Group 7 Final Report

Introduction to Industrial Informatics

Carlos Guttormsen, 245621

Qi Yuan, 267957

Table of Contents

1. Introduction

This is the final report for the course Introduction to Industrial Informatics group work. The objective was to develop a knowledge-driven monitoring and orchestration system for the FASTory assembly line but later as the course progressed we decided to use the FASTory-line simulator because of its ease of use since otherwise this project could have been too time consuming. Simulator or not, this project gave us the opportunity to develop our system engineering skills as we systematically developed the FASTory-line control system.

In this report we have put together all the diagrams that we have developed and refined through the time of this course and we also describe our thought processes behind each of them and behind the project generally.

2. System Vision and Financial Analysis

In this chapter we cover our project work's system vision and present a financial analysis which consists of a risk and cost/benefit analysis.

2.1 System Vision

In this project, we are going to develop monitoring and supervisory control system of the cellphone production line in Tampere University of Technology, FAST-Lab. In the process of monitoring the product and production line, we limit our system vision to FAST-lab simulator firstly and then extends to the whole production line which should be managed for following months in the inception phase of our project. First, the planning of project should be divided into several parts which could help us to have more clear idea about what we are going to do. In this system vision, there are four sub-branches: project management, business analysis, environment and requirement and design as well.

In the project management, we should know the objectives of the project and make some assumptions and assign group-members for different sub-tasks. The schedule should be carried out precisely to finish the project on time. This is the initial part but also the most important part in this project as it is the direction of where we should go.

For the business analysis, we need to evaluate costs for initializing the project such as the costs for the equipment, salary for the operator and manager, maintenance for robots and conveyor, and profits for the shareholders. We have to consider some issues like our possible costs, limited budget maybe and prospective profits. As the business model is interwoven with requirements, defining a good business model can cover all our needs and requirements of the project.

For the environment model, we should execute our project in the production line of TUT and configuration of production line must be well organized to ensure the assembly line running efficiently. The User interface should be well designed which is helpful to present useful information to the customers.

When designing the system which is an iterative task, we create the process outline that shows the feasibility of the project. Of course, we also need to consider the potential risks as

well. The principle of design is to make full use of each object and reach our goal best and try to lower the risk. The main costs may come from those aspects such as wage for the crew and waste of scrap and maintenance. Therefore, our priority is to control the budget and fully use our material which lowering our costs.

To better initialize our project, the first thing we should do is that we must get familiar with the surrounding of production line and avoid causing some unnecessary mistakes or maybe don't break any equipment in case of increasing our costs. What's more, we must know the instructions of some machines like Robots and conveyor and so on. For instance, we need to calibrate the cellphone production line before we do the project. Our target is to master in the manipulation of all machinery and reduce the lifecycle of product. In order to finish the project perfectly, we would start to work with simulator. Once the coding is running is perfect on the simulator, we then apply it in production line.

In creating the system vision, we should focus on the assembly line and the main components of this system such as robot, convey and RTUs. The line consists of two conveyors; main and bypass. The main conveyor is used if the pallet requires service from the robot of each workstation. Meanwhile, the bypass is used if the work cell is in busy state to bypass the pallet to the next work cell. The factory line uses SONY SCARA robots for production. Each robot is represented as an RTU in the line. The robots are programed with specific tasks. The FASTory is equipped with INICO S1000 Remote Terminal Units (RTUs). INICO S1000 is a programmable RTU device which offers process control capabilities and all the data will be selected and transferred to here for final analysis and decision.

When running the assemble line in reality, the suitable components needs to be found respectively. For instance, when we are running the conveyor, which speed is the best for transport of pallet and would not cause congestion on the line. The type and shape of the pallet is maximally flexible for all three products. Those doubts and question must be unveiled when executing the system and get the helpful data for project.

Providing all this architecture (SOA) is impossible without implementing a decent infrastructure for optimization, positioning using sensors and safety switches, resources for maintenance and staff for controlling the whole process (including quality control of the product going out from the manufacturing cell).

To fulfill the task of monitoring we should collect all the data and decide what data to use for running the production line. For instance, we use INICO S1000 to communicate through web services. It transfers all the data to a web server which can be accessed and processed on the WebStorm, so the needed information is derived for monitoring the work cell. In our case we assume that the manager has access to local server for monitoring where the location of products is under processing. Also, we assume the product we mean to monitor in our working cell is a frame of a cell phone that comes in 3 different types of models. We need to closely monitor and assign operators for accomplishing this goal.

2.2 Financial Analysis

To understand the possible project related risks the following risk analysis chart presented in the figure 1 was put together.

