Eindhoven University of Technology

MASTER

Enabling dynamic resource allocation for industrial robots by introducing a robot attribute taxonomy

Croonen, D.A.J.G.

Award date:
2019

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

Eindhoven, January 2019

Enabling Dynamic Resource Allocation for Industrial Robots by Introducing a Robot Attribute Taxonomy

by

Dennis (A.J.G.) Croonen

BSc Industrial Engineering & Management Sciences
Eindhoven University of Technology 2016

Student identity number 0811919

In partial fulfilment of the requirements for the degrees of

Master of Science
In Operations Management and Logistics

Master of Science
In Business Information Systems

Supervisors:
dr. ir. I.T.P. Vanderfeesten, TU/e, IE&IS, IS
J. Erasmus MSc, TU/e, IE&IS, IS
dr. D. Fahland, TU/e, W&I, AIS

TUE. School of Industrial Engineering.
Series Master Theses Operations Management And Logistics

Series Master Theses Business Information Systems

Subject headings: Dynamic Resource Allocation Algorithm, Industrial Robot, Robot Attribute Taxonomy, Smart Factory

Enabling Dynamic Resource Allocation for Industrial Robots by Introducing a Robot Attribute Taxonomy

Abstract

The primary purpose of this research is to create a robot attribute taxonomy, which can be used to decide which industrial robot can perform which task and which industrial robot is the best assignee for a certain manufacturing task. From an extensive literature review 203 different robot attributes are retrieved. The number of robot attributes is reduced by several criteria. The final taxonomy consists of 35 different robot attributes, which should cover all aspects of an industrial robot. The robot attribute taxonomy is used to create a method that can determine which robot can perform which task. Furthermore, a method is created that computes which robot would be the best assignee for a certain task, where execution time or execution cost is used as decision variable. These methods are introduced into an existing dynamic resource allocation algorithm, where after the adapted and the original dynamic resource allocation algorithm are compared using a simulation, based on the throughput time. When robot failure is excluded, the average throughput time of the tasks using the original algorithm is 123,35 sec, while the average throughput time using the adapted algorithm is only 115,49 sec. Furthermore, average time per task is decreased from 99,05 to 96,94. When robot failure is included, the average throughput time of the tasks using the original algorithm is 556,04 sec, while the average throughput time using the adapted algorithm is 280,76 sec. The adapted algorithm outperforms the original algorithm in both scenarios. These results show that using the designed methods can improve manufacturing processes and especially in a environments in which variability is high and malfunctioning of robots is common, the methods can be of great importance.

Keywords: Dynamic Resource Allocation Algorithm, Industrial Robot, Robot Attribute Taxonomy, Smart Factory

Summary

Nowadays, Business Process Management(BPM) is used in many companies to improve their processes. "*BPM is the art and science of overseeing how work is performed in an organization to ensure consistent outcomes and to take advantage of improvement opportunities"* (Dumas, La Rosa, Mendling, & Reijers, 2013). In addition to this definition, BPM is not about improving individual tasks, but rather about improving processes. Technical innovations are used continuously to enhance companies and their business processes. Currently, companies are preparing themselves for the fourth Industrial Revolution, also referred to as Industry 4.0 in Europe. The Internet of Things is the technology that is the real reason for this new revolution. Using IoT, manufacturing companies can be transformed into a so called smart factories. A smart factory is defined as "*the integration of all recent IoT technological advances in computer networks, data integration, and analytics to bring transparency to all manufacturing factories*" (Lee, 2015). Two main differences can be argued between traditional production lines and smart factory production systems (Wang, Wan, Li, & Zhang, 2016). First difference is that the smart factory has to produce multiple types of small-lot products, which is tackled by owning a diverse range of resources, while a traditional production lines are mostly fixed for a long time on the same product type, which requires less diverse resources. The second main difference is that in a smart factory production system the entities within the system are coping with system dynamics themselves, while in a traditional production line the system does hardly cope with any system dynamics. Any malfunction will break the full production line, since the machines are only preprogrammed to perform a certain task. Dynamic resource allocation is of great importance for smart factories.

The most common resource within manufacturing companies are human agents. Erasmus et al. (2018) enabled dynamic resource allocation for humans, by introducing a method consisting of a characterization method in which a human is described using 52 human abilities. With this method, also task requirements for humans can be created, where after a matching can be created between the task and the humans that meet the task requirements. This method was required, since all automatic resource allocation that were available did not provide any detailed explanation about humans and how it is decided which humans can perform which tasks. However, humans are not the only resource within manufacturing companies that can execute a wide range of tasks. Industrial robots are able to execute a wider range of tasks over the years. There are also several dynamic resource allocation algorithm for robots available(Das, McGinnity, Coleman, & Behera, 2015; Dias, 2004; B.P. Gerkey & Mataric, 2002). However, also those papers do not give any method for deciding which robots can perform which tasks. Since implementing dynamic robot allocation is inconsistent and difficult due to this, the aim of this research is to determine how robots should be described to enable task allocation during process run-time.

A literature study is conducted, in which 41 papers are reviewed on attributes to describe industrial robots. However, these papers are not aiming at robot allocation problems, since no papers about attribute-based robot allocation could be found. The papers target selecting the right robot in purchasing processes, in which they use some selection criteria for robots to compare the robots to each other. In the literature review, all unique attributes from the papers on the shortlist are retrieved and listed. This resulted in a list of 203 unique attributes. The shortlist is also used to define all the attributes. However, only 133 attributes could be defined by this method. Next, the attributes and especially the definitions of the attributes are reviewed by an expert. This resulted in a list of 143 defined and 60 undefined attributes. The number of attributes is reduced by using six criteria, of which an example is dublicate attributes. 128 attributes were excluded from the list. The last step is matching procedure, in which a list of all manufacturing tasks is compared to the remaining 75 attributes, where after all attributes that are not needed to describe any of the manufacturing tasks are excluded from the list of attributes. The final taxonomy contains 35 attributes, that were grouped and each of the attributes has a unit to define the attribute even further.

This taxonomy is evaluated on clarity, completeness and ease of use by experts on robotics. The experts had to fill in forms, of which one asked to fill in the attributes for a certain robot and of which the other asked to fill in the task requirements in terms of the attributes. The experts agreed on each other that the clarity and thus the definitions of the attributes were clear. On the contrary, the experts did not agree on each other on the other two evaluation criteria. From the received feedback, it is clear that the attributes of this taxonomy are not completely covered by the technical specification sheets of the industrial robots, since most experts could not find several attributes.

Nevertheless, the taxonomy is used to create a method that selects the best assignee for a task. The first step towards the method to select the best assignee for a task is to create an algorithm which decides if a robot can perform a certain task. Although that 35 attributes are included into the final robot attribute taxonomy, not all attributes will be used to compare robot requirements of tasks and robot attributes. Only 27 attributes are used to compare the task requirements and the robot attributes. For those 27 attributes, a table is created in which it is showed if the attribute must be larger or equal, smaller or equal, a subset of or to be equal to the robot requirement. With this table in mind, Algorithm 1 is created in which it is checked if the robot meets all task requirements. If the robot does meet all requirements, the algorithm will return true, otherwise false.

Next, a formula to estimate the execution time of a certain robot for a certain task is developed. This formula is combined with Algorithm 1, to create an method that decides which available robot is the best assignee for a certain task. It is assumed that the best assignee means the robot that has the lowest execution time for that task. Algorithm 2 first invokes Algorithm 1 to check whether the available robots can perform the certain task. All robots that can perform the task, make an execution offer using the formula, where after the robot with the lowest offer is returned. Algorithm 2 is verified by checking several relatively easy scenarios and in all of the test sets, the Algorithm returns the expected robot.

To show how this algorithm would improve an existing dynamic resource allocation algorithm, an allocation algorithm needs to be chosen. Choosing the algorithm is done by utilizing the iTax taxonomy (Korsah, Stentz, & Dias, 2013). For our type of problem, several allocation algorithms are proposed, of which the M+ System is one(Botelho & Alami, 1999). The M+ System is explained and some limitations of the algorithm are discovered. In the algorithm, the robot has to determine its offer when it is in the plan state. However, no explanation is given how this offer can be made. Furthermore, the allocation algorithm does not take failure of the robots into account. Therefore, the M+ System is adapted and an unavailable state is introduced in the state diagram. On top of that, it is decided that the offer is decided using the formula that is created in this thesis.

A simulation is used to see what the difference in performance is for a certain production process, while using a static allocation of robots or a dynamic allocation of robots. This comparison is done without and with failure of robots included in the simulation. The performance of the production process is measured by the average throughput time of the tasks through the whole system.

To cope with the stochastic variables in the simulation, each of the four scenarios is run 20 times for 24 hours, where after the average and the halfwidths of the performance indicators are calculated.

Comparing the static resource allocation and the dynamic resource allocation in the scenarios of no robot failure, the average throughput time of the tasks through the system is decreased from 123,35 sec to 115,49 sec. When the dynamic resource allocation is compared to the static resource allocation in the scenarios in which robot failure is included, this decrease is much higher: 556,04 sec to 280,76 sec. This shows that dynamic resource allocation enhances the performance of processes. This is also as expected. In general, when the utilization of a resource increases, the waiting times of the tasks is also increasing. With and without robot failure, the distribution of the tasks over the two robots is much more even while using the dynamic resource allocation and therefore the maximum utilization of the robots is lower when using dynamic resource allocation. These results show that using the designed methods can improve manufacturing processes and especially in a environments in which variability is high and malfunctioning of robots is common, the methods can be of great importance.

Preface

This master thesis marks the end of my journey as a student. Almost seven years ago, I walked over the campus for the very first time and was absolutely stunned about the differences between the university and my old high school. I am just a simple guy, from a 'little' town from the south of the Netherlands, coming from a small high-school and during that first walk over the campus I realized, that I would become a student in the large city of Eindhoven at the well-known and above all hugh Eindhoven University of Technology. I knew at that moment that many things would change in my life and I was not wrong.

In those almost seven years, I have enjoyed my student life a lot. I have met many great people, I have done and I have seen great things and above all, I have grown as a person enormously. But, after all those years, I am still stunned by the city and by the university when I get off the train at Eindhoven Central and I am walking towards the university. But now, while writing this, I know I am ready for my next step in life. Before doing that, I want to take this opportunity to thank all those great people. My student career would not be the same without you. Some people I want to thank in particular.

I want to thank those people that have helped me in the past year to get through this last phase of my student life, this thesis. Thank you Jonnro and Irene, for guiding me through this last year. You have helped me a lot with this thesis and I have really enjoyed our discussions about the project. You, and especially the combination of the practical-minded Jonnro and the more theoretical-minded Irene, really helped me to keep on going with this project.

I am really grateful that I could study the way I did, and for that I want to thank my parents. You have always supported me in everything and with every choice I made, even when those things or choices were not in favour of you (like going for an internation semester to Brisbane, Australia). Without your unconditional support and love, I would not have achieved the things that I have achieved now. I can't thank you enough for that! I love you guys!

Having a lovely time as a student also requires being surrounded by great people when you are not studying. So for that, I would like to thank my 'little' brother, Ton and Matheo, my family and my friends for their support and for making life great.

Last but certainly not least, I would like to thank Janou. In the last six years, we have both grown immensely. We have already reached many milestones together and now we are also graduating together. We have had ups and downs, but I know that your support me always with everything I do, like I support you. I am very proud of us and I am looking forward to reach many more milestones together. Just call it Magic!

Dennis Croonen

Eindhoven, January 2019

Table of Contents

List of Tables

List of Figures

1. Introduction

Standing still is going backwards. The meaning of this quote is that “if you are not developing yourself continuously, competition will catch up on you” (spreekwoorden.nl, n.d.). This quote is applicable to many situations in life and is for sure applicable in the business world. For companies, this quote should be a warning, since companies should reinvent themselves continuously to stand out, be better and, most important, stay better than their competition. When a company is not going forward, competitors will catch up and thus the company will move backwards in comparison to their competitors.

Nowadays, Business Process Management(BPM) is used in many companies to improve their processes. “*BPM is the art and science of overseeing how work is performed in an organization to ensure consistent outcomes and to take advantage of improvement opportunities”* (Dumas et al., 2013). In addition to this definition, BPM is not about improving individual tasks, but rather about improving processes.

Technical innovations are used continuously to enhance companies and their business processes. Also manufacturing companies are using technical innovations to enhance their business processes. In the past, three major technical innovations have changed the manufacturing industry drastically. Those three stages of industrialization were all triggered by technical innovations, like the water- and steam-powered mechanical manufacturing facilities in the 18th century, the electrically-powered mass production with the assembly lines at the beginning of the 20th century and the introduction of programmable logic controllers for automation purposes in the 1970s(Industrie 4.0 Working Group, 2013).

Currently, manufacturing companies are preparing themselves for the fourth Industrial Revolution, also referred to as Industry 4.0 in Europe. Also Industry 4.0 is enabled by a technological innovation, namely the introduction of the new Internet protocol IPv6 introduced in 2012 (Industrie 4.0 Working Group, 2013). Due to this new Internet protocol, the available number of IP-addresses have risen from 4.3 billion to 340 sextillion(3.4 *10^{38}) (Reddy, Ali, Sandeep, & Ravi, 2012). The almost infinite number of IP-addresses has enabled the Internet of Things. The basic idea of the concept Internet of Things (IoT) is that a variety of things or objects, such as RFID, tags, sensors, actuators mobile phones etc., are able to communicate with each other and cooperate with their neighbors to reach common goals, through unique addressing schemes.(Atzori, Iera, & Morabito, 2010) Where each device connected with the internet has an IP-address, the limited number of IP-addresses of the old Internet protocol would be insufficient to fulfill the needs of the growing IoT. In 2015, already 15.41 billion “things” were connected and it is forecasted that in 2025, the number of “things” connected will be 75.44 billion (“IoT,” 2016). This shows that IoT is developing and is expected to be developing rapidly in the upcoming years.

Where the new Internet protocol is the fundamental reason for the fourth Industrial Revolution, the Internet of Things is the technology that is the real reason for this new revolution. Using IoT, manufacturing companies can be transformed into a so called smart factories. A smart factory is defined as “*the integration of all recent IoT technological advances in computer networks, data integration, and analytics to bring transparency to all manufacturing factories*” (Lee, 2015). Wang, Wan, Li and Zhang(2016) enumerate differences between the traditional production line and the smart factory production system, which can also be seen as the difference between Industry 3.0 and Industry 4.0. Two of those differences are shown in table 1.

Table 1 - Differences between Smart Factory Production Systems and Traditional Production Line

Number in original paper	Smart Factory Production System	Traditional Production Line
1	**Diverse Resources.** To produce multiple types of small-lot products, more resources of different types should be able to coexist in the system.	**Limited and Predetermined Resources.** To build a fixed line for mass production of a special product type, the needed resources are carefully calculated, tailored, and configured to minimize resource redundancy.
5	**Self-Organization.** The control function distributes to multiple entities. These smart entities negotiate with each other to organize themselves to cope with system dynamics.	**Independent Control.** Every machine is preprogrammed to perform the assigned functions. Any malfunction of single device will break the full line.

A hypothetical process is introduced to be able to explain the differences in a more practical situation. The hypothetical process is a process which will be recognizable in some way for many companies, including manufacturing companies. The process is shown in figure 1.

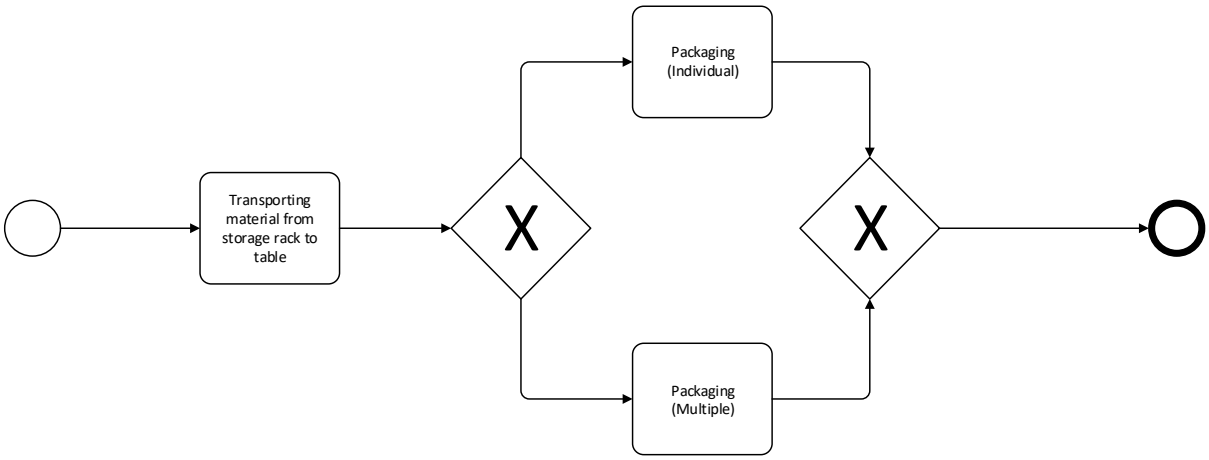

Figure 1 - The hypothetical process

The first task of the process is to transport a piece of material from a storage rack to a table. A resource should go to the storage rack, take the right material from the storage rack and bring the material to the starting point, where it is placed onto a table or an assembly line. After this task, the process has a XOR-split, which means that either the packaging(individual) task or the packaging(multiple) task will be executed. Both packaging tasks will have similar steps. Both tasks start with a resource taking the material from the table or assembly line, where after the the material is brought to the right container in which the material will be placed. The difference between the individual and the multiple packaging task is that in the individual task only one piece of material will be placed in one container while in the multiple packaging task, multiple materials will be placed into the same container. For the purposes of this discussion, we assume that it requires higher accuracy to place multiple items in the same container. Appendix A shows the tasks of this process from another perspective. Figure 25 shows how the process will look like on the floor, using arrows for the movements of the robots.

The two differences described by Wang et al(2016) will be illustrated using the hypothetical process. The first difference that will be illustrated is the increase of diversity in resources. Imagine materials that have a pretty similar weight in the traditional process, while those materials can have a much higher diversity in weight in Industry 4.0 due to mass-customization. To be able to lift all of the diverse materials, the versatility of the resource should be higher.

The second difference is the self-organization versus the independent control. In traditional processes machines are assigned and preprogrammed to perform the assigned functions, while in Industry 4.0 the entities will organize the allocation of tasks themselves, to cope with dynamics within the production process faster. In the hypothetical process, this can illustrated by imagining two different robots that each perform one of the two tasks of the hypothetical process. Since the robot that has to get the materials out of the storage rack should be tall enough to get the materials also from the top rack, this robot should be pretty tall. Generally speaking, a tall robot is not allowed to drive fast for safety reasons and will therefore be slower than smaller robots. The robot that is assigned to the packaging tasks can be smaller, since packaging does not require a tall robot. Therefore, this smaller robot is probably faster than the tall robot. In the traditional production processes, the tall robot will only execute the tasks of getting materials from the storage rack, while the smaller robot will only execute the packaging tasks. When the tall robot breaks, the small robot can only execute those tasks that are already retrieved by the tall robot before. When the small robot breaks, the tall robot can only retrieve as many tasks as can fit on the table. In Industry 4.0, when one of the robots breaks, the other robot will take over the tasks of the broken robot that it can perform. In this way, the production process will not stop when one of the robots breaks. On top of that, the robots can also swap tasks to improve efficiency in the Smart Factory production process, while this is not possible in the traditional process. This means that when a getting material task is located far away from the zero-point and the smaller robot can also retrieve the material from the rack since the material is laying on one of the bottom layers of the rack, it would be better to send the smaller robot, since it will take this robot less time to perform the task than when the tall robot should perform this task. When the small robot is executing this task, the tall robot will take over some of the packaging tasks of the small robot. In this way, the process will be more efficient.

As the differences between smart factories and traditional production processes show, dynamic resource allocation will be of great importance for the smart factories. The changes to the allocation should be made ad hoc and since the changes in allocation happen too fast and too often, human interference into the allocation methods would decrease the efficiency of the smart factories drastically. Therefore, the resource allocation should also be automated.

The most common resource within manufacturing companies are human agents. Humans are highly adaptable and are therefore very flexible resources within a manufacturing company. To enable dynamic resource allocation for humans, a method has been proposed by Erasmus et al. (2018). This method consists of a characterization method in which a human is described using 52 human abilities. On top of that, a task is described using the required abilities that a human should have to perform this task. After that, the method matches the humans that meet the requirements of the tasks to those tasks. This method is created, since all automatic resource allocation methods available did not provide any detailed specification of human factors that they used, which forced the users of those papers to come up with those specifications themselves. In the characterization method the Fleishman Taxonomy of Human Abilities is leveraged in a method to specify activities and human resources. The specified activities and human resources are used to allocate resources to the activities during process run-time. Hence, dynamic resource allocation for humans is already enabled.

Humans are not the only resource within manufacturing companies that can execute a wide range of tasks. Robots are getting more and more versatile over the years and are able to execute a wider range of tasks. For example, Rus et al(2002) advocate for self-configuring robots built from modules that can be reconfigured to execute a range of different tasks. Considering the rise of diversity of tasks in the current demand due to mass-customization and the more versatile robots, it is not strange that Stratista(2017) shows that the sales of industrial robots is growing rapidly all across the world and that it is expected to keep on growing in the upcoming years. Worldwide, the robot density has been increased from 74 to 85 industrial robots per 10,000 employees in 2017, where Europe is the region with the highest robot density of 106 in 2017 (IFR, 2018). The country with the highest robot density is the Republic of Korea with 710 robots per 10,000 employees, followed by Singapore(658 robots), Germany(322) and Japan(308). Despite that, it is clear that the growth potential for automation of manufacturing industries by using robots is still tremendous in most countries.

There are several resource allocation algorithms for robots. Most of those algorithms solve the resource allocation problems dynamically, which means that the allocation might change over time. Some of those algorithms also using heterogeneous robots(Das et al., 2015; Dias, 2004; B.P. Gerkey & Mataric, 2002). In those papers, it is decided which robots can perform which tasks. However, the decision regarding which robot can perform which task is different and case-specific in all of the papers.

Demand for customized products is high at this moment, while customers still demand low prices, high quality and low lead times(Hofmann & Rüsch, 2017; Mishra, Pundir, & Ganapathy, 2014). Customized products affect the production process, since the tasks of a certain order can have different resource requirements. Manufacturing enterprises are pursuing smart solutions to meet the demand for customized products, while keeping low prices, low lead times and high quality. Smart factories enable highly flexible production processes, increase efficiency of the resources and retain the high quality of the products. In smart factories, the diversity of resources is larger than in traditional production factories and the resource allocation is dynamical and automated(Wang et al., 2016). Dynamic resource allocation for humans is enabled by the method proposed by Erasmus et al.(2018), in which human employees are described using Fleishman's Taxonomy(Fleishman, 1975). Dynamic resource allocation for heterogeneous robots is an ongoing topic in scientific research(Das et al., 2015; Dias, 2004; B.P. Gerkey & Mataric, 2002). However, in those papers a general way of describing robots and the requirements for tasks is missing. In addition, only little research has been conducted regarding describing robots. Nof and Fisher(1982) propose a list of robot characteristics and skills, that can be used to describe robots. However, the age of that particular research warrants a fresh look at the typical characteristics used to describe robots. Furthermore, this paper has a low number of citations, which indicates that the taxonomy has not been adopted in other research.

No relevant literature can be found regarding describing robots to enable resource allocation. Nevertheless, a general method for describing robots is needed, to maximize gains for implementing Industry 4.0 into manufacturing companies. Without this method, the dynamic resource allocation for industrial robots is lacking. Hence, the problem statement of this research is:

Implementing dynamic robot allocation is inconsistent and difficult, due to the lack of a clear and consistent set of criteria that should be considered during robot-task allocation.

To enable standardized dynamic resource allocation for robots, the decision of which robot can perform which task should be standardized. Despite, no universal method of describing robots and task requirements can be found. Consequently, the aim of this research is:

Determine how robots should be described to enable task allocation during process run-time in a manufacturing environment. On top of that, this research will focus on how to match robots to the tasks that they can perform in a standardized way.

The main contribution of this thesis is threefold:

1. We propose a comprehensive taxonomy of robot attributes that can be used to describe the output of industrial robots. Only the output of the robot is described by the attributes, since the taxonomy is only used for determining which robot can perform which tasks in a standardized way. To enable this, only knowledge about the output of the robot is necessary. The taxonomy is used to describe industrial robots, but is also used to describe the requirements of the industrial robots to be able to perform a certain task.
2. We show a method that determines which industrial robot is the best assignee for a certain task using the taxonomy of robot attributes. More specifically, the method will compare the robot attributes and the robot requirements of the tasks to determine if the robot meets the requirements of the task. From all robots that meet the requirements of the tasks, the best assignee will be computed.
3. We give an example of an implementation of the method into an existing dynamic allocation algorithm. This instantiation shows how the allocation algorithm will work using the standardized method for determining which robot can perform which tasks. The instantiation will use the hypothetical process model, as showed in figure 1.

