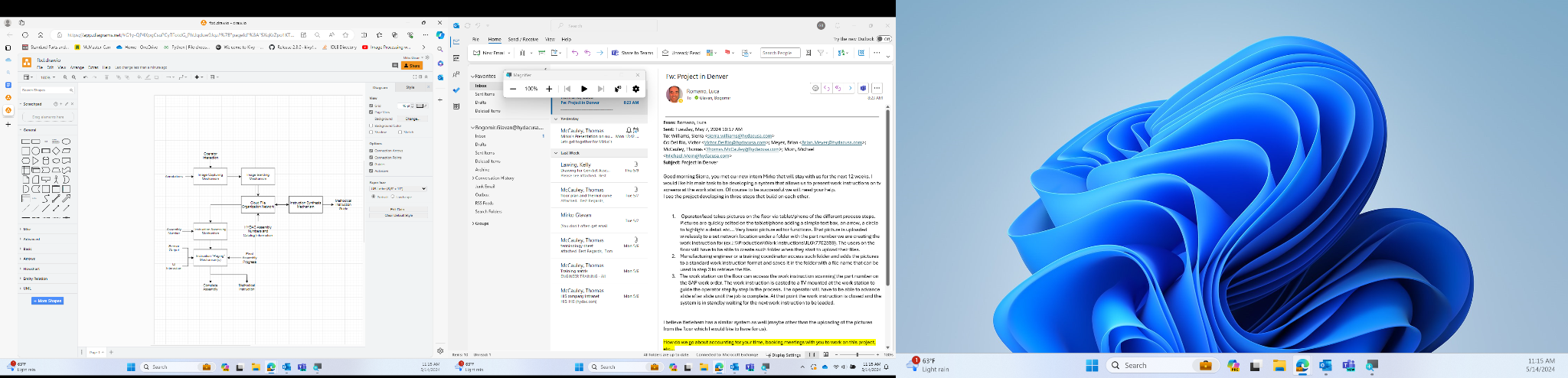


*Work Breakdown Structure Accessible Also at:* [*Work Breakdown Structure Link [External]*](https://drive.google.com/file/d/1TS79WYr2zLp0rK9Gn_5iL5aV65CD-3IS/view?usp=sharing)

Figures 2 and 3 represent project management tools to organize time spent and general anticipation for flow of the project. While only one person is working on the project, it will be completed in a linear fashion, moving onto another subtask when one is complete. It is important to note that some processes will be done alongside staff members such as IT for some cloud and file management and the operators on the Final Assembly floor for most of the communication and design intent.

## VII. Functional Block Diagram

Figure 4



*Functional Block Diagram Accessible Also at:* [*Functional Block Diagram Link [External]*](https://drive.google.com/file/d/1y-QP4XpgCsuPCyTFotdG_PhUqduw9Jqz/view?usp=sharing)

The functional block diagram shown above in figure 4 represents a general understanding of how an instructional network should operate to take the pictures from the operator and turn them into instructions, that can then be used alongside sensors to monitor various aspects of the assembly, or rather simply just provide an alternative to exploded drawings.

## VIII. Engineering Topics of Interest

**Python Programming**

The project will rely very heavily on programming, and will implement primarily Python due to the fact that it is open-source, has reasonable computing speed (few higher-order computations for this project), and abundance of modules for creating GUIs and accessing OSs. A different Python file will likely be used at every junction of the project, to generate GUIs, access files, and possibly use Image Detection or Learning Processes to synthesize images into instructions as well as implement RaspberryPi with various sensors on the back end.

**Linear Algebra**

If Machine Learning is to be used for the image synthesis (determined if the number of images becomes too large for manual sorting), a CNN will need to be made. While this can be done relatively easily and without much knowledge using YOLO modules and roboflow, it leans heavily on mapping and manipulation of kernels to identify text, annotations, and parts in frames. If future work wishes to generate a CNN by hand, this would require extensive knowledge in Linear Algebra (and time) to generate such matrices experimentally.

**Engineering Data Management (PDM)**

Product Data Management (PDM) systems will be leaned on for the first steps of the project along with its child concept, Product Lifecycle Management (PLM). This will be kept in mind as far as file security and data loss when choosing a method for cloud storage of all the images and presentations. Keeping PDM in mind, the project will likely need to add a hard drive of all the images and synthesized files to the steps it must take. In the event of an emergency crashing, all of the .py or .kv or roboflow workplaces must be saved in a similar fashion to allow for backups and the continuation of the project after August 2024.

**Manufacturing Physical Applications**

Many physical applications must be considered when selecting which sensors to use on the back end of the project, to “page” between steps of the instructions. Such as monitoring torque, identifying distances, or calculating other kinematic values will be of interest when selecting sensors such as Gyroscopes, Ultrasonic Sensors, etc. Circuitry and physics applications outside kinematics, rotation, and basic object tracking will likely not be necessary for the project outside wiring such as a RaspberryPi (GND, DNC, GPIOn).

**Statistical Analysis**

In order to draw conclusions on the effectiveness of the design, or determine if a given feature is necessary, hypothesis testing on the time standard will be among, if not the best, method of drawing quantifiable inferences. Interfaces and sensors can be used to collect data passively, of which a hypothesis test can be done independently, by hand. All of which should be one-sample t-tests that either test a population mean or a mean difference.

## IX. Step One Brainstorming

Goal: Capture images with annotations from operators and organize them in such a way that they can be easily accessed for step two, or for any individual. Images will likely be taken and annotated on a tablet, so files move from a “camera roll” to an accessible project directory or a repository folder.