Risk Description	Potential Impact on Project (High, Medium, Low)	Likelihood of occurrence (High, Medium, Low)	Difficulty of timely Anticipation (High, Medium, Low)	Overall Threat (High, Medium, Low)
Critical team member not available	High	Low	Medium	Medium
Technical malfunctions in development	Medium	Low	Hard	Low
Project cost becomes too high	High	Low	Low	Low
Changing project requirements	Medium	Low	Hard	Low

Figure 1. Risk analysis chart

In this figure the possible risks were identified and their overall threat to the project were analyzed. This was done by measuring the risk's potential impact on the project, likelihood of occurrence and difficulty of timely anticipation. All these factors were measured on a scale of low to high.

Lastly, we defined the project's costs and benefits and put together a cost/benefit analysis chart. This is illustrated in the following figure 2.

	CY	CY+1	CY+2	CY+3	CY+4	CY+5
Costs						
Purchase of equipment	185,000.00 €	200.00 €	100.00 €	50.00 €	50.00 €	50.00 €
Labor	21,000.00 €	22,000.00 €	23,000.00 €	24,000.00 €	25,000.00 €	26,000.00 €
Maintenance	1,000.00 €	1,500.00 €	1,500.00 €	1,500.00 €	1,500.00 €	1,500.00 €
Energy	2,500.00 €	2,500.00 €	2,500.00 €	2,500.00 €	2,500.00 €	2,500.00 €
Software and other tools	7,500.00 €	1,200.00 €	500.00 €	500.00 €	500.00 €	500.00 €

	CY	CY+1	CY+2	CY+3	CY+4	CY+5
Benefits						
Sales of Nokia Phones model A	16,000.00 €	55,000.00 €	50,000.00 €	60,000.00 €	62,000.00 €	65,000.00 €
Sales of Nokia Phones model B	10,000.00 €	20,000.00 €	75,000.00 €	82,000.00 €	98,000.00 €	180,000.00 €
Sales of Nokia Phones model C	11,500.00 €	35,000.00 €	50,000.00 €	45,000.00 €	50,000.00 €	70,000.00 €

	CY	CY+1	CY+2	CY+3	CY+4	CY+5	Total
Cost/Benefit analysis							
Value of benefits	37,500.00 €	110,000.00 €	175,000.00 €	187,000.00 €	210,000.00 €	315,000.00 €	
Discount factor (7,5%)	1.0000	0.9500	0.9000	0.8500	0.8000	0.7500	
Present value of benefits	34,687.50 €	102,162.50 €	163,187.50 €	175,078.75 €	197,400.00 €	297,281.25 €	969,797.50 €
Value of costs	(217,000.00 €)	(27,400.00 €)	(27,600.00 €)	(28,550.00 €)	(29,550.00 €)	(30,550.00 €)	
Discount factor (7,5%)	1.0000	0.9500	0.9000	0.8500	0.8000	0.7500	
Present value of costs	(200,725.00 €)	(25,447.75 €)	(25,737.00 €)	(26,729.94 €)	(27,777.00 €)	(28,831.56 €)	(335,248.25 €)
PV of net benefits & costs	**(166,037.50 €)**	**76,714.75 €**	**137,450.50 €**	**148,348.81 €**	**169,623.00 €**	**268,449.69 €**	
Cumulative NPV	**(166,037.50 €)**	**(89,322.75 €)**	**48,127.75 €**	**196,476.56 €**	**366,099.56 €**	**634,549.25 €**	

Payback period	**2 years and 237 days**
5-year return of investment	**189.28%**

Current Year = CY	
Initial discount factor (%) =	**7.5**

Figure 2. Cost/Benefit analysis chart

To make the cost/benefit analysis clearer it was done in three parts. First the different costs were listed, and this was followed by listing the different benefits. Finally, the sums of both the costs and the benefits were listed in the third chart. The present values for these sums were calculated using a dynamic discount factor which initially was set to 7,5%. Using the cumulative net profit values, we could calculate the payback period which came out to be two years and 237 days. The five-year return of investment was calculated by dividing the final NPV with the total costs and it turned out to be 189,28%.

3. Unified Process Diagram

For this project we created a unified process diagram to tie the different aspects of our project work together. This is a representation of how our imaginary business could work. The unified process diagram is illustrated in the figure 3 below.