The remainder of this thesis is structured as follows. Section 2 describes the methodology of this thesis. Section 3 describes the methodology of creating the taxonomy of attributes for describing industrial robots. Also the final taxonomy will be discussed in greater detail. On top of that, the evaluation of the taxonomy will be discussed. In section 4, the development of the method for determining the best assignee for a task will be discussed. Furthermore, section 5 will discuss the methodology to create a simulation of this implementation. Besides decisions regarding the simulation, the UML-diagram and the pseudocode of the simulation will be discussed. The results of the simulation will be shown in section 6. Finally, the conclusions of this research are provided in section 7. Also, a discussion will be included in this section, in which the contributions and the limitations of this research are pointed out and directions for further research are discussed.

2. Methodology

Section 1 discusses that the contributions of this thesis are the development of a taxonomy, a method and an instantiation. Hence, a design methodology would fit the best for this research. Peffers et al.(2007) introduced a design science research method(DSRM) for IS research.

As can be seen in figure 2, the process model is slightly changed for this research in comparison to the original DSRM as introduced by Peffers, since multiple IT artifacts will be developed in this research, while Peffers method is aiming at creating just one artifact. Therefore, the design and develop step will be executed two times for both the taxonomy and the method, while the instantiation is created in the demonstration part of the process. In the remainder of this section, each step of the DSRM process model will be explained in greater detail.

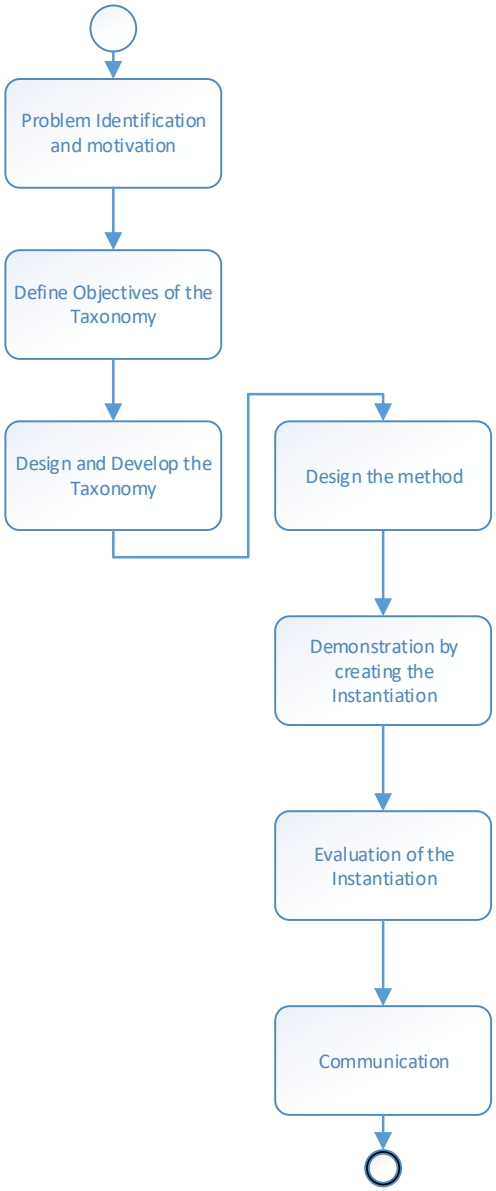

Figure 2 - Adapted DSRM process model

2.1 Problem Identification and Motivation

The first step of the DSRM process model is the problem identification and the motivation step for the research. In this step, the specific research problem is defined and the value of a solution is justified(Peffers et al., 2007). Both the defining of the research problem as the justification of the value of a solution is done in the Introduction section of this report. The remainder of the research will be built on that section.

2.2 Define Objectives of the Taxonomy

The second step of the process model is defining the objectives of a solution/artifact. The objectives will be inferred from the problem definition and the knowledge of what is possible and feasible. Objectives can be quantitative, such as terms in which a desirable solution would be an improvement in comparison to current ones, or qualitative, such as a description of how the artifact is expected to support solutions to problems up to now not addressed(Peffers et al., 2007).

2.3 Design and Develop the Taxonomy

The taxonomy will be designed and developed in this step. To design and develop the taxonomy, a sub-process will be executed, which will be discussed in greater detail in section 3. In this step, the following questions are answered

- *Which attributes of robots should be defined to enable allocation to tasks during process run-time?*
- *How can task requirements for robots be defined using the attributes of robots?*

2.4 Design the Method

In this step, the method for determining which robot is the best assignee for a certain task is designed. The method will use the robot attribute taxonomy that is developed in the previous step. First of all, for all of the attributes in the taxonomy is decided if it can be a task requirement or an attribute that will be used to determine the best assignee. With the attributes that can be task requirements, an algorithm will be created which determines if the robot meets the task requirements. Next, an algorithm is created to calculate the execution time of a robot for a certain task. In this algorithm the attributes that are not used for task requirements will be used. Last step is to combine those two algorithms into an algorithm that determines of all available robots, which robot can execute the certain task as fastest. Thereafter, the designed algorithm will be verified. The following questions are answered in this step:

- *How can be decided which tasks a robot can execute using the attributes of robots?*
- *How can multiple robots be compared to allocate the most beneficial to the task?*

2.5 Demonstration of the artifacts/solutions

In this step, the taxonomy and the method for determining which robot is the best assignee for a task will be introduced into an existing dynamic resource allocation algorithm. First, an existing allocation algorithm is chosen. Next, this algorithm will be adapted by introducing the methods and thus the taxonomy.

2.6 Evaluation of the Artifacts/Solutions

In this step, the observing and measuring of how well the artifact supports a solution to the problem will be executed, by comparing the objectives of a solution to actual observed results from the demonstration step(Peffers et al., 2007). The adapted algorithm will be compared to the original algorithm by creating a simulation. In this simulation, the hypothetical process that is introduced in section 1, will be used. The adapted and the original algorithm are simulated on the hypothetical process and the results regarding some KPI's will be compared.

2.7 Communication

In this last step, communication of the research will be executed. Concrete, this means that the research will be reported and communicated to the relevant audiences. Two oral presentations are scheduled: One mid-term presentation in one of the HORSE meetings to discuss progress up until that point in time and one final presentation including a formal review. This final presentation will take place at Eindhoven University of Technology, and will be covering the whole project including final findings, conclusion, recommendations and directions for further research. Also, a scientific paper about the project will be created and published. On top of that, the results of this research will be used and developed further within the HORSE project.

3. The Taxonomy of Robot Attributes

In this section the general taxonomy of robot attributes is presented .This section will start with summarizing the conducted literature study and showing the taxonomy created from the literature review. Thereafter, this taxonomy is revised and several other sources are used to enhance the definitions of the attributes. Next, incomplete and incomprehensible attributes are removed. With the remaining attributes, a matching procedure is started. In this procedure, the attributes are matched to a list of manufacturing tasks. The attributes that cannot be matched with at least one task, will be excluded from the list again. After that, the grouping of the remaining attributes is discussed and the final taxonomy is presented.

To illustrate the methodology within this section better, the methodology is shown in a process view in figure 3.

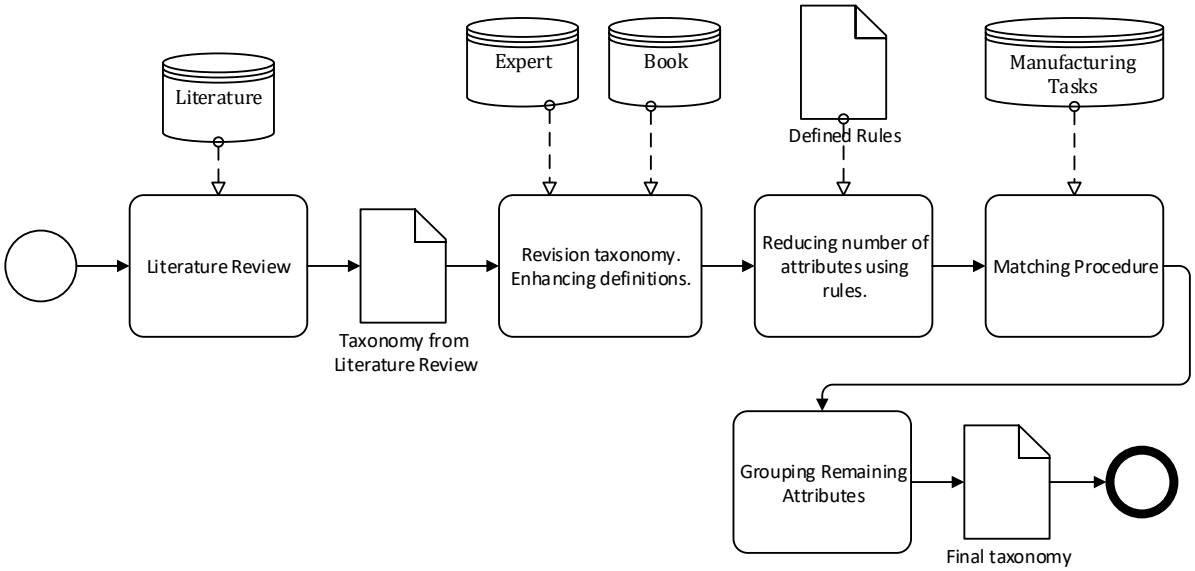

Figure 3 - Methodology towards taxonomy of Robot Attributes

However, according the DSRM process model the first step of a design methodology is to define objectives for the to be designed artifact. The objective of this thesis is to create a robot attribute taxonomy that can be used for deciding which robot can perform which task in a manufacturing company. Therefore, the objectives of the artifact are the following:

- The final taxonomy should be as complete as possible. It should be possible to describe as many as possible industrial robots and manufacturing tasks using this artifact only.
- The final taxonomy, the used attributes and the corresponding descriptions should be clear.
- The final taxonomy should be easy to use. The time needed to describe an industrial robot or a manufacturing task using the final taxonomy should be low. Furthermore, the taxonomy should also be useable for people that never used to taxonomy before.

A summary of the results of the literature study will be discussed shortly in section 3.1. Next, the revision step is discussed in section 3.2, where after section 3.3 covers the reducing number of attributes using rules, which will be also defined in this section. The last reduction step is called the matching procedure. In this procedure, it is decided which attributes are needed to define the requirements of a robot to perform each of the manufacturing tasks. This

step is discussed in detail in section 3.4. The last step is to structure the remaining attributes and group those, which is discussed in section 3.5. After that,

3.1 Summary of the Literature Study

To enable dynamic resource allocation for industrial robots, a matching procedure is needed to be able to see which robot can execute which task. Erasmus et al. (2018) enabled dynamic resource allocation for human employees by adopting the Fleishman Taxonomy of Human Abilities and using this in their proposed method for matching humans to tasks that they can perform. Comparatively, a literature review is conducted to look for a similar comprehensive taxonomy or robot attributes to describe industrial robots enabling dynamic resource allocation for robots. In the review, it is concluded that no such comprehensive taxonomy for industrial robots exist. Hence, the literature review is aimed at retrieving different robot attributes from literature, to be able to create a taxonomy from all those attributes.

In the literature study, a shortlist of 41 papers is reviewed on attributes to describe industrial robots. However, these papers are not aiming at robot allocation problems, since no papers about attribute-based robot allocation could be found. The papers target selecting the right robot in purchasing processes, in which they use some selection criteria for robots to compare the robots to each other. Also in this research field, it is argued that selection criteria is often overlooked in such problems (Parameshwaran, Praveen Kumar, & Saravanakumar, 2015; Tansel İç, Yurdakul, & Dengiz, 2013). Where most of the papers use only some attributes, some papers also claim to have a complete list of possible selection criteria of robots (Bhangale, Agrawal, & Saha, 2004; Datta, Sahu, & Mahapatra, 2013; Parameshwaran et al., 2015). However, also those lists are missing some selection criteria that have been used in other papers, which means that those lists are also not fully complete.

In the literature review, all unique attributes from the papers on the shortlist are retrieved and listed. This resulted in a list of 203 unique attributes. Next, it is tried to define all attributes using only the papers that are on the shortlist. On top of that, for the attributes that are “common sense” (like weight of the robot) and that are not defined in any of the papers, a definition is created. This resulted in 133 attributes defined and 70 attributes undefined.

To structure the list of attributes in some form, a grouping based on the proposed grouping by Bhangale(2004) is used. The grouping that Bhangale proposed is slighty changed. The number of groups is increased from eight to ten. The groups are shown in figure 4. On top of that, the groups with the corresponding attributes are shown in Appendix B.

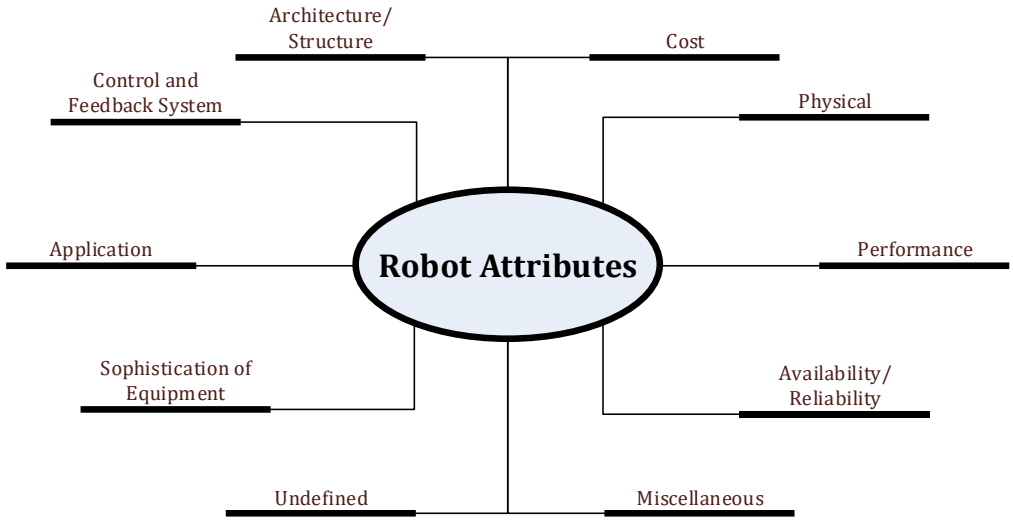

Figure 4 - Categories of Attributes proposed by Croonen(2018)

3.2 Revision of attributes identified during the literature study

This section discusses the revision of Croonen's Taxonomy. More accurate, the definitions of the attributes will be revised in two ways: The first way is to use part 12 of the book Handbook of Industrial Robotics (Ceroni & Nof, 1999), in which a long list of robotics terminology is described. Secondly, an expert in robotics is reviewing the list of attributes and the corresponding definitions.

Firstly, part twelve of Handbook of Industrial Robotics(Ceroni & Nof, 1999) is used to define some of the yet undefined attributes. In table 2 the six attributes that are defined by this method are shown with their corresponding definition.

Table 2 - Added definitions using Handbook of Industrial Robots

Attribute	Definition
Manipulator	Mechanism, usually consisting of a series of segments, or links, jointed or sliding relative to one another, for grasping and moving objects, usually in several degrees of freedom. It is remotely controlled by a human (manual manipulator) or a computer (programmable manipulator). The term refers mainly to the mechanical aspect of a robot.
Man-machine interface	Same as user interface. The interface between the robot and the operator through devices such as a teach pendant or PC. It provides the operator with the means to create programs, jog the robot, teach positions, and diagnose problems.
Path Measuring Systems	A part of the mechanical construction of each axis which provides the position coordinate for the axis. Typically, for translational axes, potentiometers or ultrasound are used for path measuring systems. But for rotational axes, resolvers, absolute optical encoders, or incremental encoders are used. A path measuring system may be located directly on a robot axis or included with the drive system.
Resolution	The smallest incremental motion which can be produced by the manipulator. Serves as one indication of the manipulator accuracy. Three factors determine the resolution: mechanical resolution, control resolution, and programming resolution.
Precision	A general concept reflecting the robot's accuracy, repeatability, and resolution.
Autonomy	The capability of an entity to create and control the execution of its own plans and/or strategies. For instance, in mobile robots the ability of the robot for determining the trajectory to reach a specific location or pose.

Next, an expert is consulted for validating the definitions of the attributes. This expert is an expert on robotics, but also is an expert on process management and therefore knows how attributes should be described for enabling dynamic robot allocation. The expert advised to remove the definitions of two attributes, *gripper control* and *Assemblability*, since the definitions did not clarify the attributes at all and also the expert did not know the definitions of the attributes exactly. On top of that, he advised to change or add the definitions of the following 28 attributes. Table 3 shows those attributes, the old definition and the definition that the expert advised.

Table 3 - Attributes of which the description changed

Attribute	Old Description	New Description
Adaptability	Adaptability of the robot	Average duration to change the configuration of the robot to perform different operations or at different locations.
Ambient Temperature	Work area temperature	Range of temperature within which the robot functions as expected.
Communication Bandwidth		Range of bandwidth on which the robot can communicate.
Communication Range		Range of the communication system of the robot.
Connection Type	The place (base, ceiling, wall) where the robot arm is settled. The facility layout, constraints, and obstacles are taken into account to choose the connection type	The place (base, ceiling, wall) where the robot is settled. The facility layout, constraints, and obstacles are taken into account to choose the connection type.
Dexterity	The dexterity of industrial robots, introduced here as C6, has many definitions in the literature. Mainly, it can be considered as the ability to change position and orientation of the end effector, between two different configurations of the robotic structure	The ability to change position and orientation of the end effector, between two different configurations of the robotic structure.
Efficiency	Efficiency of the robot	Ratio of output to input.
Environmental Impact	Similar to something like energy consumption	The effect of the robot operation on the environment.
External State Sensors	Sensors	The ability of the robot to detect externally perceivable effects that it has.
Force Output	Output of Force	Amount of force a robot can generate.
Internal state sensors	Sensors	The ability of the robot to detect the current status of its internal operation.
Maintainability /Regular Maintenance	Easy to maintain	Ease and cost of maintenance.
Material of Robot	Material of the robot	What material does the outer surface of the robot consist of.
Mounting Position	Mounting position represents the ability of a robot to fasten it securely on a given surface during the working cycle,	Possibility to attach the robot to surfaces in different orientations. (for example ground, wall, ceiling and mobile platforms).
Noise Delivery	Noise Output	The maximum noise output of the robot during runtime.
Operation Cost	Cost for running the robot	Total cost of running the robot.
Operation Time	Duration operation time	Duration of continual operation without required rest.

Attribute	Old Description	New Description
Overload Capacity of the Robot	Maximum capacity	Load capacity beyond the recommended specification.
Power Consumption	How much power does the robot use	How much electric energy is consumed during operation
Power Source Requirement	Power/Energy	The type of energy consumed by the robot.
Productivity	Output of the robot	Number of actions that can be performed in a time unit.
Programming Flexibility	while programming flexibility refers to a robot's ability to accept different programming codes.	Programming flexibility refers to a robot's ability to accept different programming codes.
Recommended Operating Humidity	Recommended humanity around the robot	Range of humidity within which the robot functions as expected.
Rest Time	Downtime	Time required by the robot to reset between operations.
Sensing Range		Distance to recognise a predefined shape, image or event.
Space Requirements		How much space does the robot need to be able to function correctly.
Velocity	Is used as Maximum Tip Speed in 1	Is used as Maximum Tip Speed.
Working Range of Joints	Range	Combination of Degrees of Freedom and Reach.

3.3 Reducing number of attributes of the Taxonomy

Currently, 143 out of 203 attributes are defined. Before the matching procedure is executed in the next section, the list of 203 attributes is revised again. This revision is executed by the author of this thesis, whereafter the expert reviewed this revision. The aim of the revision is to decrease the number of attributes, using certain criteria. Six criteria are used to exclude some of the attributes:

1. If an attribute is too abstract to use for the matching procedure. For instance Usage of Robot, which is really general and therefore too abstract.
2. If an attribute is a subset of another attribute and the excluded attribute is too precise to use for the matching procedure. For instance AC power consumption, which is a subset of power consumption.
3. If an attribute is a duplicate of another attribute. For instance connection type, which description matches exactly to mounting position.
4. If an attribute is a superset of another attribute and the excluded attribute is too abstract to use in the matching procedure. For instance External state sensors, which covers slip and tactile sensors.
5. If an attribute is a component attribute of the robot rather than an output attribute of the robot. Only output attributes can be used for the matching procedure. Since the goal of the Taxonomy is to enable dynamic resource allocation, the taxonomy of attributes should cover what the robots can do, where they can go and what risks they pose and not in what material and components they consist of. An example of such an attribute is cable layout.
6. If an attribute is not defined, since it is impossible to match an undefined attribute to a task. For instance environmental performance, where no definition could be found for.

Table 4 shows the number of excluded attributes for each of the rules. The table shows that 128 attributes are excluded based on the rules, which implies that 75 attributes remain. Those 75 attributes will be used for the matching procedure.

In Appendix C, six tables are shown. Each table represents one of the rules and the attributes that are excluded because of that rule. Those attributes will be shown in that table with their corresponding definition and the more detailed reason of excluding.

Table 4 - Number of excluded attributes

#	Rule	Number of attributes excluded
1	**Too abstract**	11
2	**Subset of**	58
3	**Duplicate of**	15
4	**Superset of**	9
5	**Component attribute**	13
6	**Not defined**	22
	Total	***128***

3.4 Matching Procedure

This section will cover the matching procedure. This matching procedure is done to reduce the number of attributes even further. To be sure that only attributes required for describing an industrial robot are in the taxonomy, but also make sure that the taxonomy is as complete as possible, the attributes are matched with a complete list of manufacturing tasks. When a certain attribute might be required for matching an industrial robot to one of the manufacturing tasks, the attribute will stay in the taxonomy, while when the attribute is certainly not needed for any of the tasks, the attribute will be excluded. Therefore, this list of manufacturing tasks will be discussed first, where after the matching procedure will be explained in greater detail.

3.4.1 List of Manufacturing Tasks

To be able to execute the matching procedure, a complete list of manufacturing tasks should be used. The list of manufacturing tasks is created using three different sources, which combined represent all manufacturing tasks(Erasmus, Vanderfeesten, Traganos, & Grefen, 2019).

IEC62264:2013 recognizes four categories of manufacturing operations: Production, Inventory, Maintenance and Quality Operations(IEC, 2013). However, the standard does not go in greater detail in any of those categories and therefore other sources are required to retrieve a list of manufacturing tasks.

Groover(2017) argues that the production operations can be grouped in only four process groups because of the nature of material processing: Materials can be shaped or not and when materials are shaped, this can only be done in three ways: Mass-reduction, mass-increasing(Assembly) and mass-conserving. Within those groups, Groover identifies ten production types. This list of production types is also confirmed by DIN 8580, a well-established German manufacturing standard(German Institute for Standardisation, 2003). The purpose of DIN 8580 is to identify all production process types and hence it can be assumed that the ten production types are all types possible within the production operations category.

The INCOSE handbook provides detailed groups and types for the Inventory, Maintenance and Quality Operations.(International Council on Systems Engineering (INCOSE), 2015) Inventory Operations are divided into packaging, handling, storage and transportation(PHS&T) groups, each with their own process types. Maintenance Operations are divided into the groups corrective and preventive maintenance and system modification. Corrective and preventive aim to deliver the intended performance of equipment, while system modification goes beyond designed performance by extending the life or improving performance. Finally, Quality Operations is divided into process quality and product quality according to the INCOSE handbook. For Process Quality, the corresponding types are Process Measuring, Equipment Calibration, Equipment Inspection and Equipment testing, while for Product Quality the corresponding types are Product Inspection, Product Measuring and Product Testing.

The complete list of manufacturing tasks is displayed in table 5. For each manufacturing task, the table shows the category and group to which the task belongs and the description of the manufacturing task.

Table 5 - Complete list of Manufacturing Tasks

Category	Groups	Manufacturing Task	Description
Production	Mass-conserving	Casting/ moulding	Creation of an initial shape from the molten, gaseous or formless solid state.
Production	Mass-conserving	Forming	The three-dimensional or plastic modification of a shape while retaining its mass and material cohesion.
Production	Mass-increasing	Joining	Input parts are combined (temporarily or permanently) to produce the product.
Production	Mass-increasing	Assembly	A mass-production arrangement whereby the work in process is progressively transferred from one operation to the next until the product is assembled.
Production	Mass-reduction	Material removal	Input materials or parts are processed by reducing their mass (cutting, milling, etc.).
Production	Non-Shaping	Heat treatment	Thermal energy is applied to materials or parts to enhance product properties.
Production	Non-Shaping	Surface coatings	Material is added to the surface of a product to improve its properties.
Production	Non-Shaping	Surface treatments	Input parts or products processed (machining, grinding, etc.) to enhance surface integrity.
Inventory	Packaging	Individual Packaging	Single product is inserted into appropriate container for transport or storage.
Inventory	Packaging	Unitizing (a.k.a. Containerization)	Multiple parts or products are inserted into a single container for transport or storage.
Inventory	Handling	Preparatory	Bringing materials closer to the workplace and preparing the machine.
Inventory	Handling	Feeding	Placing or directing materials closer to work place or point of use
Inventory	Handling	Positioning	Orienting materials in exact location, placing into fixture, jig or machine
Inventory	Handling	Manipulating	Handling of materials during actual manufacturing operation
Inventory	Handling	Removing	Taking material out of workplace, such as taking out of jig, fixture etc.
Inventory	Storing	Temporary	Static or slow holding of materials, parts or products (buffering, queueing, etc.) in preparation for further processing.
Inventory	Storing	Permanent	Static holding of materials, consumables, parts or products until retrieved for production or distribution.
Inventory	Transporting	Vehicular	Directed physical movement of materials, consumables, parts or products aboard vehicles (agv, forklift, etc.)
Inventory	Transporting	Fixed installation	Fixed, point-to-point physical movement (conveyor, cable, piping, etc.) of materials, consumables, parts, products or energy.