**Sub-Tasks:**

1. A method for annotation, typically a builtin functionality of a device
2. A method for capturing images off a camera roll
3. A method for storing of images in a series of folders organized under one main project folder
   1. ex: J:\\Production\Work instructions\JLG\7702559
4. A method for ensuring the folder can easily be accessed, either by a physical hard drive, online repository, local storage on a C drive, or PDM file storage software.
5. Testing and debugging to ensure consistency, file security, and effectiveness

**Solutions:**

Below are general processes that led to the brainstorming of many possible solutions to step one.

1. Most tablets and handheld devices have functionality that allows for annotation, most of which will keep the file still stored as a .jpg file or .jpeg on a local camera roll. It can be assumed that no matter the case, the solution must originate with a .jpg/.jpeg/.png or any other common image file on a camera roll.
2. The methods for moving an image can either be done manually, or by accessing the operating system of the device computationally. The most realistic of these are:
   1. Emailing images to the same host
   2. Uploading images to a hard drive
   3. Using OS commands in a compatible software to save images to an online directory or repository
3. The methods for sorting images in folders can either be done manually, or by manipulating the repository. This leaves:
   1. Manual moving of files
   2. DropBox for automatic sorting
   3. User Input to manipulate the online repository in variable ways
4. To ensure the folder is accessible, it needs to be owned by everyone involved in the project. In order from most accessible to least:
   1. Common Online Repository
   2. Internal Storage on company C drive
   3. Hard Drive storage
5. Given that computational methods are selected, manipulation must be done with the following modules that need to be added to pip via the Command Prompt, assuming Python3.11 is already installed with IDLE:
   1. Kivy
   2. Matplotlib, Pandas, NumPy
   3. os (or other method of compatible OS access)
   4. shutil (or other method of compatible OS access)
   5. github (or other method of repository access)
      1. [StackOverflow | .png to github](https://stackoverflow.com/questions/72668275/how-to-upload-an-image-file-to-github-using-pygithub)

**Alternatives and Iterations:**

Below are the three most likely and viable solutions to step one of the project.

1. Image -> Annotation -> Upload to Hard Drive -> Hard Drive Passed to All
2. Image -> Annotation -> File User Inputs (Python) -> File In Folder -> Folder in Online Repository -> Repository for Access (GitHub)
3. Image -> Annotation -> Email Attachment -> Download -> File in Folder -> Shared Windows Folder

In order to decide which method would be the most practical, a decision matrix was generated to quantify each of the three feasible alternatives. The matrix is shown on the next page in figure 5.

Figure 5

|  | Hard Drive | Python File Acceptors / Sorters | Email | Weights |
| --- | --- | --- | --- | --- |
| Manual Interaction With Files  (% of Process) | All (100) | None (0) | All (100) | 10 (-) |
| Processing Time | at least 20 seconds | at least 5 seconds | At least 90 seconds | 7 (-) |
| Design Time | None (0) | at least 16 hours | None (0) | 2 (-) |
| Operator’s n Step Process | 3 | 1 | 5 | 5 (-) |
| rel. File Accessibility | 0 | 10 | 10 | 7 (+) |
| Total | 9.44 | 27.61 | 9 |  |

*Decision Matrix (QFD)*

Given by the decision matrix, using Python to move files to a GitHub repository, for later access will be the preferred method for getting annotated images off of tablet camera rolls. By using GitHub, this will allow for compatibility with later programs that will draw from the repository, or add to it when instruction synthesis comes around. It can be noted that a system such as a hard drive or file storage in a C drive is not thrown out, as files can be downloaded off the repository and organized however one wishes on a local HYDAC monitor or hard drive. The GitHub repository will be structured in an organized, user friendly format to enable accessibility of others and the future programs. In addition a *Naming Convention* will be made for all files during the project. See figure 6 below.

Figure 6



*Repository and Project Naming Convention*

By figure 6, each file address will be located by the following name convention, and the file name will be the address with underscores in place of backslashes. For example a JLG assembly number 7702559:

Repository Address: <GitHub Repository>:\\WorkInstructions\JLG\7702559\Images\<filename>

Filename: WorkInstructions\_JLG\_7702559\_Image< x >

General Convention: WorkInstructions\_SectorID\_AssemblyNO\_FileTypeFileNO

Note: in GitHub folders are “branches” as file storage in a repository behaves slightly differently than in a local drive, but file navigation and interfacing will appear the same. The only major difference is in programming calls to “branches” as opposed to OS commands.

## X. Communication of Results

Completed at the end of the daily project work period

## XI. Daily Conclusions

After the first investigation in step one of the project, additional research will be done to determine how to navigate GitHub repositories. A GitHub private repository was reserved with the name WorkInstructions for storage of project documents. The main challenge that will be present, aside from learning to use GitHub modules and navigate the interface, is to ensure that the tablets will be compatible with OS commands. If the tablets have different operating systems, the program will be revised to a new module, or a new method of file transfer will be selected. By creating a naming convention and a general framework for completing step one of three for the project, the goals set by the project management tools will ideally be met. Lastly, while the GitHub repository will not be for local storage, files can still be stored in branches as desired, and accessed in a similar fashion. The GitHub module and interface will allow for much more compatible programs and networks of devices, especially as complexity is compounded in step three and beyond. Finally, by the brainstormed applications to engineering topics, a central storage system alongside documentation will aid in understanding the methodology used during the summer of 2024, primarily when such topics become prevalent.