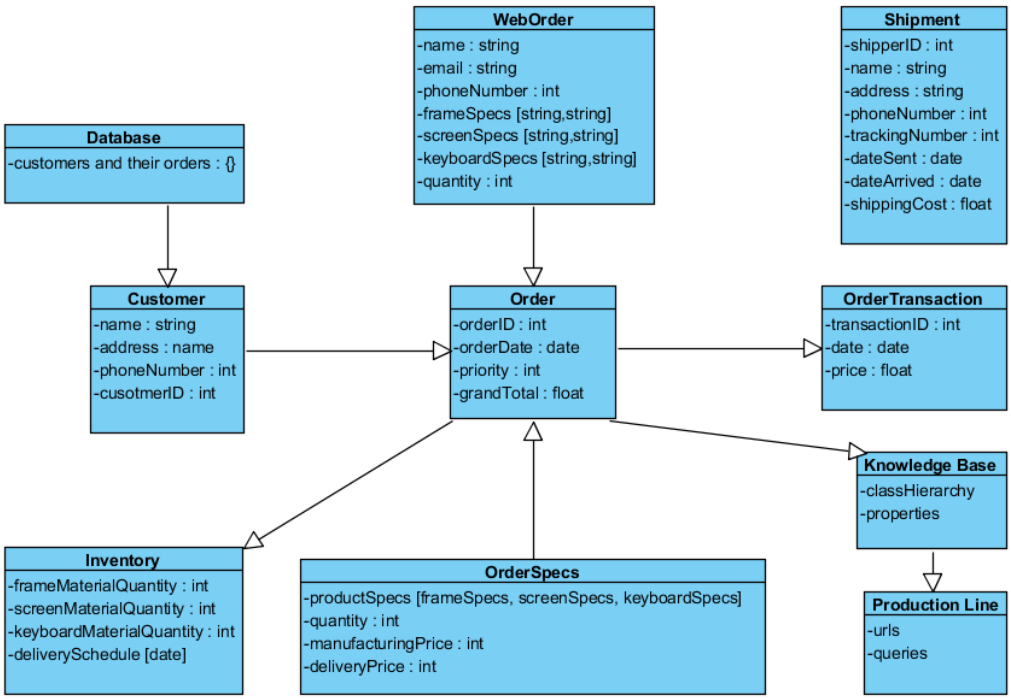

Figure 3. Unified Process Diagram

In this diagram we have represented how different parts in our project could be dependent of each other in order to process a customer's order from start to finish.

4. Use Case Diagram

In this project we went over our use cases. We took into consideration the customer's role and how it would get its order placed. The main idea is to illustrate how a customer's order gets processed in our system. With these use cases we created a use case diagram which is shown in the figure 4 below.

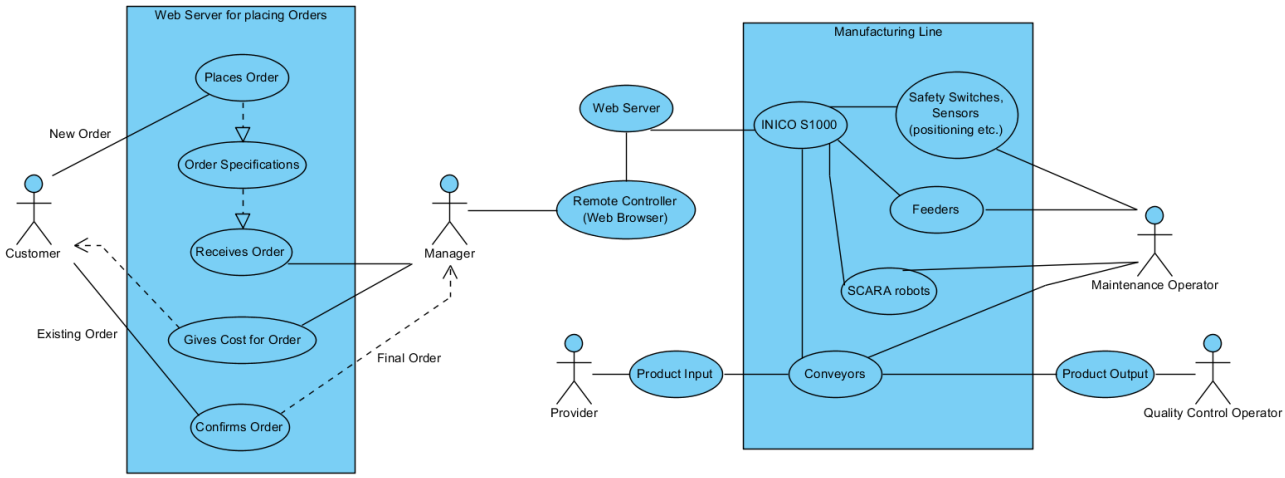

Figure 4. Use Case Diagram

This use case diagram consists of two systems; the web server in which the customer places its order and the manufacturing line which then produces the placed order accordingly. As shown in the UCD, the manager acts as the middleman between the two systems.

5. User Interface

The UI we developed for our project is illustrated in the figure 5 below. It has the fields for obtaining buyer information and of course the specifications for the buyer's order.