Category	Groups	Manufacturing Task	Description
Maintenance	Corrective Maintenance	Repairing	Any activity which returns the capability of an asset that has failed to a level of performance equal to, or greater than, that specified by its Functions, but not greater than its original maximum capability.
Maintenance	Corrective Maintenance	Replacing	A maintenance task to replace a component when it has failed or deteriorated to the point where system performance is outside specified parameters.
Maintenance	Preventive Maintenance	Servicing	Any activity which returns the capability of an asset that has not failed to a level of performance equal to, or greater than, that specified by its Functions, but not greater than its original maximum capability.
Maintenance	Preventive Maintenance	Scheduled replacing	A maintenance task to replace a component at a specified, pre-determined frequency, regardless of the condition of the component at the time of its replacement.
Maintenance	Preventive Maintenance	Condition monitoring	The use of specialist equipment to measure the condition of equipment to assess whether it will fail during some future period.
Maintenance	System Modifications	Sustain	Alter the configuration of a system to extend its useful life
Maintenance	System Modifications	Upgrade	Alter the configuration of a system to deliver additional functionality or improve performance.
Quality	Process Quality	Equipment inspection	Conformity evaluation by observation and judgement accompanied as appropriate by measurement, testing or gauging.
Quality	Process Quality	Equipment testing	An activity in which a system or component is executed under specified conditions, the results are observed or recorded, and evaluation is made of some aspect of the system or component.
Quality	Process Quality	Process measuring	A set of actions to determine the value of a quantity of a process.
Quality	Process Quality	Equipment calibration	Set or adjust a machine or tool according to product and production requirements.
Quality	Product Quality	Product Inspection	A static analysis technique that relies on visual examination of development products to detect errors, violations of development standards, and other problems.
Quality	Product Quality	Product Measuring	Record quantifiable metrics to determine the performance or health of a system.
Quality	Product Quality	Product Testing	Determination of one or more characteristics according to a procedure.

3.4.2 The Matching Procedure

The complete list of possible manufacturing tasks and the explanation of that list is given in section 3.4.1. Furthermore, the scope of this project is to match robots to tasks in a manufacturing environment only. Therefore, after matching the attributes to the list of manufacturing tasks, attributes that are not used in the matching procedure to describe at least one of the tasks will be excluded from the list. This implies that the taxonomy for describing industrial robots will be specific for the manufacturing industry after excluding those attributes. The matching procedure will be executed according to the following steps:

1. One of the tasks is chosen. It is important to make sure that the task and the corresponding actions within that task are known and clear. Therefore, the description, but also additional information is used, like the book Manufacturing Process Selection Handbook(Swift & Booker, 2013)
2. Take the next attribute of the list. Might this attribute be of any importance in determining if a robot can execute the task? On top of that, might the attribute be of any importance in determining if that robot can perform the task better than another robot? When the attribute can be of any importance, mark this attribute for this task. When the attribute is definitely not needed for matching a robot to this task, don't mark this attribute for this task.
3. Continue step two until all 75 attributes have been judged. After that, go back to step 1 until all attributes are judged for all of the tasks.

To make this matching procedure more clear, an example will be discussed. At step one for example, the task *assembly* is chosen. Next, one of the remaining attributes is chosen. Let's choose the attribute *vertical reach*. Since in *assembly* tasks also large pieces can be assembled and the robot needs to assemble parts of certain heights, *vertical reach* might be a robot requirement to execute an *assembly* task. Therefore, *vertical reach* is matched to the *assembly* task. Next, let's take the attribute *man-machine interface*. Since it is assumed that the task will be executed by only one robot without any human interference, the man-machine interface is not important for this task or any of the other tasks. Therefore, man-machine interface is excluded. This is done for all 75 attributes for each of the manufacturing tasks.

The matching procedure is executed by the author of this thesis, after he and the expert went through the procedure for two manufacturing tasks. After this, the author executed the procedure for the remaining tasks. This procedure is reviewed by the expert afterwards, to reduce the subjectiveness of the procedure as much as possible.

The matching procedure works for most of the manufacturing tasks, however nine of the tasks cannot be described using the attributes. In table 6, the tasks that cannot be described are shown and additionally also the reason why the task cannot be described.

Table 6 - (Groups of) Manufacturing tasks that could not be used in the matching procedure and the reason

Task/Group	Reason
Storing group	Storing operations are not performed by robots. The tasks to place things in storage and remove it from storage are part of the handling and transporting process groups.
Fixed installation	Fixed installation transporting is not assigned to any robot, but rather executed by equipment such as pipes or conveyor belts.
Equipment inspection	The equipment that is inspected might be a robot, but will be executed by a human.
Equipment testing	The equipment that is inspected might be a robot, but will be executed by a human.
System Modifications	The equipment that is modified might be a robot, but will be executed by a human.
Process and product measurement	These are operations executed by sensors, to detect process or product defects. Thus, the operations are not executed by industrial robots.

The matching procedure resulted in 35 attributes that are used at least once, while the other 40 attributes are not used. Appendix D shows the excluded attributes with their corresponding definitions.

3.5 Grouping the Remaining Attributes and adding units to the attributes

Section 3.4 discusses that 35 attributes are used for describing at least one of the manufacturing tasks. In this section, the remaining attributes will be grouped to make the taxonomy more user-friendly. This grouping will be different from the grouping used in the literature review, since the number of categories is too large for the remaining number of attributes. For this final model, only 7 categories are proposed. Some of the categories are copied from the grouping of the literature review, but also some are created by attributes which are covering the same aspect of the robot, like the reach attributes. This grouping step does not change anything about the attributes that are in the taxonomy, but rather makes it easier for a user to work with the taxonomy. The following categories are proposed:

1. Cost: Cost is defined as “The effort, loss, or sacrifice necessary to achieve or obtain something” (Oxford Dictionaries, n.d.). All attributes that represent a sacrifice will be grouped into this category. This category is also used in the literature review.
2. Output: In this category all attributes are grouped which are representing some sort of output, in a positive or a negative manner.
3. Structure: All attributes that have something to do with structure of the robot are grouped in this category. These attributes are mostly about hardware that the robot has.
4. External Requirements: All attributes that have something to do with external requirements are grouped in this category. External requirements are requirements to the environment in which the robot has to run.
5. Automation: All attributes that are looking at how automated to robot is, are placed in this category.
6. Precision: Precision is one of the attributes that was excluded, since it is a superset of other attributes like repeatability. In this grouping, precision is used as category -name, after which the attributes that are subsets of precision are included into this category. Precision is defined as “a general concept reflecting the robot’s accuracy, repeatability, and resolution.”

7. Reach and Distances: All attributes describing some aspect of the reach of the robot, are placed into this category.

To define the 35 attributes even more, units are proposed for each of the attributes. The unit of measurement of most attributes is obvious. For instance the unit of Noise Delivery is decibel(dB). However, units of in particular two attributes are less obvious. Therefore, the units of Degree of Protection and Autonomy will be explained in more detail.

First of all, the attribute Degree of Protection can be defined using IP(Ingress Protection)-codes. Using this international classification for describing the degree of protection of electrical equipment against intrusion of the equipment of foreign bodies and moisture, the robot degree of protection can be presented easily(International Electrotechnical Commission, 2001).

The second attribute is Autonomy. A standardized method for defining the Autonomy level of a robot is required to be able to distinguish industrial robots. Beer, Fisk and Rogers(2014) propose a framework for Levels of Robot Autonomy. This framework can be seen in Appendix E. Using this table, the level of Robot Autonomy can be decided.

3.6 Final Taxonomy

This section shows the final taxonomy, in which the attributes, their definitions and their units are divided over the seven categories. First the top-level taxonomy is shown in figure 5, where after the different categories and their corresponding attributes are displayed in tables 7 until 13.

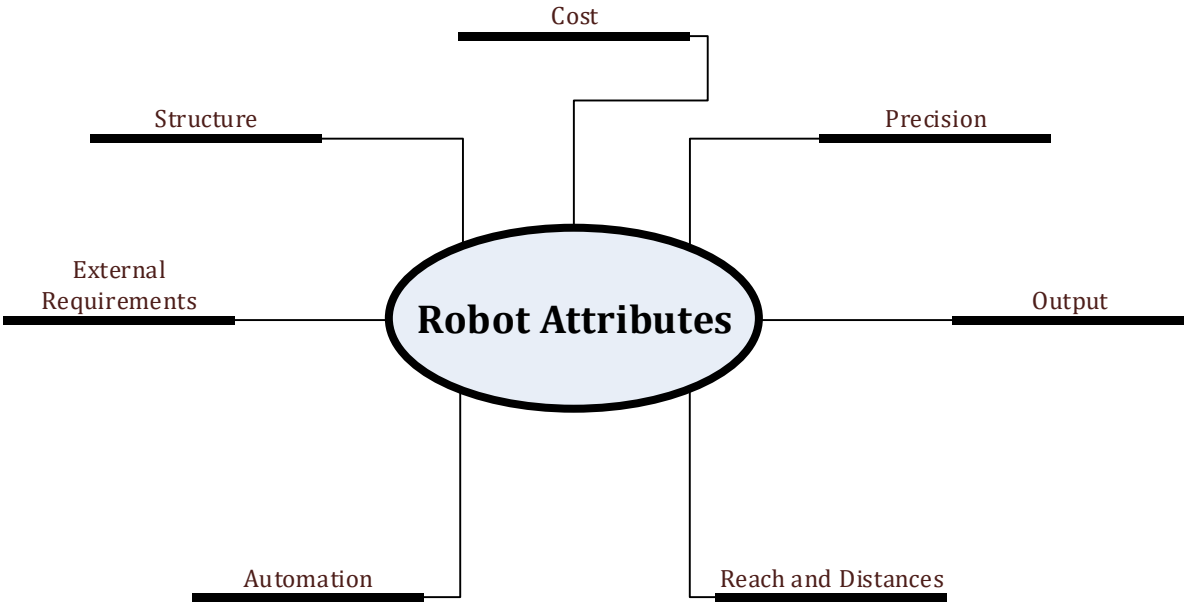

Figure 5 - Top-Level taxonomy

The top-level only shows the different categories of attributes used to describe the industrial robots. In the following tables, each of the categories will be covered, showing the corresponding attributes of those categories.

Table 7 – Proposed Cost Category

Cost	Description	Unit
Operation Costs	Total cost of running the robot.	€/s
Power Consumption	How much electric energy is consumed during operation	kWh
Maintenance Costs	Cost for maintaining the robot. This excludes cost of lost opportunity.	€/year
Environmental Impact	The effect of the robot operation on the environment.	CO_2/Day

Table 8 – Proposed Output Category

Output	Description	Unit
Operation Time	Duration of continual operation without required rest.	Min
Noise Delivery	The maximum noise output of the robot during runtime.	dB
Rest Time	Time required by the robot to reset between operations.	sec
Maximum Tip Speed	Maximum tip speed is the speed at which a robot can move in an inertial reference frame.	mm/sec
Force Output	Amount of force a robot can generate.	N
Terrain Traversability	How well does the robot travel on the certain surface.	Terrain m/s
Load Capacity	Load capacity is the maximum load that a manipulator can carry without affecting its performance.	kg
Overload capacity of the robot	Load capacity beyond the recommended specification.	kg
Reliability	Such as reliability can be expressed by Mean Time Between Failure (MTBF) or Mean Time To Repair (MTTR)	MTTF/MTTR

Table 9 - Proposed Structure Category

Structure	Description	Unit
Type of end effectors	Type of end effectors used	List of end effectors
Number of end effectors	Number of tips	#effectors
Slip Sensors	Does the robot have slip sensors?	Yes/No
Mounting Position	Possibility to attach the robot to surfaces in different orientations. (for example ground, wall, ceiling and mobile platforms).	List of surfaces
Tactile Sensors	Does the robot have tactile sensors?	Yes/No
Material Of robot	What material does the outer surface of the robot consist of.	List of materials
Degree of protection	Degree of protection denotes the protection provided by an enclosure against access to hazardous parts, ingress of solid foreign objects and water.	Protection Code (IP-Code)

Table 10 - Proposed External Requirements Category

External Requirements for robot	Description	Unit
Ambient Temperature	Range of temperature within which the robot functions as expected.	Range(^{o}C)
Space Requirements	How much space does the robot need to be able to function correctly.	Mm^3 (Length, Width, Height)
Recommended Operating Humidity	Range of humidity within which the robot functions as expected.	Range(%)

Table 11 - Proposed Automation Category

Automation	Description	Unit
Adaptability	Average duration to change the configuration of the robot to perform different operations or at different locations.	sec
Autonomy	The capability of an entity to create and control the execution of its own plans and/or strategies. For instance, in mobile robots the ability of the robot for determining the trajectory to reach a specific location or pose. See **Appendix E** for the definitions and the different levels of autonomy.	

Table 12 - Proposed Precision Category

Precision	Description	Unit
Sensing Range	Distance to recognise a predefined shape, image or event.	mm
Accuracy	Accuracy is the measure of closeness between the robot end effectors and the target point, and can usually be defined as the distance between the target point and the centre of all points to which the robot goes on repeated trials.	mm
Repeatability	Repeatability is the measure of the ability of a robot to return to the same position and orientation over and over again.	mm
Resolution	The smallest incremental motion which can be produced by the manipulator. Serves as one indication of the manipulator accuracy. Three factors determine the resolution: mechanical resolution, control resolution, and programming resolution.	mm
Stability	This is the measure of absence of oscillations at termination of arm movement.	mm
Compliance	This is the measure of displacement of the end effector in response to force (or torque) applied to it.	mm

Table 13 - Proposed Reach and Distances Category

Reach and distances	Description	Unit
Degrees of freedom	A term used to describe a robot's freedom of motion in three-dimensional space, specifically the ability to move forward and backward, up and down, and to the left and to the right	# axes
Horizontal Reach	Reach of the robot of the horizontal axis.	mm
Vertical reach	Reach of the robot of the vertical axis.	mm
Wrist reach distance	Maximum distance in which the end effector of the robot arm reaches in the work envelop	mm

3.7 Evaluation of the final Taxonomy

This section will discuss the evaluation of the final taxonomy. This section will show how the evaluation of the taxonomy is done and why it is done this way. Evaluation is needed to check whether the objectives of the artifact are met and to see if the theoretical-based developed artifact also can be used in practice. To recall the objectives for this taxonomy:

- The taxonomy should be as complete as possible.
- The taxonomy, attributes and corresponding definitions should be clear.
- The taxonomy should be easy to use.

Ideally, a formal evaluation method should be used to evaluate the taxonomy. A formal evaluation method enhances the evaluation, because the evaluation method is already validated. A well-known example of such a formal evaluation method is Moody's Method Evaluation Model(2003), in which 14 questions are asked covering perceived ease of use, perceived usefulness and intention to use. However, Moody's evaluation model is aiming at evaluating methods instead of taxonomies and does not cover anything about completeness and clarity.

Prat, Comyn-Wattiau & Akoka(2014) argue that all though evaluation pervades the Design Science Research, research regarding artifact evaluation is still at an early stage. They created a structured overview of possible evaluation criteria, using multiple different papers. The hierarchy of criteria that they have created shows which kind of criteria can be used for artifact evaluation. In line with the objectives of the created taxonomy, the criteria completeness, simplicity and clarity are also among the list of evaluation criteria.

Despite that Prat et al.(2014) cited the sources that they used to come up with this hierarchy, no formal evaluation form can be found within those sources. March and Smith(1995) proposed to use the criteria simplicity and completeness to evaluate artifacts, but give no validated evaluation model with questions to measure the completeness and simplicity of the artifact. Sonnenberg & Vom Brocke(2012) proposed clarity as a criteria for artifact evaluation, but also they don't show any validated evaluation model with questions to measure the clarity of the artifact. Since no formal evaluation method can be found regarding evaluating a taxonomy on completeness, simplicity and clarity, questions to evaluate the taxonomy regarding those criteria are created from scratch. The questions created will be shown in the remainder of this section.

To evaluate the attribute taxonomy and to check if the taxonomy can be used to describe an industrial robot and describing robot requirements for manufacturing tasks, two forms are created. Those forms are sent to robotic experts.

The first form is used to describe a certain robot. This robot form covers all the attributes that are included in the final taxonomy and the participant are asked to fill in all attributes as good as possible. After that, the participant is asked to answer 3 statements, one statement for each of the evaluation criteria. As mentioned before, those statements are created from scratch. The created statements are:

- Completeness: All aspects of the robot are covered by the attributes.
- Clarity: The attributes are clearly described and it is clear what the attributes mean.
- Simplicity: The taxonomy is easy to use.

The participant is asked to indicate how much the participant agrees with the statements, where the participant can choose between 1 until 5, where 5 means strongly agree and 1 means strongly disagree.

The second form is used to describe robot requirements for tasks. Only 27 attributes are added to this task form, because some of the attributes in the taxonomy are only there to compare the industrial robots and have nothing to do with task requirements. The attributes that are excluded from the task form are:

- Operation Cost
- Power Consumption
- Maintenance Cost
- Environmental Impact
- Operation Time
- Rest Time
- Maximum Tip Speed
- Reliability

The participant is asked to fill in only those robot requirements that the participant things are mandatory to have for a robot to be able to execute the task. When the participant filled in each robot requirement needed, he is asked to look at three statements, each representing one of the evaluation criteria. The three statements on this task form are:

- Completeness: All requirements that a robot need to perform the task are covered by the attributes in this form.
- Clarity: The attributes are clearly described and it is clear what the attributes mean.
- Simplicity: The form is easy to use.

Again, the participant is asked to indicate for each of the statements to what extend he agrees with it. The participant can choose from 1 until 5, where 5 stands for strongly agree and 1 stands for strongly disagree.

Furthermore, the participant is asked on both forms to fill in any remarks that he has regarding the form, unclarities of attributes or definitions, missing attributes and so on in the comment box. With this comment box, it is tried to get some more detailed information about the opinion of the participant regarding the taxonomy.

To make sure that the simplicity criteria is measured correctly, limited additional explanations of the taxonomy and the attributes is given to the participants. However, to make it quicker for the participant to fill in the forms, three additions should be made in both forms. Firstly, the table with the Levels of Autonomy(Beer et al., 2014) is added to both forms, because highly possible that the participant does not know this framework, while this framework is chosen to be the measure of the level of autonomy for this project.

Secondly, a table in which the IP-codes are explained is added to the forms.The IP-codes and the explanation of the code can be found anywhere on the internet. However to make it more convenient for the participant to fill in the form, an explanation of the IP-codes is added. Lastly, a description of all possible Degrees of Freedom is added to the forms, to make it easier to fill in the Degree of Freedom for people that have little knowledge about industrial robots and to make sure that everyone has the same understanding about Degrees of Freedom.

Both forms are added to Appendix F. Appendix F1 consists of the robot form, while Appendix F2 consists of the task form. The forms are also sent to the experts in this exact way. Both the robot and the task form are sent to five experts. However, the five experts to which the robot form is sent, are not exactly the same five experts as the experts to which the task form is sent. The robot form is sent to experts with a background in the robot manufacturing field, while the task form is sent to project managers within manufacturing companies in which industrial robots are or will be implemented to execute a certain task of their process. Only three experts respond on the robot form, while only two experts respond to the task form.

First of all, the comments retrieved by the robot form will be discussed. Three experts responded, of which one is an engineer at Kuka Robotics, one Research Scientist of FZI(Forschungszentrum Informatik) and one Research Scientist of Munich University of Technology, in the field of Robotics and Embedded Systems. The answers on the three statements are shown in table 14.

Table 14 - Scores of Experts on the statements for the robot form

Validation Criteria	Score Expert 1	Score Expert 2	Score Expert 3
Completeness:	2	5	3
Clarity:	5	4	4
Simplicity:	1	3	5

Clarity is scored high by all participants, which means that the participants find the descriptions of the attributes clear. However, the experts disagree on completeness and simplicity, since expert 1 scored both criteria low, while the other two experts scored the other criteria at least average. Expert one had a hard time scoring the attributes in the final robot attribute taxonomy, because he is used to different attributes in the technical specifications of robots. This experts often use comments like “Not in specification”. This shows that this expert had a hard time filling in the form, due to the gap between their specification sheet and the attributes asked. Expert two comments that he feels like that all aspects to describe a robot are included in the taxonomy. However, he thinks that some of the attributes are too specific and are therefore hard to answer. Also for this experts it can be concluded that he only uses the robot specification sheet to answer the different attributes, while attributes that are not on this specification sheet cannot be answered. Expert three feels that some specifications should be added, namely the weight of the robot, angular speed for each axis and the range of motion for each axis. On top of that, it can be concluded that also this expert had some difficulties with finding values for certain attributes. However, due to lack of more detailed comments on the different attributes, no further conclusions can be drawn from this form.

Two experts filled in the task form, which can be used to describe the robot requirements of the task. One expert is working at TRI(Thomas Regout International) and the other expert is working at Robert Bosch Espana Castellet. Both companies are involved in the HORSE project and the experts are project members of the pilots that take place in their company. Table 15 shows the scores that the experts give on each of the validation criteria.

Table 15 - Scores of Experts on the statements for the task form

Validation Criteria	Score Expert 1	Score Expert 2
Completeness	5	5
Clarity	4	4
Simplicity	3	4

Both experts score the completeness of the taxonomy with a five. Clarity is scored by both experts with a 4 and simplicity is scored a three and a four. The first expert argues that the precision related attributes are difficult to fill in, because it is difficult to differentiate the attributes. He commented the following: " I found the precision related attributes difficult to fill in. It is difficult to differentiate accuracy, resolution and repeatability. Isn't repeatability and resolution just subsets of accuracy? Perhaps a few examples will help the user to fill in these attributes". He argues for consolidating a few of those attributes or to give some examples to help the user filling in those difficult attributes. The expert states that the remaining attributes are easy to fill in for the task.

The second expert argues that all requirements that they are aware of, are included in the form. Some of the requirements are difficult to fill in. For requirements of noise delivery for instance, it would help if some sort of graph was included, in which the typical noise levels were described. On top of that, not everyone knows what slip and tactile sensors are. The last attribute which was difficult to fill in according to expert 2, is the material of robot. This requirement is a bit strange, since it is hard to define a requirement for a task for that attribute. The second expert comments that the stability is an important attribute for this specific task. Furthermore, the IP Code table and the level of autonomy table are nice additions, which help filling in the requirements.

This evaluation is conducted to check if the taxonomy needs some more adjustments or that the taxonomy is good enough as it is now. The evaluation shows that there is still a gap between the theoretical knowledge and the practical knowledge. Where all attributes are retrieved from theory, the practical experts do not know all of them that good. On top of that, they feel that some of the attributes are pretty difficult to fill in, since they do not illustrate this information about the robot on their robot specification sheets and therefore do not know what the attribute would be. It is hard to say if the taxonomy should be adapted further, or that the gap between theory and practice should be closed by changing the current way of describing robots on the specifications sheets, since those are not totally standard yet. To close this gap, however, a lot of knowledge about robotics and many connections within the robotics field is required.

It is clear that more effort is needed to close the gap, but since most feedback is rather positive and the author does not have the knowledge and connections required, the current taxonomy will be used in the remainder of the thesis to show how it can be used in practice and if and how it will enhance process performances by enabling Dynamic Resource Allocation.

4. A method to select the best assignee for a task

This section discusses the methodology towards a method to select the best assignee for a task. The robot attribute taxonomy, discussed in section 3, will be incorporated into this method. Two steps are required to come to this method. First of all, an algorithm is required, which decides if a robot can perform a certain task. This algorithm will be based on the robot attribute taxonomy. Next, this algorithm should be extended by deciding which of the robots that can perform that task, is the best assignee for that task. This extended algorithm will be the second contribution of this project.

4.1 Algorithm to decide which robot can perform which tasks

The first step towards the method to select the best assignee for a task is to create an algorithm which decides if a robot can perform a certain task. This algorithm is created by utilizing the robot attribute taxonomy and more specific, by utilizing the attributes, descriptions and the units of the taxonomy. The algorithm requires the robot requirements of the task and the specifications of the robots and should compare those. Despite that 35 attributes are included into the final robot attribute taxonomy, not all attributes will be used to compare robot requirements of tasks and robot attributes. The attributes that are excluded for the comparison of robot requirements and robot attributes are shown in the table 16, together with the reason why they are not used for this comparison.

Table 16 - Attributes Excluded from Comparing Requirements and Robot Attributes

Attribute	Reason for excluding
Operation Cost	Operation cost will be used to decide how much it will cost if a certain robot will perform a certain task.
Power Consumption	Power consumption will be used to decide how much power it will cost when a certain robot performs a certain task.
Maintenance Cost	Maintenance cost is a cost attribute which is fixed over time, as long as the robot has been active.
Environmental Impact	Environmental impact will be used to decide how much environmental impact a robot will cause when that robot will execute a certain task.
Operation Time	Operation Time will decide how long a robot will take to execute a certain task and will therefore be used to choose between robots.
Rest Time	Rest Time will decide how long a robot needs to rest between tasks, which will impact the time that a robot needs to perform the task and will therefore be used to choose between robots.
Maximum Tip Speed	Maximum Tip Speed will impact the time that a robot needs to perform the task and will therefore be used to choose between robots.
Reliability	Reliability shows the percentage of average down time of the robot. This attribute will impact the time that is needed to perform a task.