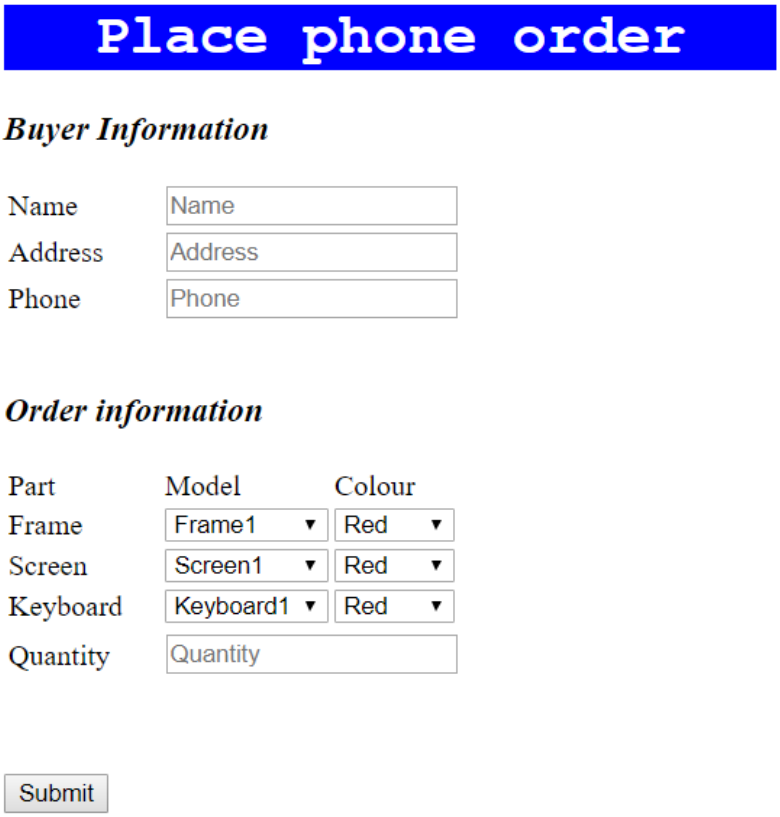

Figure 5. Web Server UI

The order information fields are the main part of the UI. According to the instructions for this project, the phone consists of three different parts that can be altered; the frame, the screen and the keyboard. Each part can be specified to be of a certain color or of a certain model. Finally, the last part of the order is the quantity. After all the information has been filled the customer can click the "Submit" button. At this instance we fetch all the information and store it in our JSON type database for further usage.

The main code can be found in the “iii_project.js” file in which we created a local server and connected it with the UI and subscribed to all the events in the simulator. We can get the response from the simulator when the event is triggered. Also, we can get the information about the status of any conveyor but the step we need to do is to connect the knowledge base with our server so that we can do the query properly. The specific explanations can be found in the comments of the “.js” file.

6. Activity Diagram and System Architecture

In this chapter we present the activity diagram and the system architecture we developed for the project. The activity diagram which demonstrates the workflow of the system is illustrated in the figure 6 and the system architecture in the figure 7 below.

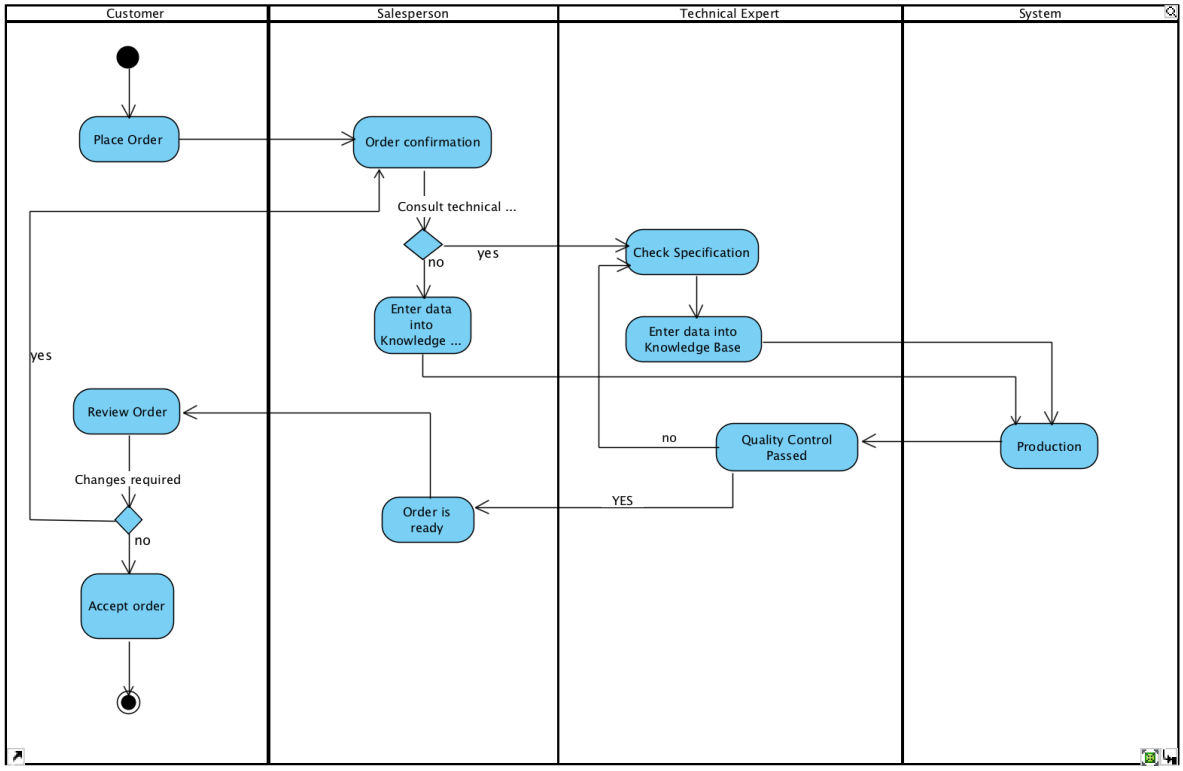

Figure 6. Activity Diagram

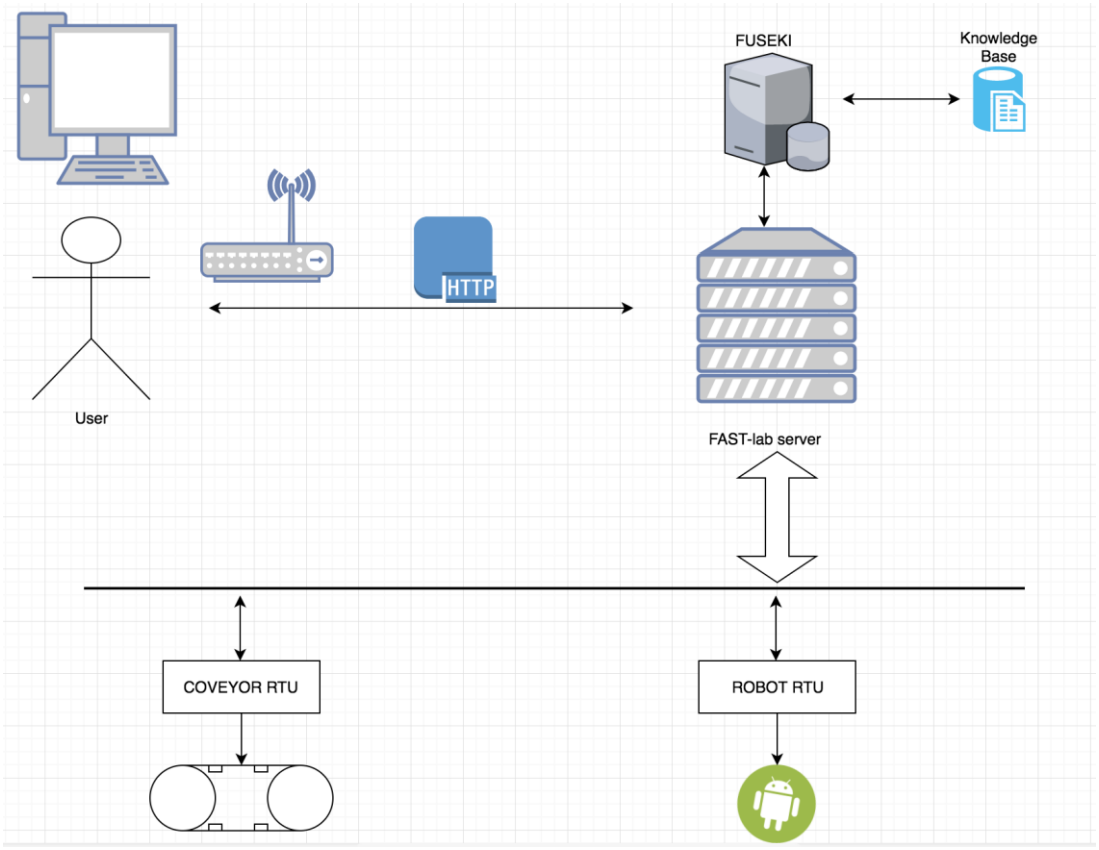

Figure 7. System Architecture

In these two diagrams, we have represented the workflow of the whole process of our project going on. Initially, the customers place the order on our UI, then it goes to our database. Our salesperson checks the order and confirms it then. If the specification of products has already existed in the knowledge base, we proceed it to the production line. If not, our technical experts need to do some changes in our production line. When it goes into the production line, the factory is rather automated and could finish the task properly. After production is done, it proceeds to quality control progress. If the products are qualified, our sales department would inform the customers to review the order again. If the product does not pass quality control, we must check the problem and put into production line once more.

If customers are satisfied with products, we will do the last step-delivery, otherwise, we might negotiate till both are satisfied.

7. Deployment Diagram

The deployment diagram we developed for our project is illustrated in the figure 8 below. It has the information that shows how different mechanism interacts.