The remaining 27 attributes will be used in the algorithm that compares the requirements and the robot attributes. To be able to check if a robot is able to execute a certain task, it needs to be decided how each of the attributes and the corresponding robot requirements will be compared. Some of the robot attributes must be larger than the robot requirements, while other robot attributes must be smaller than the robot requirements. Table 17 shows for each of the 27 attributes, if the attribute must be larger or equal(≥), must be smaller or equal(≤), must be a subset of(⊂) or must be equal to(=) the robot requirements.

Table 17 - Attributes and the operators to compare them with the requirements

Attribute	Operator Used
Noise Delivery	≤
Force Output	≥
Terrain Traversability	≥
Load Capacity	≥
Overload capacity of the robot	≥
Type of end effectors	⊂
Number of end effectors	≥
Slip Sensors	=
Mounting Position	⊂
Tactile Sensors	=
Material Of robot	⊂
Degree of protection	≥ (Both numerical codes in the IP code)
Ambient Temperature	Lb Robot ≤ lb Req. and ub robot ≥ ub req.
Space Requirements	≥
Recommended Operating Humidity	Lb Robot ≤ lb Req. and ub robot ≥ ub req.
Adaptability	≥
Autonomy	≥ (Higher level of autonomy than required)
Sensing Range	≥
Accuracy	≤
Repeatability	≤
Resolution	≤
Stability	≤
Compliance	≤
Degrees of freedom	≥
Horizontal Reach	≥
Vertical reach	≥
Wrist reach distance	≥

The robot and the task form, discussed in section 4.2, will be used to describe both robot attributes as robot requirements. Using the robot form, the attributes of a certain robot will be filled in. For the task, only those robot requirements that are required to perform the task, will be filled in. This means that it is possible that for a certain task only one requirement should be met. Using the requirements, the attributes and the table of operators, Algorithm 1, the Algorithm requirementsMet is created. It is assumed that all requirements are hard requirements, which means that all requirements should be met.

Algorithm 1 (requirementsMet Algorithm): Decide if a robot can perform a task

```
A Create Boolean OutputBool = True
B For each Task.Requirement ≠ null
C     If (Task.Requirement = "Noise Delivery" &&
      Not(Robot.NoiseDelivery ≤ Task.NoiseDelivery) = True)
            OutputBool = False
            Break For each
      Else If (Task.Requirement = "Force Output" &&
      Not(Robot.ForceOutput ≥ Task.ForceOutput) = True)
            OutputBool = False
            Break For each
      Else If ……
      .
      . Covering all Robot Attributes like this
      .
D     End If
E End For each
F Return OutputBool
```

In line A, a new boolean is created, which is set as true. Next, in line B a loop is initialized in which all requirements of the certain task that are not empty, will go through the code of line C. In Line C, an if-statement is initialized. The if-statement covers all robot attributes and checks if the requirement of the attribute is met using the operators corresponding to the corresponding attribute according to table 17. After that, Line F will return the outputBool, which will be false when at least one requirement is not met by the robot for the certain task.

Imagine that for a certain task, only the requirement Load Capacity is needed. The Load Capacity of the robot should be 10kg, to meet the requirements. Next, imagine Robot A with a Load Capacity of 7kg. When the algorithm will run using Robot A and the task, the following if statement will be checked:

Not(Robot.LoadCapacity ≥ Task.LoadCapacity)

Since 7 is not larger or equal than 10, this statement returns true and will run the code in the if-statement. Therefore, OutputBool will be set to false and the for each loop will be broken. The algorithm will return false.

4.2 Extending the Algorithm with a Method to decide on the best assignee

Section 5.1 discusses the creation of an algorithm that decides if a robot can perform a task. This section discusses the extension of this algorithm, in which it is decided which robot will be the best assignee for the task. To get to this extended algorithm, a few things are required.

First of all, for each available robot in the system, it should be checked if the robot can perform the task. Therefore, Algorithm requirementsMet will be invoked once for each available robot, which will return false or true each time algorithm 1 is run.

Next, a ranking method should be added to the list of robots. To get the ranking method, the goal of the HORSE Project is used. The goal of the HORSE Project is to realize manufacturing tasks in an efficient manner(Horse, 2015). Efficiency is described as the factor between sacrifices that should be taken and sacrifices that have been taken. Sacrifices are described as "everything that we put in something to be able to get it done" (Veld, Veld, & Slatius, 2007). Money, time and materials, but also resources can be sacrifices. Money and time can be

computed, but materials are not in the scope of this project. Therefore it is decided to rank the robots based on both money and time. The outcomes of the final algorithm will be compared for both ranking methods. Therefore, Algorithm requirementsMet should be extended by adding a part, in which the expected time that a robot needs to execute the task is computed. Furthermore, the expected cost that a robot will approximately make to execute the task will be estimated.

The expected cost that a robot will make to perform a certain task can be estimated by utilizing the expected execution time of the robot for the certain task by using formula 4.1:

$$E_{cost}^{i,j} = E_{time}^{i,j} \; x \; OC_i \qquad (4.1)$$

Where,

$$E_{cost}^{i,j} = Expected\ Cost\ of\ Robot\ i\ Performing\ Task\ j$$

$$E_{time}^{i,j} = Expected\ Execution\ time\ of\ Robot\ i\ performing\ Task\ j$$

$$OC_i = Expected\ Operation\ Cost\ of\ Robot\ i\ in\ €\ per\ sec$$

Since the goal of this project is to enable dynamic resource allocation, it is assumed that $E_{time}^{1,j}$ for all tasks can be different. Therefore, to estimate $E_{time}^{i,j}$, several robot attributes will be used from the Robot Attribute Taxonomy. Table 18 contains all attributes that can be used to estimate $E_{time}^{i,j}$, together with the corresponding explanation why this attribute should be used for the estimation. Table 19 contains all robot requirements of the task that might be needed to estimate $E_{time}^{i,j}$. However, also less attributes can be used, depending on what attributes should be taken into account in a certain use case.

Table 18 - Robot Attributes needed for estimating $E_{time}^{i,j}$.

Robot Attribute	Reason for using in the estimation
Terrain Traversability	It is possible that the robot has to travel to get to the task. To calculate this travelling time, Terrain Traversability is used. Terrain Traversability is stated as speed over a certain terrain.
Maximum Tip Speed	Maximum Tip Speed needs to be used to calculate the time that the robot needs to get the tip of the end effector to the right position.
Adaptability	Adaptability is needed, to check whether a robot needs to change composition to be able to execute the next task. When a robot needs to change composition, the adaptability time is added to the execution time.
Rest Time	If the robot needs to rest before it can perform the next task, the rest time attribute should be added to the execution time.

Table 19 - Robot Requirements needed for estimating $E_{time}^{i,j}$.

Robot Requirements	Reason for using in the estimation
Needed Terrain Traversability	Needed Terrain Traversability is used to be able to calculate the distance that the robot has to travel over which terrains.
Horizontal Reach Needed	Horizontal Reach Needed will be used to know how far the tip should travel over the horizontal axis.
Vertical Reach Needed	Vertical Reach Needed will be used to know how far the tip should travel over the vertical axis.

Using these attributes and robot requirements, $E_{time}^{i,j}$ can be computed by formula 4.2, also known as *durationTask*:

$$E_{time}^{i,j} = \frac{2 * Needed\ Terrain\ Traversability}{Terrain\ Traversability} + \frac{2*(Horizontal\ Reach\ Needed + Vertical\ Reach\ Needed)}{Maximum\ Tip\ Speed} + Adaptability + Rest\ Time \quad (4.2)$$

When the robot should travel over multiple surfaces, the first part of the formula is used multiple times, where the total distance that the robot needs to travel is devided. It is assumed that when a task is executed, the Needed Terrain Traversability and the Horizontal and Vertical Reach Needed will be needed twice, since the task is only finished when the robot is back at its original position. Therefore, those three distances will be traveled twice. Furthermore, it is assumed that the tip will travel in only one axis at the same time. This means that the robot arm cannot travel in both horizontal and vertical axis. Last assumption is that adaptability time can be zero or larger than zero, depending on the type of task that will be executed and the setup of the robot at the start time of the task. If the robot has already the right setup for the task, the adaptability time will be zero, while if the setup of the robot is incorrect, an adaptability time is added.

The durationTask Formula is pretty standard and shall be only applicable to the most standard tasks, where the robot drives to one stop, lifts the tip, where after the tip will be dropped back down to the standard position and the robot drives back to the point where the robot started. When the task gets more complicated, for instance the robot needs drive to a second place to drop the package, the formula should be adapted to this more complicated situation.

Utilizing formulas 4.1 and 4.2, the expected execution time of a certain robot for a certain task and the corresponding costs can be estimated. Next step is to use those formulas to expand Algorithm requirementsMet. The extended algorithm is illustrated in pseudocode as Algorithm 2, the *auctionAlgorithm*:

Algorithm 2 (auctionAlgorithm): Decide which robot is the best assignee for a task

```
A List robotsAvailable, Task certainTask,
  Robot bestRobot = null, Var minCost = 999999,
  Var minTime = 999999
B For each robot in robotsAvailable
C    If Algorithm 1 = True (robot can perform the task)
D         Calculate E_time^(i,j) using formula 4.2
E         Calculate E_cost^(i,j) using formula 4.1
F         If (E_time^(i,j) < minTime Or E_cost^(i,j) < minCost)
               bestRobot = robot
               minTime = E_time^(i,j) Or minCost = E_cost^(i,j)
G         End If
     End If
  End For each
H Return bestRobot
```

Line A defines the list of all robots available, takes a certain task, clears bestRobot and sets both the minCost and minTime as very high. Next, a loop is invoked, where all robots that are in the list of available robots are used for one cycle. When the list of available robots is empty, the for each loop is not invoked, which results in a return value of null. Line C checks if the certain Robot is able to perform the certain task. When the robot can perform the task, Line D computes the Expected execution time using the DurationTask formula, while Line E computes the expected cost. Line F depends on the ranking that has been chosen. When cost is used to rank the robots, expected cost will be compared to the minCost. When time is chosen as the ranking criteria, expected execution time is compared to the current minTime. If the if-statement is true, bestRobot is set as the currentRobot and minTime or minCost is set as the cost of time of the current robot. When all robots in the list have been used in one cycle of the for each loop, the algorithm will show the best robot for that task. It is possible that no robot can execute the task and then the variable bestRobot is empty.

4.3 Verification of the method

To make sure that the auctionAlgorithm is working correctly, the Algorithm is validated. This verification is done by using the getting materials task of the hypothetical process, which has been explained in section 2. The verification should check if the requirementsMet algorithm works, in which is decided if a robot can perform the task. Two fictitious robots are created, Robot Large and Robot Small. The verification should show that if none of the robots is able to perform the task, no robot should be selected. If only one robot is able to perform the task, this robot should be assigned to the task. Lastly, when both robots can perform the task, the ranking methods should rank the robots and take the best robot.

Figure 6 shows the class diagram of this setup. The class diagram shows that one task is performed by zero or one robot, since it is also possible that no robot is able to perform the task. The class diagram also shows the Large and Small Robot and the corresponding robot attributes for both. The operation cost of the large robot is equal to c, which means that this attribute will be changed across the different scenarios of this verification. The diagram shows only the robot attributes that will be used during the verification to keep the diagram small. Furthermore, the diagram shows the example task and the requirements. Two requirements are variable and will change for the different scenarios of this validation.

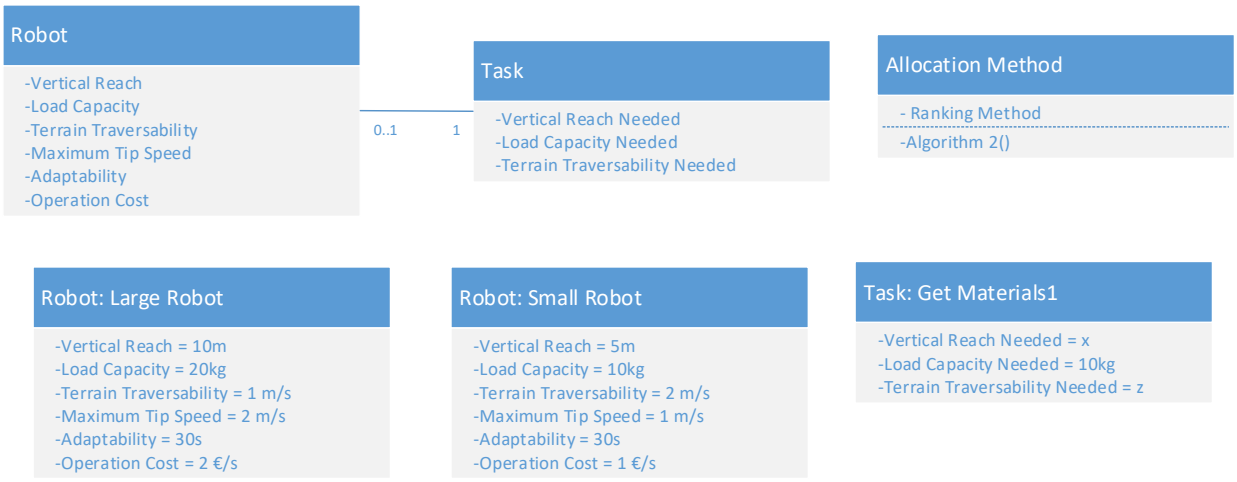

Figure 6 - Class Diagram of the Verification

Seven scenarios are created to verify the auctionAlgorithm, where in each scenario both ranking methods are checked. Hence, table 20 consist of 14 different combinations of the variables c, x, z and the ranking method. The first scenario shows a set of c, x and z in such a way that both robots do not met the requirements. Scenario two shows a set of c, x and z that results in only one robot that meets the requirements. In the remaining scenarios the variables are set in such way, both robots are capable of performing the task. Therefore, those five scenarios will show how the ranking methods will work.

Table 20 - Test values, expected outcomes and outcomes for Algorithm 2.

Scenario	C	X	Z	Ranking Method	Exp. Outcome	Outcome
1	1	12	10	Time	null	null
1	1	12	10	Cost	null	null
2	1	8	10	Time	Large Robot	Large Robot
2	1	8	10	Cost	Large Robot	Large Robot
3	1	4	3	Time	Large Robot	Large Robot
3	1	4	3	Cost	Large Robot	Large Robot
4	2	4	3	Time	Large Robot	Large Robot
4	2	4	3	Cost	Small Robot	Small Robot
5	1	3	4	Time	Small Robot	Small Robot
5	1	3	4	Cost	Small Robot	Small Robot
6	0,5	3	4	Time	Small Robot	Small Robot
6	0,5	3	4	Cost	Large Robot	Large Robot
7	2	5	2	Time	Large Robot	Large Robot
7	2	5	2	Cost	Large Robot	Large Robot

Table 20 shows that the auctionAlgorithm works as it should. The algorithm is capable of deciding which robots can perform a task. Furthermore, the Algorithm is capable of selecting the best robot for the task, based on two different ranking methods. The auctionAlgorithm is the second deliverable of this thesis.

5. Demonstration: Implementing the auction Algorithm into a existing Dynamic Resource Allocation Algorithm

Section 4 covers the methodology towards a method for deciding which robot is the best assignee for a task. The section also shows the auctionAlgorithm and also verifies this algorithm. This section covers the methodology towards a Dynamic Resource Allocation Algorithm in which the auctionAlgorithm is implemented. As discussed in section 1, the current Dynamic Resource Allocation Algorithm require that the user determines which robots can perform which tasks in advance, while customization and therefore versatility of tasks is increasing. By using the robot attribute taxonomy and the auctionAlgorithm, also dynamic resource allocation can be done with changing task requirements. So the auctionAlgorithm will cover the selection part of the allocation algorithm.

First step is to choose an existing Dynamic Resource Allocation Algorithm for robots. After that, the Robot Attribute Taxonomy and the auctionAlgorithm are implemented into this allocation allocation algorithm. Finally, a simulation is created to validate the extended allocation algorithm and to see how it would behave in practice.

5.1 Choosing the Dynamic Resource Allocation Algorithm

In this section, the Dynamic Robot Allocation Algorithm will be introduced. *Dynamic* task allocation is a class of task allocation in which the assignment of the resources, or in this specific case robots, to tasks is a dynamic process and may need to be continuously adjusted in response to changes in the task environment or group performance, of which changes in number of tasks(new arrivals) and number of robots(new robots or robots broken-down) are examples. (Lerman, Jones, Galstyan, & Matarić, 2006).

Extensive research has been conducted on Multi-Robot Task Allocation (MRTA) Algorithms. To be able to select an allocation algorithm in a structured way, the taxonomy of Task Allocation in Multi-Robot Systems(Brian P. Gerkey & Matarić, 2004) is used. This paper gives some structure to the different MRTA Algorithms by creating a taxonomy which can group the different algorithms. They propose three criteria for grouping the different algorithms into eight main groups as can be seen in figure 7.

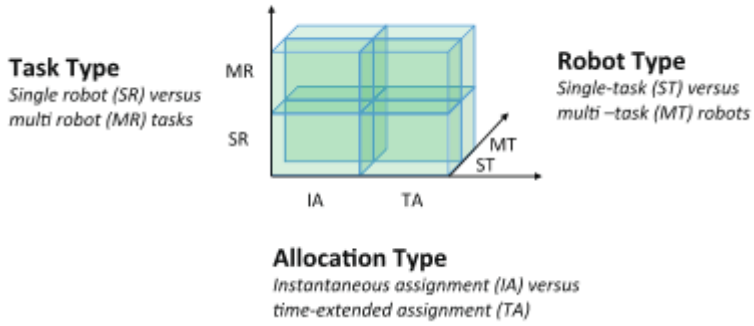

Figure 7 - Taxonomy of MRTA Algorithms

The first criterion is task type, which covers the number of robots required to execute the tasks within the process. When all tasks require exactly one robot, task type will be single robot(SR), while when at least one task within the process requires multiple robots, the task type is multi robot(MR). The second criterion is the robot type, which covers the number of tasks that the robots within the process are capable of executing simultaneously. When all robots can execute at most one task at a time, the robot type will be single-task(ST). Otherwise, when at least one robot is capable to execute multiple tasks at the same time, the robot type is multi-

task(MT). The third criteria is the allocation type, which covers how often the tasks will be allocated to the robots. The allocation type is Instantaneous assignment(IA) when the available information of the system(robots, tasks, environment) allows only one assignment, without any planning for future assignments. When such information is available, for instance the arrival of tasks over time, or when the system may consist of more tasks than robots, the allocation type is time-extended(TA).

In this project, it is assumed that robots will execute at most one task at a time and that a task requires just one robot. Also, it is assumed that it is possible that there are more tasks than robots at some time. Hence, this problem requires a ST-SR-TA algorithm according to the taxonomy.

Despite that the taxonomy is well-known and used many times (cited 1224 times), Gerkey & Mataric already argue that their taxonomy is not complete, since the taxonomy does not capture problems with interrelated utilities and task constraints. Korsah, Stentz and Dias(2013) agree with them and argue that the most important constraints missing are the precedence constraints, which may specify that one task must be performed before another. They are introducing a degree of interdependence to incorporate in the taxonomy of Gerkey and Mataric, which they named iTax. In iTax four degrees of dependencies are differentiated as can be seen in figure 8.

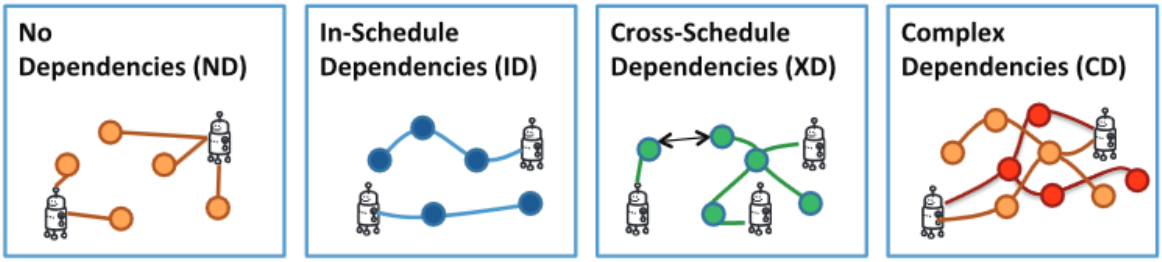

Figure 8 - Fourth dimension of the ITax

1. No dependencies (ND): No dependencies between any tasks.
2. In-Schedule Dependencies(ID): Dependent tasks will be executed by the same robot.
3. Cross-Schedule Dependencies(XD): Dependent tasks can be executed by different robots.
4. Complex Dependencies(CD): Dependent complex tasks that can be executed by different robots. Complex tasks are tasks that can be decomposed in multiple ways, which means that that the outcome of one subtask influence the next subtask.

According to iTax, the problem of this project can be described as XD [ST-SR-TA] problem, since different tasks might have dependencies and will be executed by different robots for optimization purposes. However, it is assumed that the tasks will be decomposed in just one way, which excludes the complex dependencies. Korsah et al(2013) also propose some solution approaches for this type of MRTA problem.

The first approach is the M+ system(Botelho & Alami, 1999). The system performs the allocation by a market system in which all robots that are not yet selected as best candidate for another task to make an offer on the task. The robot with the best offer will be the “best candidate”. The robot will be in the best candidate state until the robot starts on the task or when another robot has made a better offer for that particular task. After that, the robot will return to the so called evaluation state, in which the robot is able to send offers for tasks. The system deals with dependency by showing only the executable tasks, of which the

antecedents have been achieved. However, the M+-system is not showing how to decide in which order the executable tasks will be offered to the robots.

The second approach that is proposed is a variant of a market-based economy(MacKenzie, 2013). Several tasks are put up and the agents submits bids with cost expressed as a function of constrained variables such as location and time. After submitting all bids, the auctioneer uses a cost minimization algorithm to determine the allocations of tasks to the agents and the values of the constrained variables. The limitation of this approach is that it only supports instantaneous assignment.

The third approach is a robot routing problem corresponding to a geological scenario in which a team of robot must perform a set of goals(Chien, Barrett, Estlin, & Rabideau, 2000). Cross-schedule constraint are needed for the resources that need to be shared in the team. Three different approaches for this problem are presented.

The fourth proposed approach is an approach in which simple ordering constraints between tasks are used (Lemaire, Alami, & Lacroix, 2004). One of the tasks is auctioned to a robot, which makes this robot the “master”. This task determines the start time of the other connected tasks. The robots that get those other tasks assigned will be the “slave” robot, which means that this robot will follow every change that the master robot will make for this task sequence.

The last proposed approach for XD [ST-SR-TA] problems is again aiming at solving routing problems while assuming that the mobile robots have a limited communication range (Mosteo, Montano, & Lagoudakis, 2008). In the approach, the schedule of an individual robot is coupled with other team members in such a way that ensures connectivity for communication.

Of those five approaches, two approaches aim at solving routing problems and therefore, those will not be used in this project. Mackenzie’s approach is not documented in great detail, which makes it difficult to implement that approach. Furthermore, Lemaire et al. have also not worked out their approach in great detail. Hence, the M+ System will be used to integrate the auctionAlgorithm into a dynamic resource allocation algorithm. Section 5.1.1 explains the M+ System in greater detail.

5.1.1 M+ System

The M+ Task Allocation is a dynamic resource allocation algorithm, in which a task is only available when all its antecedent tasks are executed. All robots have the same set of ordered tasks available. How to order this set of tasks, is not specified by the authors of the approach. The approach uses multiple states in which the robots can be. Figure 9 shows the state diagram of the robots.