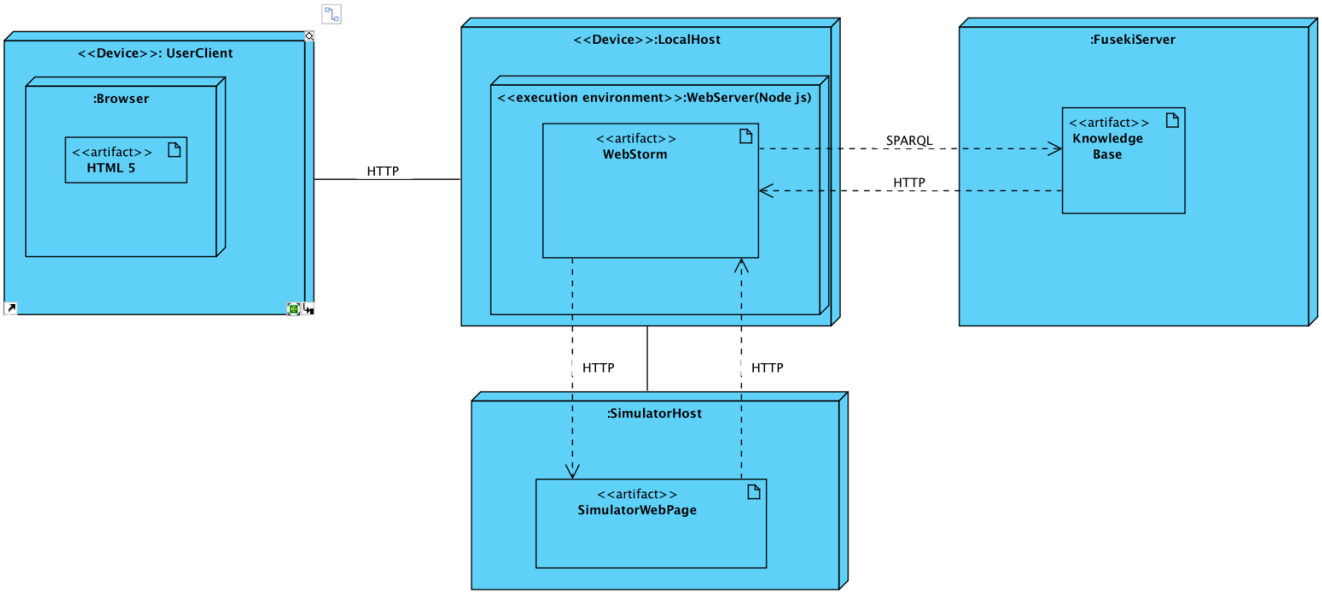

Figure 8. Deployment Diagram

On the customer side, the server runs on its device and they use a browser like Firefox or Google Chrome visiting our website. The page will be presented in HTML. The client-server communicates with our server by HTTP protocol. When the server receives the order, it will parse the important information and send it to the knowledge base. The localhost interacts with Fuseki server with SPARQL query via HTTP protocol. Also, the localhost connects with each RTU from production line by HTTP.

Each time, when the conveyor or robot moves, it communicates with the knowledge base to call the next action. Those three servers (Fuseki server, localhost, RTUs) constantly interact to ensure best options for pallet movements.

8. Sequence Diagram

The sequence diagram we refined for our system is focused on the detailed workflow of our whole project. This diagram shows the principle of how the program we created works. It is the backbone of the logic we developed. The sequence diagram is presented in the figure 9 below.