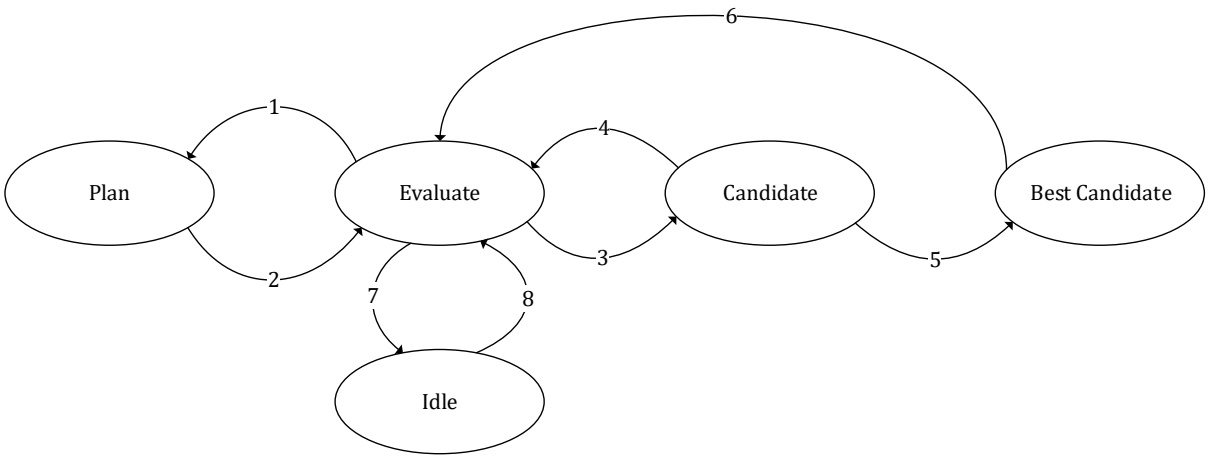

Figure 9 - The State Diagram (Botelho & Alami, 1999)

When a robot is in the Evaluate state, the robot will look for a task in the set of available tasks. The robot will start at the first task in the ordered list and will make an offer for this task. To create this offer, the robot will go to the plan state for a moment **(1)**. Next, the state of the robot will change back to evaluate, since the robot has his offer determined now **(2)**. Next, the robot will go into the candidate state **(3)**, where it is waiting for a response of the robot that organizes the communication around this specific task. Two responses are possible: The offer is rejected or the offer is accepted. When the offer is rejected, it means that the offer that the robot made, is not the best offer on this task. At this point, the state of the robot goes back to Evaluate **(4)**, in which he will continue making offers to other tasks in the available task set. When the offer is accepted, which means that it was the best offer for the task, the robot will go to the Best Candidate state **(5)**. The robot will change state from here when the robot starts executing this task. However, since all other robots can make offers for this task as long as the execution of the task is not yet started, a change of state will also occur when another robot makes a better offer for the task before the robot can start the task execution. In both cases, the state of the robot will change to the Evaluate state **(6)**. From the Evaluate state, the robot can change to the Idle state **(7)**. This state change only happens when a robot is rejected for all tasks in the current available task set. The Idle state will only change back to the Evaluate state, when a new task is added to the available task set **(8)**.

State change from Best Candidate to Evaluate happens when the robot starts executing the task. This implies that a robot has two spots available for tasks: One spot that contains the task that the robot is performing at this moment and the other spot that works as a queue. As long as the task is in the queue of the robot, other robots are allowed to make offers to the waiting task and can take this task away. This makes the algorithm dynamic, since tasks are allocated to robots beforehand, but this allocation might still change, depending on which robots are getting available in the meanwhile.

Some of the aspects of the M+ System are not explained extensively. An example of this is the sorting of the set of tasks, which is mentioned before. Also, the Plan state, in which the robot determines what the offer will be, is not explained in great detail. Botelho and Alami do not discuss how this offer is determined and do not even discuss what the content of an offer. Furthermore, it is not described what happens with the algorithm when a robot breaks down.

5.2 Towards the adapted M+-System

Section 5.1 discusses why the M+ System is chosen and explains how the M+ System works. The section also discusses the aspects of the algorithm that have not been explained extensively enough. This section discusses how the M+ System is adapted to meet the requirements of this project and how the aspects of the system that lack explanation are filled in.

First of all, a state should be added, in which the robot will change into when it is unavailable. In the current state-diagram, it is assumed that the robot is always operating, which is very unlikely in real-life. Therefore, the state “Unavailable” is added to the state-diagram. Theoretically, the robot can change to the unavailable state from all other states. However, since the robot is in the plan, the evaluate and the candidate states for only the computation time of the algorithm, it is assumed that the robot can only change to the unavailable state from either idle or the best-candidate state. It is assumed that the task that the robot is working on, will go back into the sorted set of tasks available. This ‘new’ arrival will change the state of all idle robots, since they now will bid on this task. Furthermore, the assumption is made that when this task will be performed by another robot, the new robot will have to perform the task from scratch. This implies that when a robot goes to the unavailable state, due to for instance a malfunction, and it is performing a certain task, the execution time that the robot has already spent on this task is lost. If the robot goes from the Best Candidate state to the unavailable state, the assumption is made that this task of which the robot was the best candidate will be removed from the robot. All robots in the Idle state will change to the Evaluate state and make an offer on this task. A robot that is in the unavailable state can only change states when it is repaired, since in this example it is assumed that the robot can only go to the unavailable state when it has a malfunction. When the robot is repaired, the status of the robot will go straight into the Evaluate state.

Secondly, the M+ System is using a sorted set of tasks. However, no explanation is given in which way the tasks should be sorted. Therefore, it is assumed in this demonstration that the tasks are sorted based on the arrival time, where the task with the earliest arrival time will be first in line and the task with the latest arrival time will be last. This corresponds with a First In, First Out (FIFO) queuing method. There are two types of arrivals: A totally new task and a task which gets available due to the completion of the antecedents. In the first option, it is obvious that the arrival time of the task is equal to the time that this task arrives. However, for the second option it is assumed that the arrival time of this task is equal to the arrival time of the first antecedent. This assumption makes sure that when the first task of a job is completed, the next tasks in that job are also performed as soon as possible.

Thirdly, the Plan state has not been explained extensively. When the robot is in the Plan State, the robot is determining its offer for a certain task. However, Botelho and Alami have not explained how this offer is determined. Moreover, they do not give any idea how and what the offer should contain. Since this project aims to implement the Robot Attribute Taxonomy into a Dynamic Resource Allocation Algorithm and for this the auctionAlgorithm has been created, the same criteria that are used in auctionAlgorithm will be used for the offers. This implies that, depending on the chosen ranking method, the offer will contain either time or cost needed to let that robot perform that task. Nevertheless, the auctionAlgorithm compares multiple robots and selects that robot of which the cost or the time needed to execute the certain task is the lowest. The offer part is already be incorporated into auctionAlgorithm. To make sure that a robot can determine the offer for a task, the part of the computation of the offer should be subtracted from the auctionAlgorithm. The subtracted part of the auctionAlgorithm is shown in Algorithm 3, also called the determineOffer Algorithm.

Algorithm 3 (determineOffer Algorithm): Determine Robot Offer for Task

```
A Task certainTask, Robot ThisRobot
  E_time^i,j = 999999, E_cost^i,j = 999999
B If Algorithm 1 = True (Robot can perform the task)
C    Calculate E_time^i,j using formula 4.2
D    Calculate E_cost^i,j using formula 4.1
E End If

G Return E_time^i,j Or E_cost^i,j (Depending on chosen ranking method)
```

In Line A, a task and a robot are specified. Furthermore, the expected time and expected cost are set on 999999. In Line B it is checked if ThisRobot can perform certainTask. When he can perform the task, the expected time and the expected cost are set in Line C and D, using formula 4.1 and formula 4.2. In Line G, expected time or expected cost is show, depending on the chosen ranking method. When a certain robot cannot perform a task, Line C and D are not invoked. This means that the expected cost or the expected time that is shown in Line G will be 999999. The probability that the outcome of formula 4.2 or 4.1 is really equal to this value, is negligible.

Lastly, in the M+ System the 'owner' of a task, which is the robot that made the first offer on a task, has to select the best offer for that task. When an offer arrives for the task, the owner has to accept or reject the offer. When the offer is accepted, the robot that made the offer will go to the best candidate state. Furthermore, the robot that has the previous best offer will change from Best Candidate to the Evaluate state. The comparison of the offers is already been tackled in the auctionAlgorithm. However, in this Algorithm it is part of a bunch of robots making offers to a task, where after the robot with the best offer is chosen. To enable this in the system, it is necessary to store the best offer as a variable of the task-object. Algorithm 4, also called the checkOffer Algorithm, will then compare the offers and accept or reject the offer.

Algorithm 4 (checkOffer Algorithm): Accept or Reject an offer for a task

```
A Task certainTask, Robot thisRobot, Var bestOffer
  Var offer

B bestOffer = Retrieve current best offer on certainTask.
  offer = Retrieve offer of thisRobot on certainTask

C If (offer < bestOffer && offer != 999999)
D    Set best Offer on certain Task as offer
     Allocate certainTask to thisRobot
     Return true
E Else
     Return false
F End If
```

Line A set the Robot and the Task, and initializes the variables bestOffer and offer. Line B sets the variables, where bestOffer is retrieved from the task-object and offer is retrieved from the robot-object. Line C compares those and when offer is smaller than the bestOffer and the offer is not equal to 999999, Line D will be invoked. Line D sets the bestOffer value of the task-object as the variable offer, allocates the task to the task and returns true. However, when the offer is not better than the bestOffer, Line E will return that the Offer is rejected by returning false.

Figure 10 shows the adjust state diagram. The red components of the diagram have been added to or changed in this state-diagram in comparison with the standard state diagram proposed by Botelho and Alami.

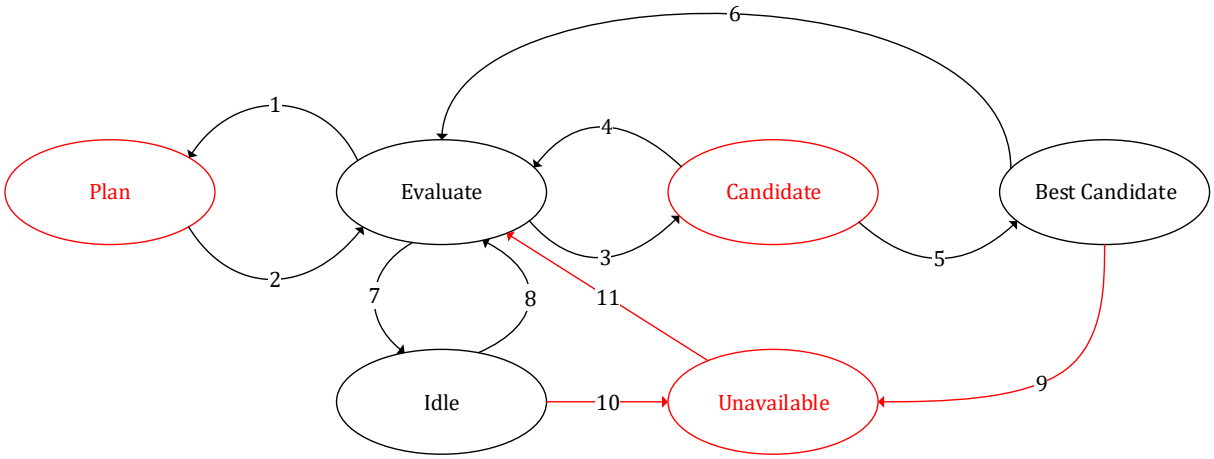

Figure 10 - Adapted State-diagram

When a robot has a malfunction when it is in the best candidate state, it will go to the unavailable state **(9)**. The working task and the best candidate task are removed from the robot and are added to the set of available tasks. When a robot get a malfunction while being idle, this means that there is no best candidate task and it is not sure if the robot has a working task. When the robot has a working task, this task will be removed from the robot and added to the available task list. The robot will change from the Idle-state to the Down-state**(10)**. When a robot is repaired, the robot will change from the Down-state to the Evaluate-state**(11)**. The robot will start making offers to available tasks. In the planning state, the determineOffer Algorithm is invoked to determine the offer of the robot for the certain task. Furthermore, to reject or accept an offer, the checkOffer Algorithm is invoked in the candidate state.

The adapted M+ System is the Dynamic Resource Allocation Algorithm that can be used for the purpose of this project. A simulation will be created to see how the allocation algorithm will run and what the effects of the algorithm are on a process and on the process performances.

5.3 Preparation steps for the Simulation of the adapted M+ System

Section 5.2 discusses how the M+ System is adapted to make sure that it can be used in this project. This section will discuss the simulation, which will be used to show how the allocation algorithm will be implemented in an information system and what the effects of the allocation algorithm are.

The process that will be simulated, is the hypothetical process introduced in section 1. The process consists of three tasks, where the first task will be performed for each job. After that, each job requires packaging. There are two possible packaging tasks and each job will needs one of the two tasks. The hypothetical process is small, but is meant as a subprocess in a larger business process. The hypothetical process is proposed, since it is easy to understand and can still show the effects of using the taxonomy within a resource allocation algorithm.

The goal of this simulation is to compare a traditional production process setting to a production process of Industry 4.0. More specific, the simulation compares a production process in which the allocation of the robots is static to a production process in which the allocation of the robots is dynamic. Furthermore, the allocation methods are compared in a process in which breakdown is included and in which breakdown in not included. Table 21 shows four different scenarios that will be compared to each other.

Table 21 - Simulation Scenarios

Scenarios	Without breakdown	With breakdown
Static Allocation of robots	Scenario 1	Scenario 3
Dynamic Allocation of robots	Scenario 2	Scenario 4

The different scenarios will be compared in several ways. However, the most important performance indicators to compare the four scenarios are the average execution cost, the utilization of the robots and the average throughput time of the jobs.

To show how the different scenarios differ from each other in a process model view, the process model of all four scenarios are illustrated in figure 11 until 14.

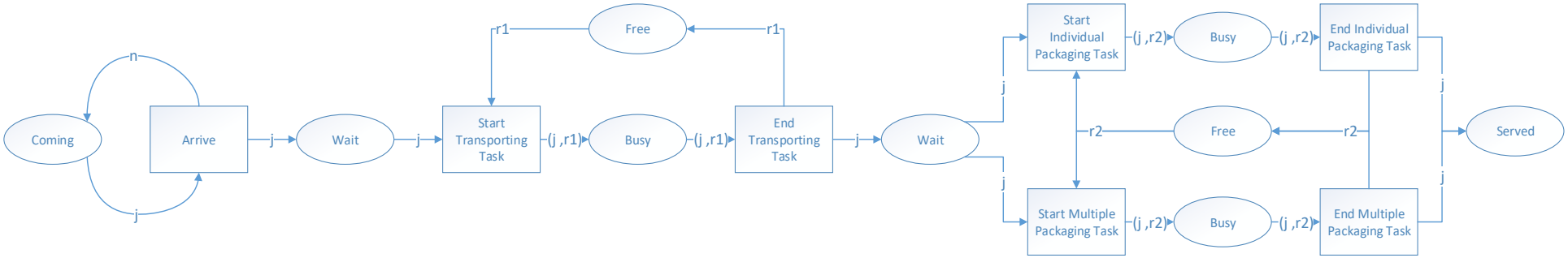

Figure 11 - Scenario 1

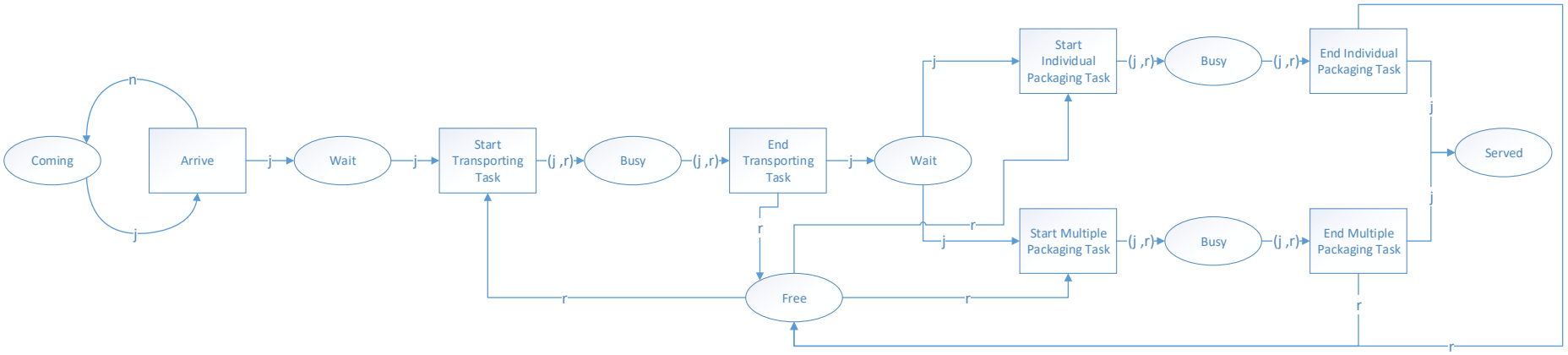

Figure 12 - Scenario 2

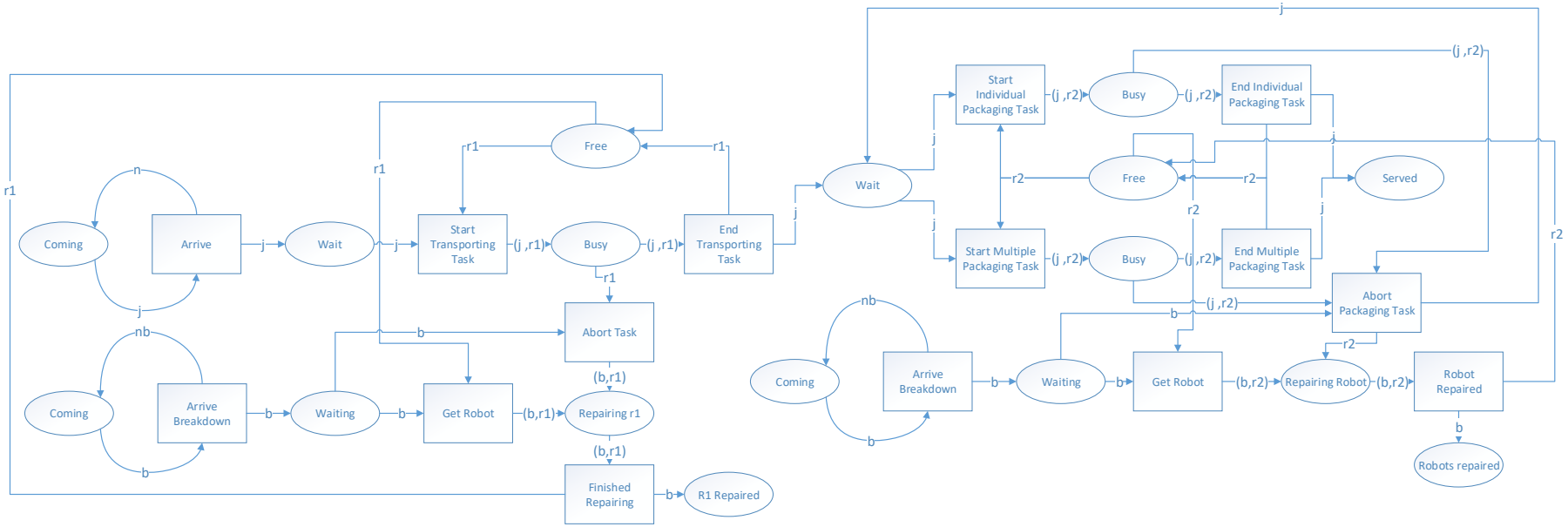

Figure 13 - Scenario 3

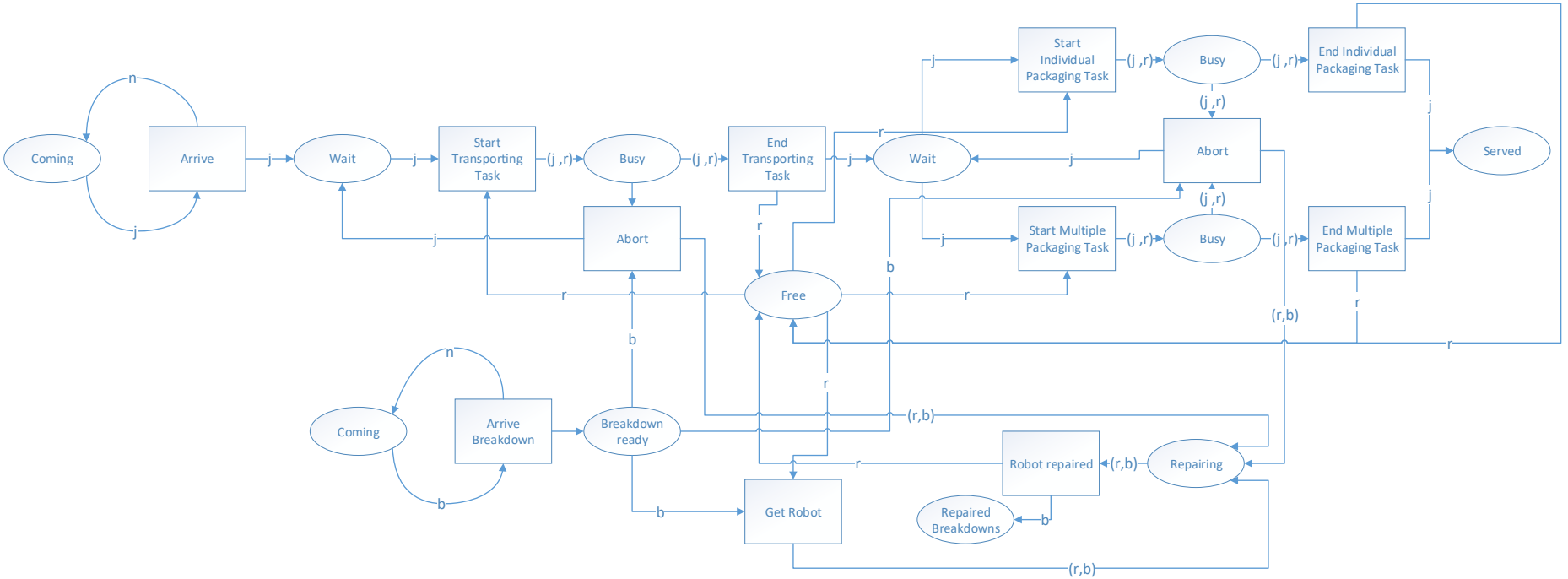

Figure 14 - Scenario 4

There are some similarities in all of the scenarios. First of all, the arrivals of the jobs is similar in all four of the scenarios. Each time the transition Arrive is fired, a new arrival is created. The arrival time of the new token is set as current time plus a certain interarrival time. In this simulation, it is assumed that the interarrival time of the jobs is uniformly distributed, with a minimum interarrival time of 40 seconds and a maximum interarrival time of 120 seconds. Furthermore, the ratio of jobs that will be packed individual in comparison to the multiple packaging jobs should be known, since this is also an attribute that the new token will have. It is assumed that 50% of the jobs will need multiple packaging, while the other 50% of the jobs needs individual packaging. Since the robot attributes and the requirements of the tasks determine the service time of the robot, the requirements of each of the tasks are specified. The distributions and the requirements are similar in all scenarios. The values of the requirements are randomly generated, to illustrate the variability in the robot tasks. It is assumed that the Transporting task requires the robot attributes and the corresponding distributions for those requirements, as shown in table 22.

Table 22 - Requirements Transporting Task and the Distribution

Requirements	Distribution
Load Capacity Needed (Weight of Material)	Uniform(1,50)
Vertical Reach Needed	Uniform(1,5) as integer
Terrain Traversability(Distance)	Uniform(10,50)

These requirements mean that a piece of material that is transported from the storage rack to the table, will weigh between 1 and 50 kgs. Furthermore, the height at which the material is stocked between 1 and 5 meters, since the storage rack is assumed to be 5 meters high. Lastly, Terrain Traversability is the distance that the robot needs to travel to get from starting point to the point of the material. The distance is uniformly distributed between 10 and 50 meters.

Table 23 shows the assumed requirements of the individual packaging task an the corresponding distributions of the values of those requirements.

Table 23 - Requirements Individual Packaging and the Distribution

Requirements	Distribution
Load Capacity Needed (Weight of Material)	Uniform(1,50)
Vertical Reach Needed	Uniform(0,5 ; 1)
Terrain Traversability(Distance)	(Uniform(1,4) as integer) * 2 + 10
Accuracy Needed	50
Horizontal Reach Needed	1

Load Capacity is again needed, since the material that is packed will be of the same weight as in the Transporting task. Vertical reach Needed is uniformly distributed between 0,5 and 1 meter. Terrain Traversability, also known as Distance, can be 12,14,16 and 18 meters, equally distributed. This distribution can also be computed by multiplying an discrete uniform distribution between 1 and 4, after which the outcome will be multiplied with two. Imagine this as containers, standing beside each other with a width of 2 meters. Since this is individual packaging, the accuracy and the horizontal reach needed are always the same, relatively 50 mm and 1 meter. This implies that the material is always placed in the middle of the container and that an error of 50 mm can be made, without any problems. The material should be placed in the middle of the container, to keep the container as steady as possible.

Table 24 shows the assumed requirements of the multiple packaging task and the corresponding distributions of the values of those requirements.

Table 24 - Requirements of Multiple Packaging and the Distribution

Requirements	Distribution
Load Capacity Needed (Weight of Material)	Uniform(1,50)
Vertical Reach Needed	Uniform(0,5 ; 2)
Terrain Traversability(Distance)	(Uniform(1,4) as integer) * 2 + 10
Accuracy Needed	10
Horizontal Reach Needed	Uniform(0,4;2)

The requirements of the multiple packaging are comparable with those of individual packaging. However, some of the distribution have changed slightly. First of all, the upper bound of the vertical reach needed is increased, since in multiple packaging, it is possible to pile some materials. The accuracy is decreased, which means that the robot should be more precise in placing the materials into the container. The reason for this is that more materials needs to get in the container, which makes it important to place the materials more accurate. Lastly, the Horizontal reach needed is changed. In the individual packaging task, the material is always placed in the middle of the container. Now, the material can be place anywhere in the container.

The requirements, together with the individual vs. multiple packaging can be combined. Therefore, the list of attributes that a job will receive upon creating a new token, together with the corresponding distributions are illustrated in table 25.

Table 25 - Attributes of a job

Job Attributes	Distribution
Arrival Time	TNow + Uniform(40,120)
PackagingType	Uniform(0,1)>0,5 → Multiple *else* Individual
Load Capacity Needed (Weight of Material)	Uniform(1,50)
Vertical Reach Needed(Transport)	Uniform(1,5)
Terrain Traversability(Distance) (Transp.)	Uniform(10,50)
Vertical Reach Needed(Packaging)	Multiple: Uniform(0,5 ; 2) Individual: Uniform(0,5 ; 1)
Terrain Traversability(Distance)(Packa.)	(Uniform(1,4) as integer) * 2 + 10
Accuracy Needed	Multiple: 10 Individual: 50
Horizontal Reach Needed	Multiple: Uniform(0,4;2) Individual: 1

Besides the similarity of the standard process, the arrival of new jobs and the distribution corresponding to the attributes that all jobs have, there are some differences in the process models. First of all, the difference between scenario one and two is the way that the robot tokens are used in the process. In the first scenario, token r1 can only be used in the transporting task transition, while token r2 can only be used in the packaging task transitions. In contrary, the process model of the second scenario shows that both robots can be used in all of the transitions.

Secondly, the difference between the first scenario and the third scenario is that break downs are incorporated into the process model of scenario 3. Table 26 shows the attributes and the corresponding distributions that are initialized when a new break down token is created in scenario 3.

Table 26 - Attributes for breakdown Tokens

Attribute	Distribution
Arrival Time	Tnow + Uniform(3600,7200)
Repairing Time	Tnow + Uniform(600,900)

The table shows that two attributes are added to a breakdown token in scenario 3. Each time a breakdown token arrives, a new arrival is planned using the formula for arrival time. The repair time will be determined in another transition. When an arrive transition is fired, also one token goes to the place breakdown ready. From here, one of the two possible transitions is activated immediately, depending on where the robot token is at that moment. At each of the transitions, the repairing time is determined using the formula as shown in table 26. When the token is at the place free, the transition 'get robot' is activated. The robot token is taken from the free spot and goes into the repair robot place, together with the breakdown token. However, when the robot token is at the place busy, the transition abort task is activated. When a task is aborted, the task that is aborted will go back to the queue, which is located at the wait place. The robot that is aborted, will go to the repair robot spot, together with the breakdown token. The robot is repaired, when TNow is equal to the repairing Time of the breakdown token. When the robot is repaired and the robot repaired transition is fired, the robot token will go to the free place, while the breakdown token will go to the robot repaired spot, where it will stay.

Lastly, when the first and the fourth scenario are compared, many differences can be seen. Scenario four is a combination of second and third scenario, where all robots can perform all tasks and where breakdown tokens are created. However, at the start of the simulation, two breakdown tokens are initialized, one for each robot. Hence, the attributes that the breakdown tokens have, will change slightly in this scenario. The attributes and the distributions are shown in table 27.

Table 27 - Breakdown Attributes Scenario 4

Attribute	Distribution
Arrival Time	Tnow + Uniform(3600,7200)
Repairing Time	Tnow + Uniform(600,900)
RobotID	[1,2]

To be able to know which robot will break, a robot ID should be added to the breakdown tokens. Each time a new breakdown token arrives and the arrive transition is fired, a new token is created in which the RobotID is equal to the RobotID of the just arrived token. The

breakdown token will behave the same as in scenario 3. However, the only difference is that the robotID of the robot that breaks down, is equal to the RobotID of the breakdown token. This implies that the breakdown token is only used by that robot, of which the breakdown token has the same RobotID.

The different process models for the four scenarios are discussed shortly. Next step is to create a UML-diagram, which will contain all important objects. The UML-diagram is shown in figure 15.

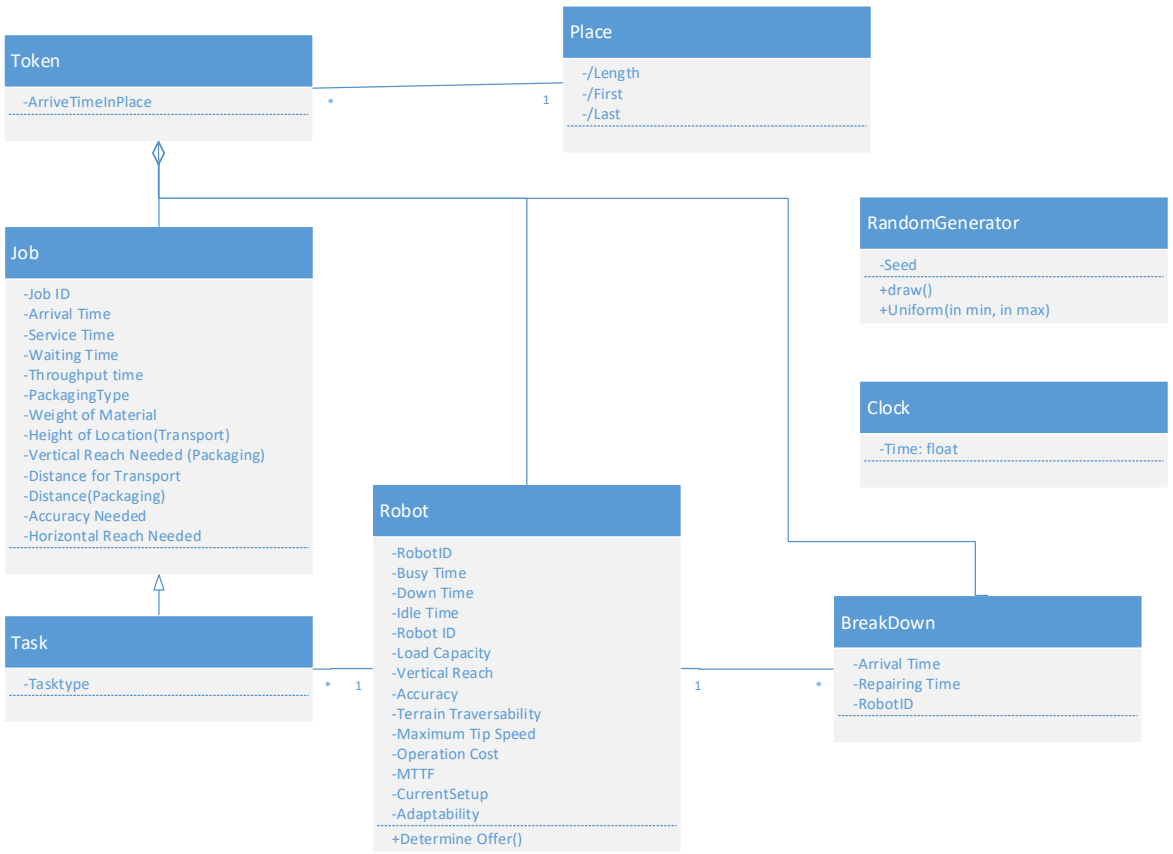

Figure 15 - UML Diagram of the system

The UML Diagram shows 8 different objects and their connections. First of all, a Job is the token that will go through the process. However, the robot will not perform the whole job, but only perform one of the two tasks of the job. However, the task will still have the same attributes as the Job has and is therefore an inherit of the Job object. Jobs, robots and breakdowns instances are tokens in the process model. Each token is located at only one place at the time, but multiple times can be located at the same place at the same time. A task is executed by only one robot, however a robot performs multiple tasks during the simulation cycle. A breakdown instance will only break down one robot, however a robot might break down multiple times. The robot object has the discussed attributes. Furthermore, the robot has a function in which the robot can determine his offer for a task. This function cannot be shown in any of the process models, but is important for the resource allocation algorithm that is implemented. Besides the three objects that can be tokens, also a RandomGenerator is needed, to generate values out of the uniformly distributed values. The last object that is shown in the UML-diagram, is the clock object. This object is needed to create the TNow.

5.4 The Simulation of the M+ System

Section 5.3 discusses the preparing steps for a simulation. Process models, objects and attributes for these attributes are discussed. This section will explain how the simulation is created and which classes, methods and algorithms are used in the simulation. The simulation is created in Java 8.

The key class in the simulation is the so called FES, short for Future Event Set. The FES contains only one attribute, which is an arraylist containing all scheduled events. This seems unusual, since the simulation contains stochastic distributions, like the arrival of jobs. When a job arrives, the arrival of the next job is already planned by utilizing the stochastic distribution of the interarrival time of the jobs. Hence, the events can be planned. Important is that the list of events is sorted based on the arrival time. The first event in the arraylist is also the event that will happen firstly. Each time a new event is created, this event is added to the FES sorted. Placing the new event in the right place in the arraylist is one of the methods that is specified in the FES class. Furthermore, a method is created that returns the first item of the Arraylist, where after the event is removed from the FES. The last method that is specified for the FES, is the method which returns the size of the FES. Figure 16 shows the class diagram of the FES, including the attribute and the methods.

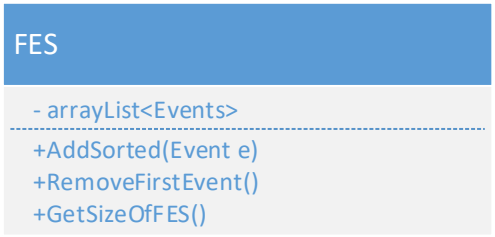

Figure 16 - Class Diagram FES

The Future Event Set uses Events. Event is another important class, since the simulation will invoke certain methods based on the type of the event. Therefore, the first attribute of the event is the event type. Six types of events are used in this simulation:

1. Arrival of a job
2. Transport task finished
3. Individual Packaging finished
4. Multiple Packaging finished
5. Breakdown of a robot
6. Robot repaired

On top of the type, an event has two different attributes of time, which is the time that an event is created and the arrival time of the event. Based on this arrival time, the events are sorted in the FES. The time that an event is created is important, to be able to know how long a certain event took. To know, for instance, the time needed to repair the robot, the start time of the event should be subtracted from the arrival time. To know about which robot the event is about, also an attribute robot is specified. Every time an instance of an event is created, some of the attributes are initialized. Depending on the type of task, different attributes are initialized. When the task type is arrival of a new job, only the arrival time and the task type are specified. In all other event types, also the robot and the start time are specified. The methods of the event-class are only returning the attributes of the instance. So for each of the four attributes, a method is method is created which will return the value of the attribute of the specific event instance. Figure 17 shows the class diagram of Event.

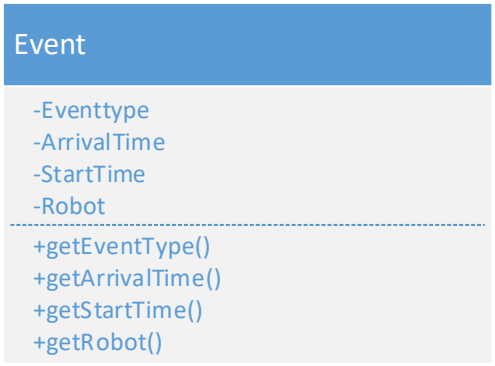

Figure 17 - Class Diagram Event

In the event-class, the class robot is introduced. Robots are the resource that execute the tasks of the process model and are involved in all events excluding the arrival event. Most of the attributes of the Robot class are attributes that are also included in the Robot Attribute Taxonomy and are needed for the different Algorithms. The attributes load capacity, vertical reach, accuracy, traversability, maximumTipSpeed, operationcost and adaptability are attributes that are required for the Algorithms, as have been discussed in section 4. On top of that, the attributes failureUpperBound and failureLowerBound hold the upper and lower bound for the uniformly distributed time to failure. There is one attribute of the robot that will not change throughout the simulation, which is the robotID. This attribute is unique for each robot. To compute the time that a robot is working and the downtime of the robot, also the attributes TotalTimeBusy and TotalDowntime are introduced. To know if the robot should adapt, also an attribute Setup should be introduced. The Setup of the robot can have three values: transporting, Individual packaging and Multiple packaging and stores the current setup of the robot.. Furthermore, the class should contain an attribute that checks if the robot is working at any given time and which task that the robot is performing. This is done by introducing an attribute taskWorking. Lastly, the class should also contain an attribute that checks if the robot has made a best offer at one of the tasks in the available task list. Therefore, an attribute taskWaiting is introduced.

For each of the attributes a method is created which returns the value of the attribute. The attributes taskWorking, taskWaiting and status will change throughout the simulation and therefore each of them gets also a method which sets the value of the attribute. TotalTimeBusy and TotalDowntime are attributes that are adding up all periods in which the robot works and in which the robot is broken. Therefore, a method which adds a certain value to the current attribute is used to add up those timeperiods.

Figure 18 shows the class diagram of the robot. The diagram shows only one get-method, since all get-methods are similar. The only difference is that each of the get-methods returns the value of a different attribute. On top of that, the two robots that will be used in the simulation, are created as instances. The instances are illustrated in figure 19.

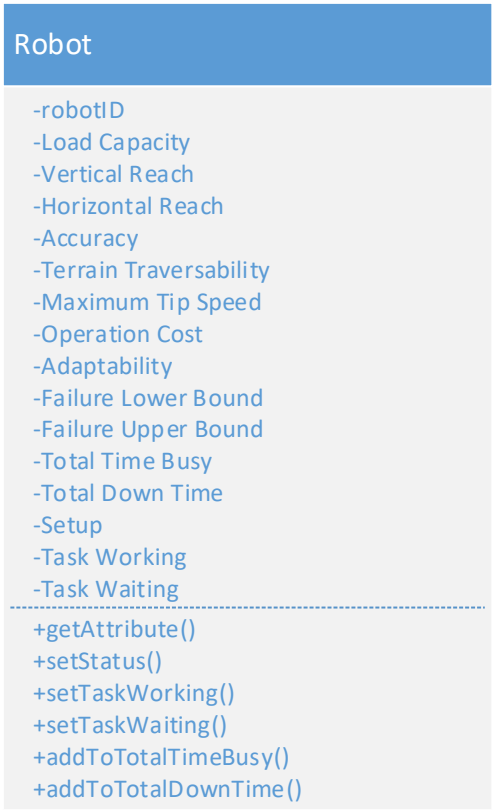

Figure 18 - Class Diagram Robot

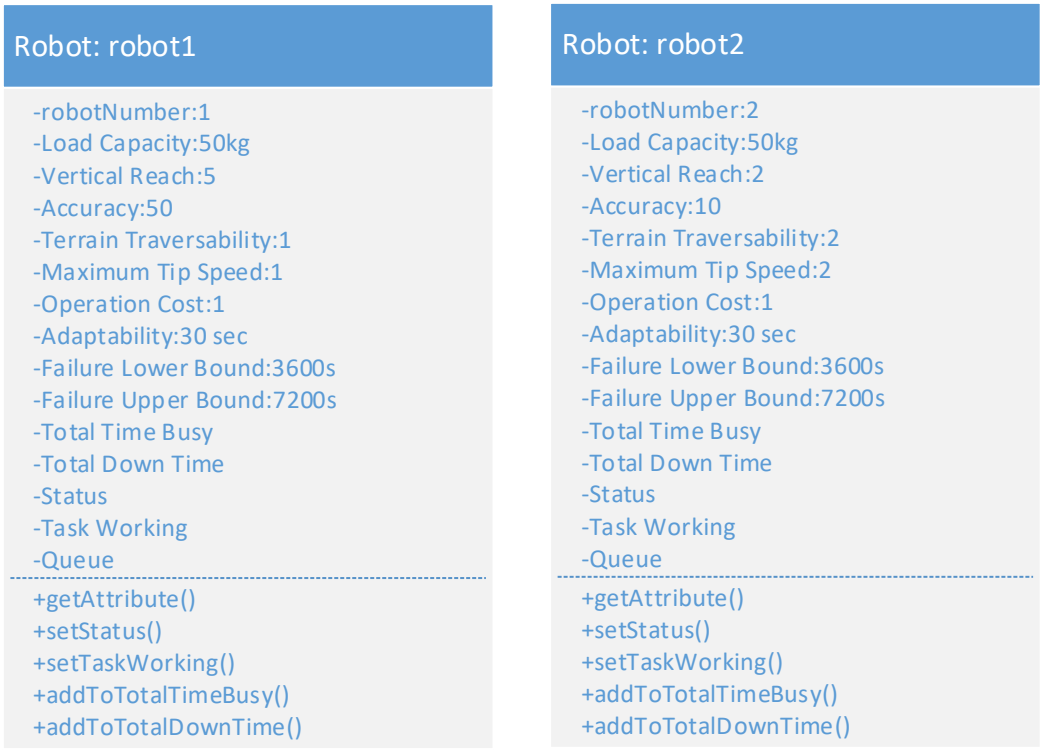

Figure 19 - Instances of the robot Class that will be used

Some attributes of the robot class already mentioned the Task class. The task instances are used together with the robot instances to run the Algorithms, determining which robot is the best assignee for a certain task. Therefore, the task class will contain the robot requirements, as introduced by the Robot Attribute Taxonomy. Those requirements are covered by the class attributes of weight, verticalReachNeeded, accuracyNeeded, distanceTransport, distancePackaging, PackagingHeight and PackagingWidth. On top of that, the tasknumber, tasktype and the singleOrMulti attribute are specified. Also the task has attributes which show to which robot the task is assigned and which robot made the best offer until now on the task. To store the best offer, one attribute is made which shows the best cost offer, while another attribute shows the best time offer. Furthermore, probably the most important attribute of the task class is the timeArrival of the task. Based on this attribute, the task is ranked in the Available task list. To be able to record the Performance Indicators of the system, the task also needs attributes for the timeExecution and the timeFinishedTransport.

Again, all of the attributes will have a method, which will return the value of the attribute. The attributes TaskType, RobotAssignedTo, RobotWithBestOffer and Time Finished Transport have also a set-method. The only method that is slightly different, is the calculate time Execution method. In this method, the execution time is added after each performed task. Figure 20 shows the class diagram.

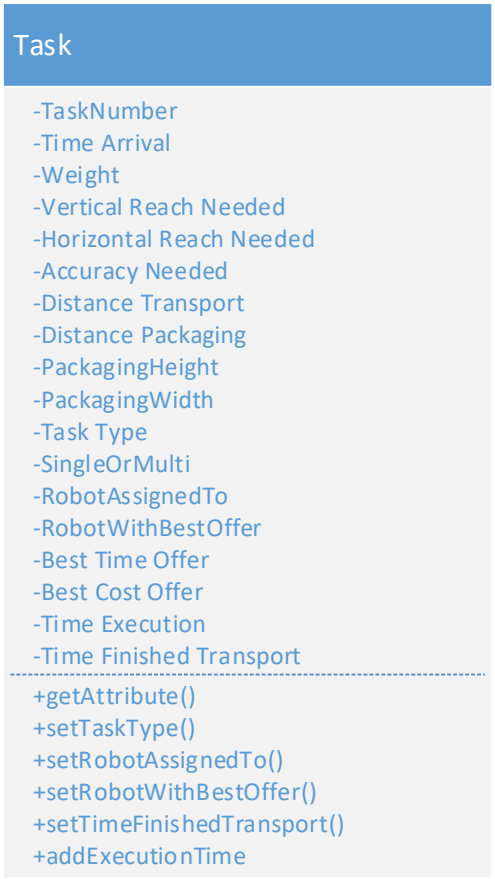

Figure 20 - Class Diagram Task

To make sure that the tasks are performed using the FIFO principle, a queue is needed in which the tasks are added sorted. This Queue is also a class, which has to be defined in the java program. The queue class is pretty similar to the FES class, since also the Queue class has only one attribute, which is also an arraylist. However, in contrary to the arraylist in the FES class, this arraylist consists of Tasks. The class has four methods: One method that returns the size of the queue, while another method returns a task, based on the index number asked. Furthermore, a method that adds tasks in a sorted way to the arraylist and a method that can remove a certain task are required. The class diagram of the Queue task is illustrated in figure 21.

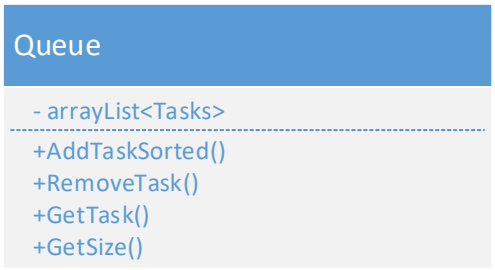

Figure 21 - Class Diagram Queue

The last class defined in the simulation is the main class of the program. The TaskAllocation class invokes and uses all other classes to run the simulation as needed. The class has three attributes, of which two are Arraylists. The third attribute is a random generator. The array lists keep track of which robots are free and which are fully occupied. A robot is free when the attribute “taskBestOffer” is empty.

The taskAllocation class is the main class of the program. This means that this class has a main-method, in which several other methods are created. The methods that are created inside of the main-method, will be explained in greater detail. The first method is the sampleUniform method, in which a random number is drawn, using the uniform distribution. To invoke this class, the minimum and maximum of the distribution should be inserted. The next method is the requirementsMet-method, in which it is checked if a certain robot meets the requirements of a certain task. The method returns true of false, where true means that the robot meets the requirements and false means that the robot does not meet the requirements of the task. The next method determines the expected duration of the task and returns this value. This method is comparable with formula 4.2, discussed in section 4.2. Next method is the auction Algorithm method, which is used when a new task arrives. The auction Algorithm checks for each of the available robot the expected time and cost that the robot needs to perform the certain task, where after the algorithm will decide, based on either time or cost, which robot would be the best assignee. Section 5.2 introduces the checkOffer Algorithm, in which an offer of one robot is rejected or accepted. This Algorithm is used in the method CheckIfRobotCanMakeBestOffer method.

Also, a method is created which decides which of which type a certain task is. The method will return this type. The most important, but also the most extensive method of the simulation, is the simulate method. The simulate method does the actual work of the simulation. The method first initializes the simulation by creating some starting events, where after an extensive while loop is used. In this while loop, each type of event is treated differently, using an if-statement. Each type of event has a corresponding treatment. The while loop will keep on running as long as the more than 0 events are in the FES. The simulation method will return an instance of simResults. Figure 22 shows the class diagram of the TaskAllocation class.

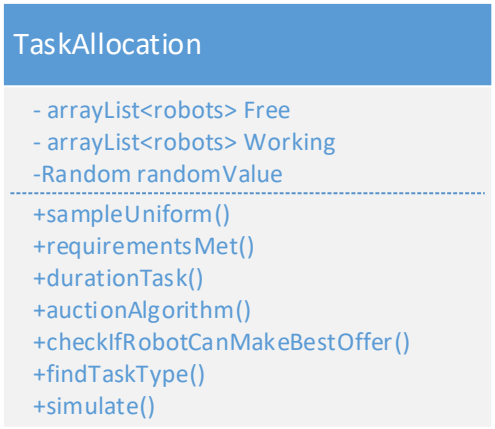

Figure 22 - Class Diagram TaskAllocation

The simulate method of the main class will be explained in greater detail, since this method is the core of the simulation. First of all, an initialization phase is needed to create the first events. In the first and second scenario, only an arrival event is initialized, while in the third and fourth scenario, also one breakdown event is created for each of the robots. Furthermore, a time variable t is initialized as 0 and the Queue of the idle tasks is created. Next, a while loop looks if there are any events left in the FES. As long as the number of events in the FES is more than 0, the while loop will keep on going. In scenario three and four however, the while loop will keep on going when more than two events are in the FES, due to the events of the breakdowns. In each loop of the while loop, the first event of the FES is taken. The time variable t is now equal to the arrival time of this event. Since there are six different types of events, also six different if-blocks are created. The pseudocode for that shows illustrates this code is shown next.

```
Simulate(): The simulate method in which everything is simulated
A Add First arrival token, and possible break down tokens to
  FES. If break down tokens, var addingEvents = 2, else
  addingEvents = 0
B Set t = 0, Queue idleTasks, FES fes
C while (FES.Length> (0 + addingEvents)
      Set Event e = FES.nextEvent()
      Set t = e.getTime()

D     If e.getType() = Arrival
            See Arrival Event Pseudo
E     If e.getType() = FinishTransport
            See Finish Transport Event Pseudo
F     If e.getType() = FinishPackSingle
            See Finish Packaging Event Pseudo
G     If e.getType() = FinishPackMulti
            See Finish Packaging Event Pseudo
H     If e.getType() = BreakDown
            See Break Down Event Pseudo
I     If e.getType() = FinishRepairingRobot
            See Finish Reparing Robot Event Pseudo
J End While Loop
K return Results
```

Line D until I represents how the code will deal with different eventtypes. Since the codes used in each of those six blocks is pretty extensive, the pseudocode corresponding to each of the blocks is shown separately, in the following pseudocodes.

```
If Event type = Arrival
A Create new Transport Task, with Arrivaltime = tnow
B If (Robots are Available)
C     Robot r = run AuctionAlgorithm(Transport Task)
D     If r = null
            Add Task to Available Task Queue Task
E     Else
F           If r is not Working yet
                  R Starts working on Transport Task
                  New Event Finish Transport is created
G           Else
                  Put Task in Queue of Robot r
                  Add Task to Available Task Queue
                  Change Availability of Robot to false
            End If
      End If
H Else
      Add Task to Available Task Queue
  End If
I If TNOW < Maximum T
      Create New Arrival Event
```

In Line A, a new transport task is created. When robots are available, Line C until G are invoked. Line C determines which robot is the best assignee for the task by running the auction Algorithm(Line C). When there is no robot that can perform the task, the task will only be inserted into the Available task queue. When a robot is able to perform the task, it is checked if this robot is already working on another task. When the robot is not working, the robot will start performing the task immediately. Therefore, also an new Finish Transport event is created. When the robot is working on another task already, the new Task will go in the Queue of the robot and also in the Available task queue. Also, the robot will not be available anymore, since the robot is now in the Best Candidate state. When no robots are available in Line B, Line H is activated. In here, the task will only be added to the Available Task Queue. Nevertheless, each time this if-block is invoked, it is checked if a new arrival event should be created. A new arrival event is created when the specified maximum time is not yet passed.