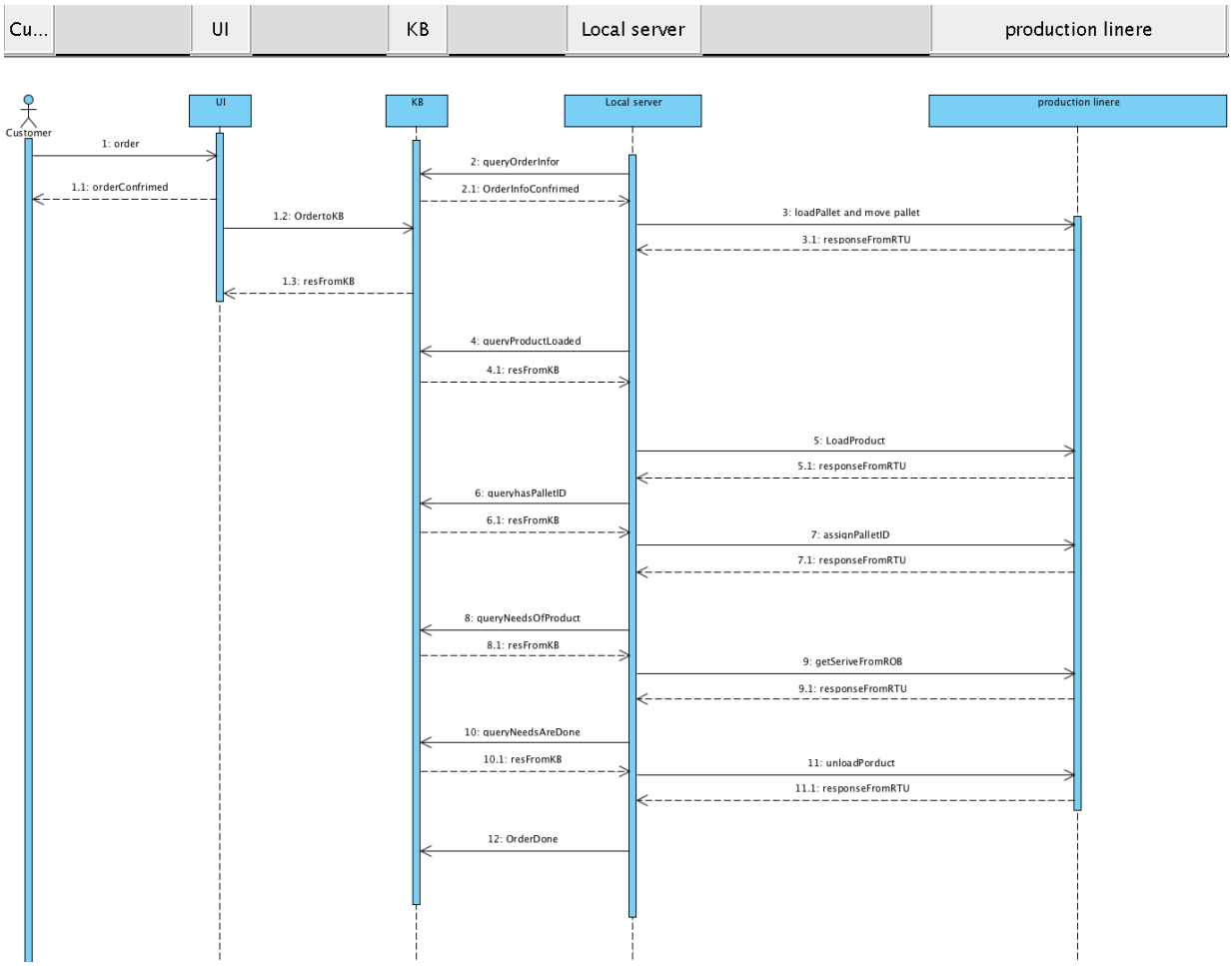

Figure 9. Sequence Diagram

In this diagram, it shows the sequence of each step about when it should be happening. The customer places an order via our UI. Once the order is confirmed, the specification of the products will be inserted into the knowledge base. From this step, the Fuseki server starts to interact with localhost all the time. Localhost makes a query and gets the response from the knowledge base. The commands for RTUs is encompassed in each response.

When the conveyor moves, the localhost knows the corresponding events. According to the events, the localhost will communicate with Fuseki and then command the RTUs.

The whole process on the production line could be illustrated like this: first, the loading robot loads a pallet and keep the pallet moving on. If the pallet has no paper, it goes to all bypass conveyors to load a paper as soon as possible. Once it gets a paper, it moves to get service from a robot. At zone 1 of each workstation, it checks the needs of paper to ensure if the pallet should go into the workstation or bypass conveyor. If the need matches the service, it should move into zone 3. Before it proceeds to do a movement, the pallet checks the status of neighbors. A pallet will not move until one of the neighbors is free. After all the needs are done for one pallet, it goes to workstation 1, zone 3 where it gets unloading service.

9. Knowledge Base

In this chapter we present the knowledge base for our project work using the tool Olingvo. The collection of the class hierarchy and the properties used in our knowledge base are represented in figure 10 below. Our knowledge base consists of all the parts of the manufacturing line and of our customers and their orders. By incorporating the customer to our knowledge base, we can track its order requirements and when they are met. This way we get the information of completed orders in real time while the manufacturing line is running.