The following Pseudocode will show what happens when the event is of type Finished Transport Task.

```
If Event type = Finish Transport task
A  Remove Allocation of OldRobot and OldTask
B  Prepare OldTask for new allocation
C  If Oldrobot has a task in Queue
D      Start Performing this Task
       Create New Event, Type based on Tasktype.
E      Robot is Available Again
F      Search Available Task Queue For new Task
G      If A new Task is Found
            Robot is not Available Anymore
            Assign Task to Robot as BestCandidate
            Check if this task was already Allocated
H           If Task was Already Allocated
                 Take previous Allocated Robot, Redo from Line E
            End If
       End If
   End If

I  If (Robots are Available)
J      Robot r = run auctionAlgorithm(OldTask)
K      If r = null
            Add OldTask to Available Task Queue Task
L      Else
M           If r is not Working yet
                 R Starts working on OldTask
                 New Event TaskType is created
N           Else
                 Put OldTask in Queue of Robot r
                 Add OldTask to Available Task Queue
                 Change Availability of Robot to false
            End If
       End If
O  Else
       Add OldTask to Available Task Queue
   End If
```

Line A removes all allocations between the task and the robot. After that, the task is prepared for the next allocation. This is done by changing the tasktype to Multiple or Individual Packaging, depending on the MultiVSIndividual attribute of the task. Line C checks if the oldrobot has a task in the queue. When the robot has a task in the queue, the robot starts performing this task and create a new Event, of which the event is determined based on the type of the task. Line E makes sure that the Queue of the Robot is empty again and makes the robot available. In Line F, the robot will check if he can make the best offer to any of the tasks in the Available task queue. If a new task is found, the robot is not available anymore and the task that has been found, is placed in the queue of the robot. After that, it is checked if this new task was assigned to another robot previously. When the task was assigned to another robot, go back to line E and start from there with this other robot. For the oldtask, it is checked if there are any available robots. The code on Line I until Line O is the same as the code on Line B until Line H of the Arrival Event.

The following Pseudocode shows how a Finish Single Packaging Task is treated. Furthermore, is the pseudocode is exactly the same for the Finish Multiple Packaging Task. Therefore, the pseudocode is illustrated only once. The code is pretty similar to the pseudocode of the event Finish Transport Task. The difference is that the second part of the Finish Transport Task is removed and that Line B is removed.

```
If Event type = Finish Packaging task
A Remove Allocation of OldRobot and OldTask
C If Oldrobot has a task in Queue
D    Start Performing this Task
     Create New Event, Type based on Tasktype.
E    Robot is Available Again
F    Search Available Task Queue For new Task
G    If A new Task is Found
          Robot is not Available Anymore
          Assign Task to Robot as BestCandidate
          Check if this task was already Allocated
H         If Task was Already Allocated
               Take previous Allocated Robot, Redo from Line E
          End If
     End If
  End If
```

The next pseudocode shows how a breakdown event is treated. This event can only happen in the third and fourth scenario. To make sure that only those scenarios break down is implemented, the two initialization events for breakdown are not created in the first and second scenario. Without a first breakdown event, no breakdown events will be created.

```
If Event type = Break Down
A Change Availability of Robot to false
B If Robot is performing a task
C     Stop Performing Task
      Remove Allocation of Robot and Task
      Remove Future Event of for finish
D     If (Robots are Available)
E           Robot r = run auctionAlgorithm(Task)
F           If r = null
                  Add Task to Available Task Queue Task
G           Else
H                 If r is not Working yet
                        R Starts working on Task
                        New Event TaskType is created
I                 Else
                        Put Task in Queue of Robot r
                        Add Task to Available Task Queue
                        Change Availability of Robot to false
                  End If
            End If
J     Else
            Add Task to Available Task Queue
      End If

K     If Robot has Task In Queue
            Remove Allocation of Robot and Task
L           If (Robots are Available)
M                 Robot r = run auctionAlgorithm(Task)
N                 If r = null
                        Add Task to Available Task Queue Task
O                 Else
P                       If r is not Working yet
                              R Starts working on Task
                              New Event TaskType is created
                              Remove Task From Available Task Queue
Q                       Else
                              Put Task in Queue of Robot r
                              Change Availability of Robot to false
                        End If
                  End If
            End If
      End If
  End If
R Create new Finished repairing Event
```

The code that deals with break downs of robots is mainly trying to find another robot which can perform the tasks. If no other robot can perform the tasks, the tasks will go or stay in the Available Task Queue. Furthermore is the robot not available anymore and created the code a new finished repairing event.

The last piece of the simulate method, is the part of the code that will deal with a Finished Repairing event. The next pseudocode will show this part.

```
If Event type = Finish Repairing Robot
A Robot is Available Again
B Search For
C     Start Performing this Task
      Create New Event, Type based on Tasktype.
E     Robot is Available Again
F     Search Available Task Queue For new Task
G     If A new Task is Found
            Robot is not Available Anymore
            Assign Task to Robot as BestCandidate
            Check if this task was already Allocated
H           If Task was Already Allocated
                  Take previous Allocated Robot, Redo from Line E
            End If
      End If
  End If
```

5.5 Results of the Simulation

This section shows the results of the simulation. To compare the four scenarios, each scenario will be simulated. It is decided that one run has the length of 24 hours, since the simulation will be stable for some time at this point in time. This means that after t = 24*3600, no more tasks are arriving. The tasks that are already in the system will be executed. Since parameters of the simulations are stochastic, each run of the simulation will have a slight different outcome. By running the same scenario multiple times, an average of the output variable and a halfwidth can be computed. Stability of the output variables increases when the number of runs increases. Therefore, 20 runs are used to compute the output variables in each of the scenarios. No warm up period is used. The runlength and not using a warmup period is validated by comparing the results of the run of length 3600*24 with run with length 3600*120. The accuracy of the KPI's increases in the longer run, but the 95% confidence interval of the longer run is still within the range of the run of 24*3600 secondes. This shows that the run length is long enough to return correct KPI's

Table 28 shows the different scenarios with the corresponding output variables. The first and second scenario don't have any downtime percentages for the robots, since downtime is excluded in those scenarios. Each of the output variables has an average and a halfwidth of 95%, computed over 20 runs.

Table 28 - Output of the four scenarios

Averages	Scenario 1	Scenario 2	Scenario 3	Scenario 4
End time run	86548s +/- 31s	86543s +/-12s	86890 +/- 141	86607s+/-58
Throughput time	123,35 +/-1,8s	115,49 +/-0,7s	556,04 +/-40,3s	280,76 +/-8,4s
Utilization R1	0,825 +/-0,0047	0,642 +/-0,0054	0,822+/-0,0036	0,687 +/-0,0075
Utilization R2	0,413 +/-0,0025	0,571 +/-0,0041	0,411+/-0,0023	0,566 +/-0,0054
Downtime % R1	0	0	0,122+/-0,0031	0,121+/-0,0027
Downtime % R2	0	0	0,120+/-0,0030	0,119+/-0,0023
Cost Transport	66,03 +/-0,308	60,92 +/-0.284	66,02 +/-0,281	62,32 +/-0,357
Cost SP	33,45 +/-0,234	36,35 +/-0,240	33,47 +/-0,246	37,70 +/-0,295
Cost MP	32,59 +/-0,325	35,68 +/-0,237	32,59 +/-0,252	38,68 +/-0,260

The table shows the effect of the introducing the dynamic resource allocation algorithm, in which the Robot Attribute Taxonomy is incorporated. The differences of the performance of the four scenarios will be discussed in the remainder of this section. Figure 23 highlights the average throughput time in the different scenarios, while figure 24 highlights the average cost per job in the different scenarios. On top of that, figure 25 emphasizes the busy time of the two robots in the different scenarios.

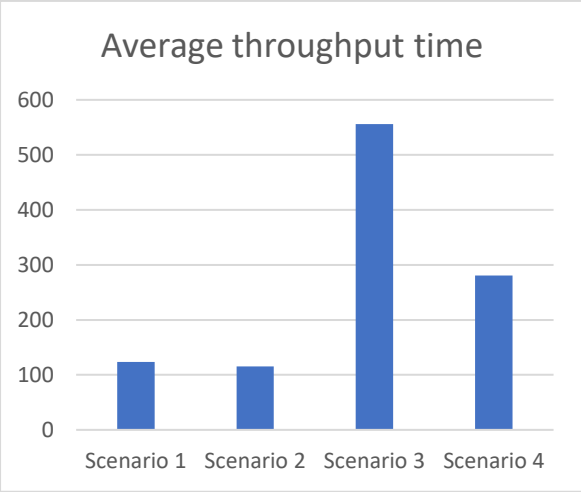

Figure 23 - Average throughput time

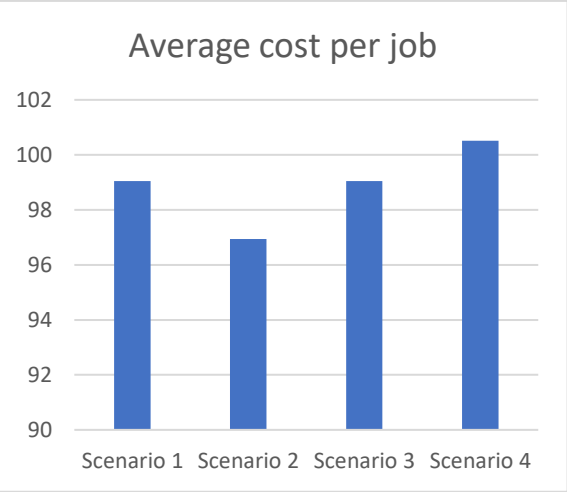

Figure 24 - Average cost per job

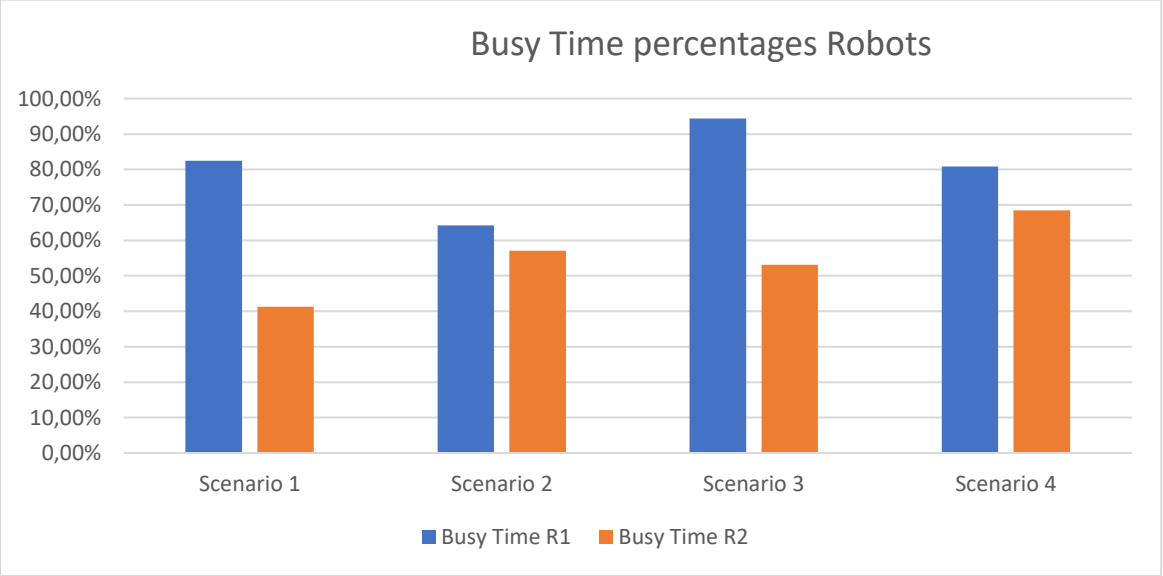

Figure 25 - Busy Time percentages of the robots

A major difference between the performance of scenario one and two is that the utilization of the robots is distributed more evenly in the second scenario. In scenario 1, the first robot has a utilization 82,5%, while the second robot has a utilization of 41,3%. This means that the average service time of the first task is higher than the average service time of the packaging tasks. In scenario 2, robot 2 will help robot 1 and take over some of its work, which leads to a more equal distributed work load. Another difference between the first and second scenario is that the throughput time is slightly lower in the second scenario. This is also a result of robot 2 helping robot 1, since queues at the first task are solved quicker when two resources are able to work on the tasks in that queue. The last difference is that

the costs of especially the transport task are significantly lower in scenario 2. The reason for this is that the smaller robot will do certain tasks faster and therefore cheaper. On the other hand, the costs of the packaging tasks are higher. This is caused by the adaptability time of robot 2, since this robot will continuously switch robot status and therefore adds the adaptability time and thus the cost to this task.

Next, the performance of the first and the third scenario will be compared. Breakdown of robots is included in the third scenario. Both robots will spent the same percentage of time on performing the tasks in both scenarios and therefore the utilization of the robots is almost equal. However, the performance of the third scenario shows that 12% of the time, the robots are broken. This means that the percentages of time that the robots are idle, are 5,6% and 47% respectively. The throughput time rises from 123,35s to 556,04s, which is an increase of over 350%. This is due to the increase in waiting time at the first task. The utilization time of robot 1 is high, which causes a queue in front of the task. Furthermore, each time the robot breaks down, the queue will extend and when the robot is repaired again, the robot has a hard time to decrease the queue length, due to the relatively high utilization.

When the second and the fourth scenario are compared, is can be seen that the throughput time also increases immensely. The throughput time rises from 115,49s to 280,76s, an increase of over 140%. Furthermore shows the table that the average costs of the tasks are increased slightly. This is due to the fact the in less cases two robots are available, which means that in those occasions, the tasks are performed by that robot that is available, which might be not the most beneficial assignee. However, the algorithm cannot chose a robot that is not available.

The last comparison that is made is between the third and the fourth scenario. The results show that the throughput time will not rise as much when a dynamic resource allocation algorithm is used in contrary to a static allocation. The reason for this is again that queues will be served quicker when two resources are working on a queue. Also, when robot 1 breaks, robot 2 can still perform some of the tasks that are in the queue of task 1. Due to this, the queues will not grow as long in scenario four in comparison to scenario three and it does not take as much time to decrease the size of the queue again.

Summarizing the results, the effect of break downs on the system and the performance indicators is immense, due to the fact that the utilization of the robot will remain the same, while robots will also be broken in 12% of the time. This results in a lower percentage idle time of the robots and therefore a higher throughput time of the tasks. In the dynamic resource allocation scenarios, the utilization of the robots is more evenly distributed, which makes the idle time of the robots higher, which results in a lower throughput time. On top of that, when both robots are available, the robot that performs the task as cheapest and thus also as fastest will get the task assigned.

This section has showed that the Robot Attribute Taxonomy and the algorithms can be introduced into a dynamic resource allocation algorithm, by executing the implementation once. Furthermore, the extended resource allocation algorithm is implemented into a simulation, where four different scenarios where tried. For each of the scenarios, some performance indicators are generated and those are compared to each other.

6. Conclusion and Discussion

This thesis enables dynamic resource allocation for industrial robots by introducing a theoretical based robot attribute taxonomy, which can be introduced in dynamic resource allocation algorithms. Firstly, the robot attribute taxonomy is introduced and the methodology towards this final robot attribute taxonomy is discussed. Thereafter, the taxonomy is validated by using experts that had to describe a robot or a task using this taxonomy and some forms. Secondly, this taxonomy is used to create an algorithm that can decide if a robot can perform a task. Utilizing this algorithm, another algorithm is created which checks which robot is the best assignee for a certain task, using the attributes of the taxonomy to compute the required time to perform the task for each robot. Lastly, these algorithms and the taxonomy are introduced in the M+ System, which is the dynamic resource allocation algorithm that has been chosen. The adaption of the M+ System is discussed in detail, where after the extended dynamic resource allocation algorithm is used in the hypothetical process, to check if the extension works. Four different scenarios are simulated, of which two have a static allocation of robots, while in the other two scenarios the dynamic resource allocation algorithm is used. Performance Indicators of each of the scenarios are retrieved and compared.

Different from most contributions regarding dynamic resource allocation algorithms, this thesis tries to enable dynamic resource allocation by creating a taxonomy which can be used to decide which robots can perform which tasks, while other contributions in this research field just assume which robots can perform which tasks. The created robot attribute taxonomy is the first attempt to create such a general taxonomy to describe industrial robots for this purpose. The feedback that the experts gave during the validation of this taxonomy, which is discussed in section 3.9, shows that the attributes and the corresponding descriptions received from the literature are clear. Despite the experts think that the attributes are clear, they still have a hard time filling in all the attributes, due to the fact that they use the current technical specifications sheet of their robot to fill in the attributes and this sheet does not contain information about all the asked robot attributes. On top of that, some of the attributes could be filled in by one of the experts, while the other experts could not fill in the attribute. This indicates again that there is no universal way of describing industrial robots in regard to attributes, as has been concluded from the literature study. This also shows that there is a gap between theory and practice, since the practical experts are not recognizing some important attributes received from the theory.

Despite that not all objectives of the taxonomy are achieved totally, the taxonomy is still used to create a method for deciding which robot would be the best assignee for the certain task. Next, this method is implemented into the M+ System. This extended dynamic resource allocation algorithm is compared to the normal M+ System using a simulation, to show if the method and the taxonomy can improve a dynamic allocation algorithm. The simulation results of the extended dynamic resource allocation algorithm shows that dynamic resource allocation can improve performance of processes in terms of throughput time and cost, especially when the requirements of tasks are highly variable or when break downs of industrial robots have a high influence on the process. The lower throughput time was also expected, since in the adapted M+ System utilization of the robots is distributed more evenly due to the fact that the robots take over some of the tasks of the other robot. The robot that has the highest utilization in the scenario with the standard M+ System, will have a lower utilization in the scenario with the adapted M+ System. In general, a lower utilization of a resource means an decrease in waiting time and this is also exactly what happened in this simulation. On top of that, when a new task arrived and both robots are available, the robot that has the lowest execution time for that task will get assigned that task in the scenarios with the adapted M+ System, while in the other scenarios the robot that is assigned to that specific task will get assigned the task.

In the simulation, the allocation of the tasks to robots is done automatically, which is also needed to enable the performance improvements. When humans should interfere in the allocation process, the allocation will be slower and would therefore be less beneficial.

An important limitation of this research is that the taxonomy is completely created from a literature review. Despite the taxonomy is revised by an expert, the expert only looked at the definitions of the possible attributes. During the revision, the expert did not give any advice of adding other attributes. This means that practical experts did not bring in any of the attributes. On top of that, reducing the attributes is done by the writer of this thesis and is only verified by the expert. This might cause the gap between the knowledge of the practical experts and the attributes within the robot attribute taxonomy.

Another limitation of this research is that only one use-case is used, which is also pretty small and not totally representive for practical manufacturing processes. Performance can be improved in this hypothetical process due to the taxonomy, but this does not mean that performance is improved for other processes. Ideally, multiple reallife processes were simulated to check whether the taxonomy really improves process performance.

Suggestions for future research are twofold. First option is to improve the current taxonomy, by revising the taxonomy again using multiple experts. By using multiple experts, the gap between the taxonomy and the practical environments can be closed. The group of experts can add attributes to the taxonomy, group the attributes differently and can seek for better connection to the practical side of this field. This can push forward the generalization of the taxonomy. If the gap between the taxonomy and practice is closed, the taxonomy will be adopted more frequently. On top of that, the taxonomy should be tested in different reallife manufacturing processes to check if it is still improving the performance of the processes.

Second, a future research can look at how to extend this research. The robot attribute taxonomy can be combined with the method for dynamic resource allocation for human employees(Erasmus et al., 2018). Combining those methods into one method, dynamic resource allocation can be enabled for systems with both types of resources. Additionally, future research can also focus on dynamic resource allocation of teams of resources. In here, a method should be shaped, which uses both the robot attribute taxonomy and the human employees method to rank all possible teams of resources for a certain task. The difficulties in both extensions is that the robot attribute taxonomy should be compared to the method for humans. The challenge will lay in how to decide which of the assignees is the best, since not all attributes in the taxonomy for industrial robots will be comparable to the attributes in the taxonomy for humans. Where all attributes within the robot attribute taxonomy are focused on manufacturing environments due to the matching procedure, possesses the method for describing human employees still attributes which are not focused on manufacturing. Additionally, the method for describing human employees uses a 7-point scale to fill in all attributes and requirements, while the attributes and requirements of the robot attribute taxonomy can be strings, integers and real numbers. Therefore, both taxonomies should be synchronized in terms of which attributes and the scales of those attributes.

References

Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, *54*(15), 2787–2805. https://doi.org/10.1016/j.comnet.2010.05.010

Beer, J. M., Fisk, A. D., & Rogers, W. A. (2014). Toward a Framework for Levels of Robot Autonomy in Human-Robot Interaction. *Journal of Human-Robot Interaction*, *3*(2), 27. https://doi.org/10.5898/JHRI.3.2.Beer

Bhangale, P. P., Agrawal, V. P., & Saha, S. K. (2004). Attribute based specification, comparison and selection of a robot. *Mechanism and Machine Theory*, *39*(12), 1345–1366. https://doi.org/10.1016/j.mechmachtheory.2004.05.020

Botelho, S. C., & Alami, R. (1999). M+: a scheme for multi-robot cooperation through negotiated task allocation and achievement. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)* (Vol. 2, pp. 1234–1239). Detroit, MI, USA: IEEE. https://doi.org/10.1109/ROBOT.1999.772530

Ceroni, J. A., & Nof, S. Y. (1999). Robotics Terminology. In S. Y. Nof (Ed.), *Handbook of Industrial Robotics* (pp. 1259–1317). Hoboken, NJ, USA: John Wiley & Sons, Inc. https://doi.org/10.1002/9780470172506.gloss

Chien, S., Barrett, A., Estlin, T., & Rabideau, G. (2000). A comparison of coordinated planning methods for cooperating rovers. In *Proceedings of the fourth international conference on Autonomous agents - AGENTS '00* (pp. 100–101). Barcelona, Spain: ACM Press. https://doi.org/10.1145/336595.337057

Das, G. P., McGinnity, T. M., Coleman, S. A., & Behera, L. (2015). A Distributed Task Allocation Algorithm for a Multi-Robot System in Healthcare Facilities. *Journal of Intelligent & Robotic Systems*, *80*(1), 33–58. https://doi.org/10.1007/s10846-014-0154-2

Datta, S., Sahu, N., & Mahapatra, S. (2013). Robot selection based on grey-MULTIMOORA approach. *Grey Systems: Theory and Application*, *3*(2), 201–232. https://doi.org/10.1108/GS-05-2013-0008

Dias, M. B. (2004). *TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments*.

Dumas, M., La Rosa, M., Mendling, J., & Reijers, H. A. (Eds.). (2013). *Fundamentals of business process management*. Berlin: Springer.

Erasmus, J., Vanderfeesten, I., Traganos, K., & Grefen, P. (2019). On the suitability of BPMN for manufacturing operations. *Computers in Industry*, (In press).

Erasmus, J., Vanderfeesten, I., Traganos, K., Jie-A-Looi, X., Kleingeld, A., & Grefen, P. (2018). A Method to Enable Ability-Based Human Resource Allocation in Business Process Management Systems. In R. A. Buchmann, D. Karagiannis, & M. Kirikova (Eds.), *The Practice of Enterprise Modeling* (Vol. 335, pp. 37–52). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-02302-7_3

Fleishman, E. A. (1975). Toward a taxonomy of human performance. *American Psychologist*, *30*(12), 1127–1149. https://doi.org/10.1037/0003-066X.30.12.1127

Gerkey, B.P., & Mataric, M. J. (2002). Sold!: auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, *18*(5), 758–768. https://doi.org/10.1109/TRA.2002.803462

Gerkey, Brian P., & Matarić, M. J. (2004). A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research*, *23*(9), 939–954. https://doi.org/10.1177/0278364904045564

German Institute for Standardisation. (2003). DIN 8580 -Manufacturing processes - Terms and definitions, division.

Groover, M. P. (2017). *Groover's principles of modern manufacturing: materials, processes, and systems* (6th ed.). Hoboken, New Jersey: Wiley & Sons.

Hofmann, E., & Rüsch, M. (2017). Industry 4.0 and the current status as well as future prospects on logistics. *Computers in Industry*, *89*, 23–34. https://doi.org/10.1016/j.compind.2017.04.002

Horse. (2015). HORSE. Retrieved June 7, 2018, from http://www.horse-project.eu/The-project

IEC. (2013). *Enterprise-control system integration - Part 1: Models and terminology* (2nd ed.). Geneva, Switzerland: The International Electrotechnical Commission (IEC).