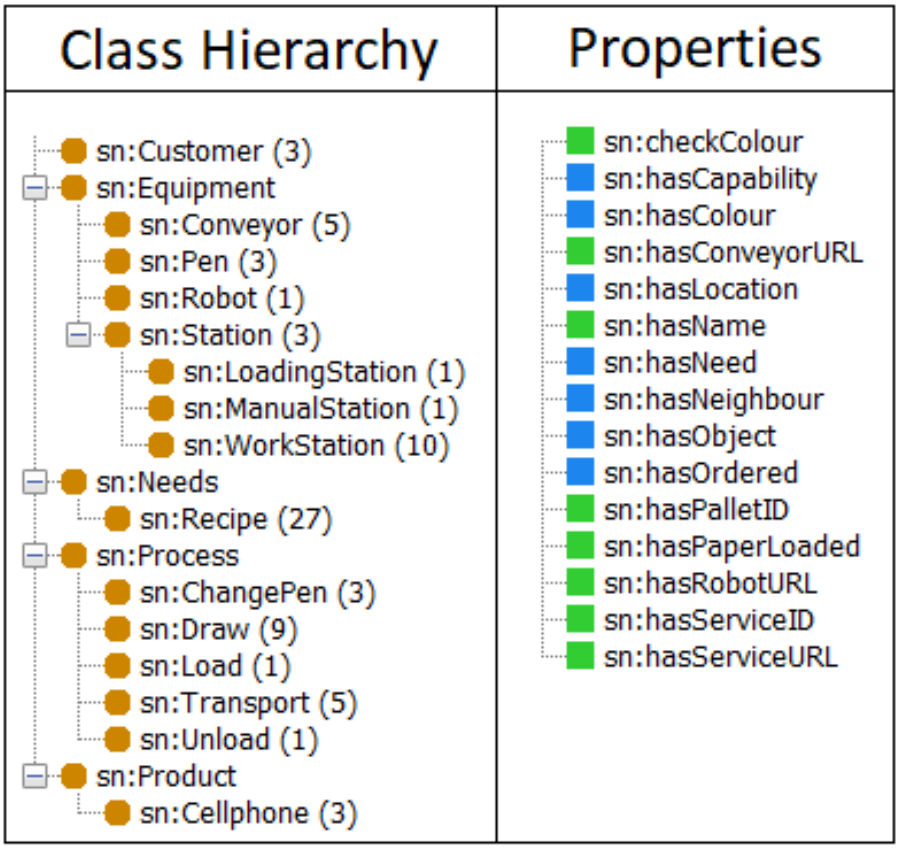

Figure 10. Representation of the Knowledge Base

The first level can be thought to contain different pieces of equipment that show up in more than one station which brings us to the second level which consists of the mentioned stations. The connection between these two levels is formed with the property hasObject. For instance, each type of station has a certain number of conveyors but only workstations have pens. The needs class holds within all the possible recipes that can be assigned to a product. The quantity of different recipes in this project is 27 because of the options to have three different parts, models and colors. The process class consists of all the different processes that can be achieved by the robots and the conveyors and the product class holds within all the ordered cellphones which have some needs assigned to them.

Each class can have various individuals and these individuals can have various properties and with these class-property relations, different queries can be formed in our program to extract specific information that is needed to run the manufacturing line. The two different colors (green and blue) in the properties section of the figure 10 describe different types of class-property relations. The properties that are marked with the blue color have a range in which they operate in the knowledge base. The ones marked with the green color on the other hand don't have a range in the knowledge base, but a value assigned to them. Depending on the nature of the property we have used values of Boolean, String or Integer type.

How the different properties we have defined are used are clarified in the following table 1. In this table we have collected all the properties we defined and explained what they return when they are queried.

Property	Returns
sn:checkColour	URL for checking robot pen color (string)
sn:hasCapability	List of capabilities defined in the KB
sn:hasColour	One of the colors defined in the KB
sn:hasConveyorURL	URL for the conveyor (string)
sn:hasLocation	Defined for possible expansion of the KB (not used in current version)
sn:hasName	Name of customer (string)
sn:hasNeed	List of needs that are defined in the KB
sn:hasNeighbour	List of defined neighbours in the KB a conveyor has
sn:hasObject	List of defined objects in the KB a station has
sn:hasOrdered	List of defined orders in the KB a customer has
sn:hasPalletID	Pallet ID assigned to a cellphone (int)
sn:hasPaperLoaded	Value describing if a pallet has a paper loaded (Boolean)

sn:hasRobotURL	URL for the robot (string)
sn:hasServiceID	ID of a service (string)
sn:hasServiceURL	URL for a service which is appended to either the conveyor URL or the robot URL (string)

Table 1. The Properties in the Knowledge Base and the Values They Return

10. Evolution of Diagrams

In the inception phase, this project was hard to understand for our group because the concepts were new and the instructions for the project were not enough. However, as the course proceeded, and we went through each lecture, we got many inspirations from the lectures and the demos. Then we started to code a small part of this project such as the subscribing to the events and getting the requests and responses to then use the post method to achieve the functionality of acquiring the status of the neighbors. Gradually, we understood better and better the logic of this project and we were able to improve our work in each iteration.

For the activity diagram, initially, we had a simple version, but after we created a polished version of our knowledge base, we refined this activity diagram which shows the workflow of our system. Our system architecture diagram evolved as we gained more understanding of the course materials. The key to this project is we have a clear knowledge base which illustrates our logic of coding. Refinement of the knowledge base is an iterative job as we deepen our understanding of this project and finished small functionalities, like movements of pallets.

After the knowledge base was done, we refined the sequence diagram, because it was at this moment that we finally understood the whole process of our UI, our main program and our knowledge base interacting with each other. Finally, we improved our use case diagram after we developed the second prototype of source code as we felt that at this point we had a clearer idea on the matter.

11. Conclusion

This course's FASTory-lab production line simulator project was a two-period project which evolved with each iteration as we gained more knowledge on how to continue develop and better our work further and further. We were able to meet the objectives of this project according to the logic of our code. In the simulator's production line, we achieved the different tasks the robots and the conveyors were required to perform in order to complete customer orders. The logic on which we based our programming was initially based on the real production line; therefore, we did not develop the functionality of automatically loading the pallets. However, in the testing phase, the maximum number of pallets that could be loaded on the conveyors was three because usually the simulator server would crash when

overloaded. If the problems with the server could be fixed, the performance of the system could be improved significantly and made more stable as well.