IFR. (2018). Executive Summary. In *World Robotics* (2018th ed.). IFR.

Industrie 4.0 Working Group. (2013). *Recommendations for implementing the strategic initiative INDUSTRIE 4.0.pdf* (No. Final Report) (p. 82).

International Council on Systems Engineering (INCOSE). (2015). *Systems engineering handbook: a guide for system life cycle processes and activities*. (D. D. Walden, G. J. Roedler, K. Forsberg, R. D. Hamelin, & T. M. Shortell, Eds.) (Fourth edition). Hoboken, New Jersey: Wiley.

International Electrotechnical Commission. (2001). *Degrees of protection provided by enclosures (IP code).* Geneva, Switzerland: International Electrotechnical Commission.

IoT: number of connected devices worldwide 2012-2025. (2016, November). Retrieved November 8, 2018, from https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

Korsah, G. A., Stentz, A., & Dias, M. B. (2013). A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, *32*(12), 1495–1512. https://doi.org/10.1177/0278364913496484

Lee, J. (2015). Smart Factory Systems. *Informatik-Spektrum*, *38*(3), 230–235. https://doi.org/10.1007/s00287-015-0891-z

Lemaire, T., Alami, R., & Lacroix, S. (2004). A distributed tasks allocation scheme in multi-UAV context. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004* (pp. 3622-3627 Vol.4). New Orleans, LA, USA: IEEE. https://doi.org/10.1109/ROBOT.2004.1308816

Lerman, K., Jones, C., Galstyan, A., & Matarić, M. J. (2006). Analysis of Dynamic Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research*, *25*(3), 225–241. https://doi.org/10.1177/0278364906063426

MacKenzie, D. C. (2013). Collaborative Tasking Of Tightly Constrained Multi-Robot Missions, 13.

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*(4), 251–266. https://doi.org/10.1016/0167-9236(94)00041-2

Mishra, R., Pundir, A. K., & Ganapathy, L. (2014). Manufacturing Flexibility Research: A Review of Literature and Agenda for Future Research. *Global Journal of Flexible Systems Management*, *15*(2), 101–112. https://doi.org/10.1007/s40171-013-0057-2

Moody, D. L. (2003). The Method Evaluation Model: A Theoretical Model for Validating Information Systems Design Methods, 18.

Mosteo, A. R., Montano, L., & Lagoudakis, M. G. (2008). Multi-robot routing under limited communication range. In *2008 IEEE International Conference on Robotics and Automation* (pp. 1531–1536). Pasadena, CA, USA: IEEE. https://doi.org/10.1109/ROBOT.2008.4543419

Nof, S. Y., & Fisher, E. L. (1982). Analysis of robot work characteristics. *Industrial Robot: An International Journal*, *9*(3), 166–171. https://doi.org/10.1108/eb004525

Oxford Dictionaries. (n.d.). cost | Definition of cost in English by Oxford Dictionaries. Retrieved January 28, 2019, from https://en.oxforddictionaries.com/definition/cost

Parameshwaran, R., Praveen Kumar, S., & Saravanakumar, K. (2015). An integrated fuzzy MCDM based approach for robot selection considering objective and subjective criteria. *Applied Soft Computing*, *26*, 31–41. https://doi.org/10.1016/j.asoc.2014.09.025

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77. https://doi.org/10.2753/MIS0742-1222240302

Prat, N., Comyn-Wattiau, I., & Akoka, J. (2014). Artifact Evaluation in Information Systems Design-Science Research - A Holistic View, 17.

Reddy, P. V. P., Ali, K. M. I., Sandeep, B., & Ravi, T. (2012). Importance and Benefits of IPV6 over IPV4: A Study, *2*(12), 2.

Rus, D., Butler, Z., Kotay, K., & Vona, M. (2002). Self-reconfiguring robots. *COMMUNICATIONS OF THE ACM*, *45*(3), 39–45.

Sonnenberg, C., & vom Brocke, J. (2012). Evaluation Patterns for Design Science Research Artefacts. In M. Helfert & B. Donnellan (Eds.), *Practical Aspects of Design Science* (Vol. 286, pp. 71–83). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-33681-2_7

spreekwoorden.nl. (n.d.). the definition of Stilstand is achteruitgang. Retrieved November 9, 2018, from https://spreekwoorden.nl/spreekwoord/stilstand-is-achteruitgang

Stratista. (2017, October). Shipments of industrial robots by region 2020| Forecast. Retrieved June 7, 2018, from https://www.statista.com/statistics/272179/shipments-of-industrial-robots-by-world-region/

Swift, K. G., & Booker, J. D. (2013). *Manufacturing process selection handbook*. Amsterdam ; Boston: Elsevier, Butterworth-Heinemann.

Tansel İç, Y., Yurdakul, M., & Dengiz, B. (2013). Development of a decision support system for robot selection. *Robotics and Computer-Integrated Manufacturing*, *29*(4), 142–157. https://doi.org/10.1016/j.rcim.2012.11.008

Veld, J. in ’t., Veld, M. in ’t., & Slatius, B. (2007). *Analyse van bedrijfsprocessen : een toepassing van denken in systemen.* (9e dr.). Groningen : Wolters-Noordhoff,.

Wang, S., Wan, J., Li, D., & Zhang, C. (2016). Implementing Smart Factory of Industrie 4.0: An Outlook. *International Journal of Distributed Sensor Networks*, *12*(1), 3159805. https://doi.org/10.1155/2016/3159805

Appendix A – Floor map of the hypothetical process

In figure 23, the movement of the resources during the tasks of the hypothetical process are shown. Arrow 1 and 2 represent the getting the materials from the storage rack, while arrow 3 and 4 represent the packaging tasks.

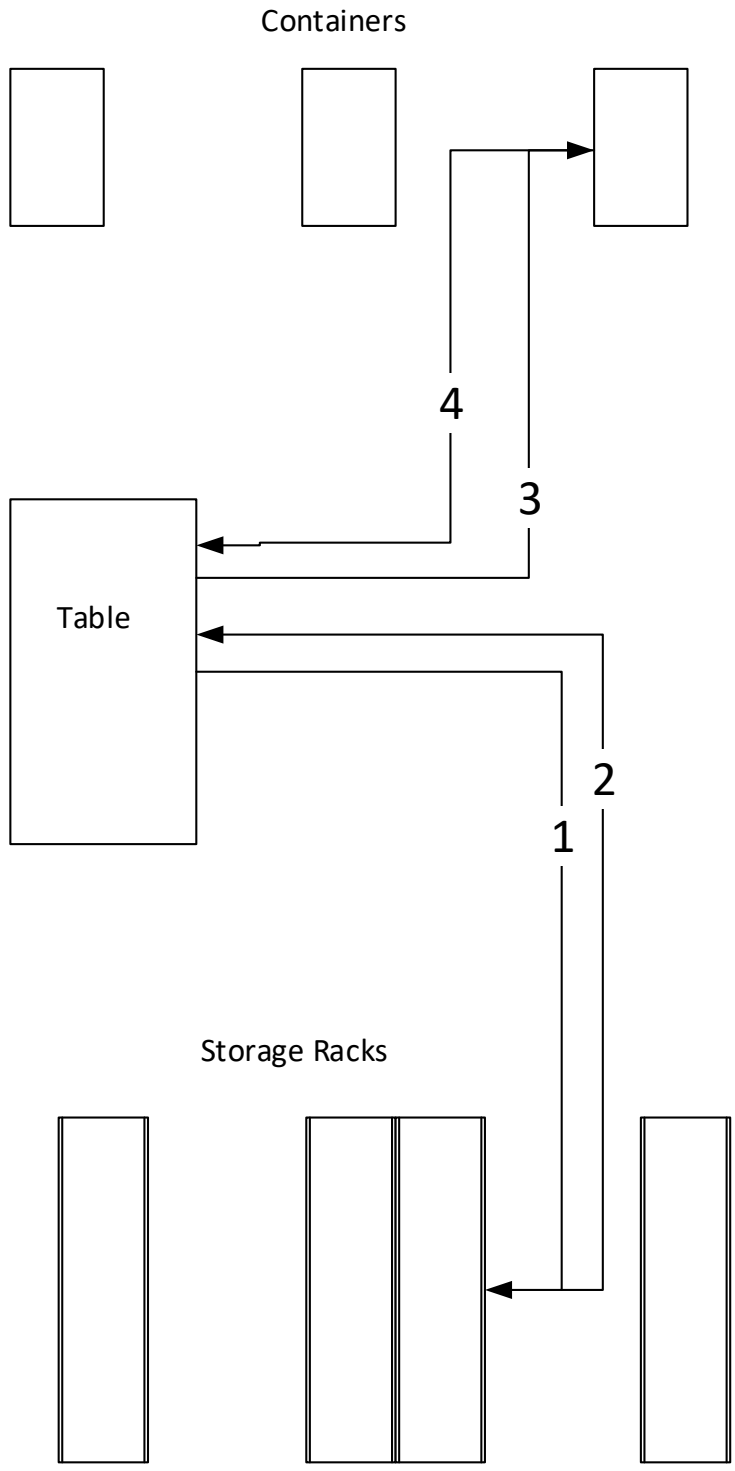

Figure 26 - Drawing of movements of the resource during the tasks

Appendix B – Taxonomy created from the Literature Review

Table 29 - Cost Attributes in Taxonomy out of LR

Cost-attributes	Definition
Maintenance Costs	Cost for maintaining the robot
Operation Costs	Cost for running the robot
Depreciation Costs	Cost of depreciation
Environmental Impact	Similar to something like energy consumption
Power Consumption	How much power does the robot use
Cost of energy	How much energy is used by the robot
Insurance Cost	Cost for the insurance
Depreciation of robot	Depreciation of robot
Noise Delivery	Noise Output

Table 30 - Availability Attributes in Taxonomy out of LR

Availability-attributes	Definition
Reliability	Such as reliability can be expressed by Mean Time Between Failure (MTBF) or Mean Time To Repair (MTTR)
Downtime	How long/often is the robot unavailable
warranty	Warranty Period
Service	Service provided by producer
Rest Time	Downtime

Table 31 - Physical Attributes in Taxonomy out of LR

Physical-attributes	Definition
Weight Of the Robot	Weight of the Robot
Size of the robot	Size of the robot
Drive Systems	Electric/Hydraulic
Processor	Processor
Technical Features	General term for some technical features
AC Current Supply	Power
AC Frequency	Power
AC Phase	Power
AC power consumption	Power
AC Voltage Requirement	Power
base size	Size of the base of the robot
Electrical Drive System	Drive System
Hydraulic drive System	Drive System
Material Of robot	Material of the robot
Material used for links	Material of the robot
Power	Power needed
RAM	RAM
ROM	ROM

Table 32 - Sophistication of Equipment Attributes in Taxonomy out of LR

Sophistication of Equipment - attributes	Definition
Maintainability/regular maintenance	Easy to maintain
Safety	Safety of the robot for the user
Assemblability	How easy to assemble
Degree of protection	Degree of protection denotes the protection provided by an enclosure against access to hazardous parts, ingress of solid foreign objects and water.
Dissemblability	How easy to dissemble
Maintainability and Safety Features	Maintainability
protection class	A term stating the durability which is supported by enclosure of electrical equipment towards environmental conditions
Convenience of use	How easy to use?
Adaptability	Adaptability of the robot
Convertibility (design for functionality changes)	How easy to change robot for different functionalities
Simplicity	How easy is the robot
Implementation easiness	How easy to implement the robot

Table 33 - Performance Attributes in Taxonomy out of LR

Performance-attributes	Definition
Load Capacity	Load capacity is the maximum load that a manipulator can carry without affecting its performance.
Repeatability	Repeatability is the measure of the ability of a robot to return to the same position and orientation over and over again.
velocity	Is used as Maximum Tip Speed in 1
Maximum Tip Speed	Maximum tip speed is the speed at which a robot can move in an inertial reference frame.
Accuracy	accuracy is the measure of closeness between the robot end effectors and the target point, and can usually be defined as the distance between the target point and the center of all points to which the robot goes on repeated trials.
Manipulator Reach	Manipulator reach is the maximum distance that can be covered by the robotic manipulator so as to grasp objects for the given pick-n-place operation.
Positioning Accuracy	Probably same as accuracy
Compliance	This is the measure of displacement of the end effector in response to force (or torque) applied to it.
Stability	This is the measure of absence of oscillations at termination of arm movement.
Repeatability Error	Same as Repeatability?
Dexterity	The dexterity of industrial robots, introduced here as C6, has many definitions in the literature. Mainly, it can be considered as the ability to change position and orientation of the end effector, between two different configurations of the robotic structure
Geometrical Dexterity	Same as dexterity?
Productivity	Output of the robot
Force Output	Output of Force
Maximum end Effectors speed	Same as Maximum Tip Speed
Maximum joint speed	joint speed is defined as how quickly a robot can position its arm/actuator.
maximum speed end effector	Maximum tip speed
Operation Time	Duration operation time
Overload capacity of the robot	Maximum capacity
Position Repeatability	position repeatability is articulated as how well a robot can come back to a programmed location and orientation over and over again
Runtime	Runtime
Speed	Velocity
Efficiency	Efficiency of the robot
Life expectancy	Expected Life range

Table 34 - Architecture/Structure Attributes in Taxonomy out of LR

Architecture/Structure attributes -	Definition
Degree of freedom	A term used to describe a robot's freedom of motion in three-dimensional space, specifically the ability to move forward and backward, up and down, and to the left and to the right
Vertical reach	vertical reach of the robot
Number of Axes	Sort of Same as degree of freedom
Reach	Reach in vertical, horizontal…
Type of Actuators	Type of actuators used
type of grippers supported	Which grippers are supported at the robot
Type of Joints	Which Type of Joints used
B axis Max Speed and Range	Speed and Range of certain Axis
Gripper parameters(Fantoni)	Gripper Type
Horizontal Reach	Reach of the robot of the horizontal axis.
Internal state sensors	Sensors
Joint Orientations	Joints
Joint Sequence	Joints
L axis Max Speed and Range	Speed and Range of certain Axis
Means used for rotary to linear motion conversion	Joints
Means used for rotary to rotary motion conversion	Joints
number of end effectors	Number of tips
Number of joints	Joints
Number of offset joints	Joints
R axis Max Speed and Range	Speed and Range of certain Axis
Range Of Joint Motions	Joint
S axis Max Speed and Range	Speed and Range of certain Axis
T axis Max Speed and Range	Speed and Range of certain Axis
Type of Cables used	Types Of Cables
Type of Control System	Control System
Type of end effectors	Type of end effectors used
Type of flexible drive system used	Drive system used
type of memory	Type of memory used
U axis Max Speed and Range	Speed and Range of certain Axis
Working range of joints	Range
wrist reach distance	Maximum distance in which the end effector of the robot arm reaches in the work envelop

Table 35 - Application Attributes in Taxonomy out of LR

Application-attributes	Definition
Ambient Temperature	Work area temperature
Space Requirements	How much space does the robot need.
connection type	The place (base, ceiling, wall) where the robot arm is settled. The facility layout, constraints, and obstacles are taken into account to choose the connection type
footprint	Footprint is the amount of space required for installing a robot.
Mounting Position	Mounting position represents the ability of a robot to fasten it securely on a given surface during the working cycle,
Power Source Requirement	Power/Energy
Recommended Operating Humidity	Recommended humanity around the robot
Terrain Traversability	Terrain Traversability
Type of Robot	Robot Type
Usage of Robot	The usage of the robot
Workspace	Workspace needed

Table 36 - Control and Feedback Systems Attributes in Taxonomy out of LR

Control and Feedback systems - attributes	Definition
Programming Flexibility	while programming flexibility refers to a robot's ability to accept different programming codes.
Memory Capacity	Memory capacity of a robot is measured in terms of number of points or steps that it can store in its memory while traversing along a predefined path.
Programming Method	Programming Method
Gripper Control	Gripper
Number Of in and output channels of the controller	Controller
External state sensors	Sensors
Force and Torque sensors	Sensors
Input channels	Controller
Output channels	Controller
Robotic Arc Welding sensors	Sensors
Sensors	Sensors
Slip Sensors	Sensors
Tactile Sensors	Sensors

Table 37 - Miscallaneous Attributes in Taxonomy out of LR

Miscellaneous-attributes	Definition
Handling Coefficient	The value of handling coefficient can be determined from various features, like diameter (in mm), elevation (in mm), basic rotation (in degree), roll (in degree), pitch (in degree) and yaw (in degree). The diameter, elevation and basic rotation are related to the work area to a robot arm, whereas, roll, pitch and yaw are related to rotational angles of the robot wrist about the three principal axes.
Versions of robot Installations	Version of software
Working Automation	How automated is the robot

Table 38 - Undefined Attributes in Taxonomy out of LR

Attribute	Definition
Man-machine interface	
Control of Robotic Joints	
Path Measuring Systems	
Resolution	
Stroke	
Supporting Channel Partner's performance	
Velocity Ratio	
Working Environment	
Autonomy	
Cable layout	
Classification by control method	
Classification by type of mechanism	
Communication Bandwidth	
Communication Range	
Computational Capacity	
Computational Complexity of dynamic equations	
Configuration of robot	
Coordinate System	
dexterous workspace	
DH parameters	
Environmental Performance	
Inconsistency with infrastructure	
Link cross sections	
Link length Ratios	
Link Masses and inertia properties	
Manipulability measure	
Manipulator	
Motion	
Motion transformation from actuator to link	
Mounting Arrangement	
Natural Frequency	
Reaction Speed	

Robot arm configuration	
Sensing Distribution	
Sensing Quantity	
Sensing Range	
Sensitivity	
Simulation Software	
Singularities present	
Type of basic robot configurations	
Type of gear train used	
Type of gears used for power transmission	
Scheduling Utilization	
Risk	
Overall Flexibility	
Quality Aspect	
Overall Performance	
User Support	
Program steps	
Travelling Time	
Functionality	
Favorable Appearance and Proper Structure	
Precision	
Learning	
Exposure to operator Unrest	
Part Delivery Position Accuracy	
Control Technologies	
Defect Prevention handling Parts	
Vibration aspect	
Integrated with a PC	
Spline Interpolation	
6 DOF F/T (Force Torque Measurement)	
Force-position (compliance) Control	
online and offline programming	
Teach-in play-back programming technology	
Path Correction Facilities	
Online Program correction	

Appendix C – Excluded Attributes, grouped by excluding rules

The following 11 attributes are excluded from the list since they are too abstract.

Table 39 - Attributes Excluded because they are too abstract

Attribute	Description	Reason for excluding
Functionality		Abstract
Motion		Abstract
Online and offline programming		Abstract
Overall Performance		Abstract
Programming Flexibility	Programming flexibility refers to a robot's ability to accept different programming codes.	Abstract
Quality Aspect		Abstract
Risk		Abstract
Safety	Safety of the robot for the user.	Abstract
Sensors	Sensors	Abstract
Technical Features	General term for some technical features.	Abstract
Usage of Robot	The usage of the robot.	Abstract

The following 58 attributes are excluded since they are a subset of another attribute in the list.

Table 40 - Attributes Excluded because they are a subset of another attribute

Attribute	Description	Reason for excluding
AC Current Supply	Power	Subset of Power Consumption
AC Frequency	Power	Subset of Power Consumption
AC Phase	Power	Subset of Power Consumption
AC power consumption	Power	Subset of Power Consumption
AC Voltage Requirement	Power	Subset of Power Consumption
Assemblability		Subset of Adaptability
B axis Max Speed and Range	Speed and Range of certain Axis	Subset of Speed and Reach, respectively.
Base size	Size of the base of the robot	Subset of Size of the Robot
Computational Complexity of dynamic equations		Subset of Computational Capacity
Control of Robotic Joints		Subset of Accuracy

Attribute	Description	Reason for excluding
Cost of energy	How much energy is used by the robot	Subset of Power Consumption
Cost of Feeder	Cost of the feeder	Subset of Purchase Cost
Cost of Gripper Mechanics	Cost of the mechanics of the gripper	Subset of Purchase Cost
Cost of sensors	Cost of sensors in the robot	Subset of Purchase Cost
Depreciation Costs	Cost of depreciation	Subset of Operation Costs
Dexterous workspace		Subset of Dexterity
Dissemblability	How easy to dissemble	Subset of Adaptability
Downtime	How long/often is the robot unavailable	Subset of Reliability
Electrical Drive System	Drive System	Subset of Drive Systems
Footprint	Footprint is the amount of space required for installing a robot.	Subset of Space Requirements
Force and Torque sensors	Sensors	Subset of Tactile Sensors
Geometrical Dexterity	Same as dexterity?	Subset of Dexterity
Hydraulic drive System	Drive System	Subset of Drive Systems
Input channels	Controller	Subset of Number of in and output channels of the controller
Insurance Cost	Cost for the insurance	Subset of Operation Costs
L axis Max Speed and Range	Speed and Range of certain Axis	Subset of Speed and Reach, respectively.
Link cross sections		Subset of Degrees of Freedom
Link length Ratios		Subset of Reach
Material used for links	Material of the robot	Subset of Material of Robot
Maximum joint speed	joint speed is defined as how quickly a robot can position its arm/actuator.	Subset of Maximum Tip Speed
Number of Axes	Sort of Same as degree of freedom	Subset of Degrees of Freedom
Number of joints	Joints	Subset of Degrees of Freedom

Attribute	Description	Reason for excluding
Number of offset joints	Joints	Subset of Number of Joints
Output channels	Controller	Subset of Number of in and output channels of the controller
Overall Flexibility		Subset of Degrees of Freedom
Part Delivery Position Accuracy		Subset of Accuracy
Path Correction Facilities		Subset of Accuracy
Position Repeatability	position repeatability is articulated as how well a robot can come back to a programmed location and orientation over and over again.	Subset of Repeatability
Positioning Accuracy	Probably same as accuracy	Subset of Accuracy
Power	Power needed	Subset of Power Consumption
Programming Method	Programming Method	Subset of Programming Flexibility
Purchase Cost	Cost for buying the robot	Subset of Operation Costs
R axis Max Speed and Range	Speed and Range of certain Axis	Subset of Speed and Reach, respectively.
Reaction Speed		Subset of Velocity
Repeatability Error	Same as Repeatability?	Subset of Repeatability
Robotic Arc Welding sensors	Sensors	Subset of External State Sensors
S axis Max Speed and Range	Speed and Range of certain Axis	Subset of Speed and Reach, respectively.
Size of the robot	Size of the robot	Subset of Space Requirements
Speed	Velocity	Subset of Maximum Tip Speed
T axis Max Speed and Range	Speed and Range of certain Axis	Subset of Speed and Reach, respectively.
Total Cost of Layout	Cost	Subset of Purchase Cost
Travelling Time		Subset of Velocity

Attribute	Description	Reason for excluding
Type of grippers supported	Which grippers are supported at the robot	Subset of type of End Effectors
U axis Max Speed and Range	Speed and Range of certain Axis	Subset of Speed and Reach, respectively.
Velocity	Is used as Maximum Tip Speed	Subset of Maximum Tip Speed
Velocity Ratio		Subset of Velocity
Vibration aspect		Subset of Accuracy
Working range of joints	Combination of Degrees of Freedom and Reach	Subset of Degrees of Freedom

The following 15 attributes are excluded since they are a duplicate of another attribute in the list.

Table 41 - Attributes excluded because they are a duplicate of another attribute

Attribute	Description	Reason for excluding
6 DOF F/T (Force Torque Measurement)		Duplication of Degrees of Freedom
Connection type	The place (base, ceiling, wall) where the robot is settled. The facility layout, constraints, and obstacles are taken into account to choose the connection type	Duplicate of Mounting Position
Depreciation of robot	Depreciation of robot	Duplicate of Depreciation Costs
Force-position (compliance) Control		Duplication of Compliance
Gripper parameters(Fantoni)	Gripper Type	Duplicate of Type of Grippers Supported
Maximum end Effectors speed	Same as Maximum Tip Speed	Duplication of Maximum Tip Speed
Maximum speed end effector	Maximum tip speed	Duplication of Maximum Tip Speed
Mounting Arrangement		Duplicate of Mounting Position
Protection Class	A term stating the durability which is supported by enclosure of electrical equipment towards environmental conditions	Duplicate of Degrees of Protection
Range Of Joint Motions	Joint	Duplicate of Working Range of Joints
Reach	Reach in vertical, horizontal…	Duplicate of vertical and horizontal reach.
Runtime	Runtime	Duplication of Operation Time
Type of Actuators	Type of actuators used	Duplicate of Type of End Effectors
Working Automation	How automated is the robot.	Duplicate of Autonomy
Workspace	Workspace needed	Duplicate of Space Requirements

The following nine attributes are excluded since they are the superset of different other attributes:

Table 42 - Attributes Excluded because they are a superset of other attributes

Attribute	Description	Reason for excluding
Efficiency	Ratio of output to input.	Superset of output and costs attributes.
External state sensors	The ability of the robot to detect externally perceivable effects that it has.	Combination of Force, Torque, Slip and Tactile Sensors
Handling Coefficient	The value of handling coefficient can be determined from various features, like diameter (in mm), elevation (in mm), basic rotation (in degree), roll (in degree), pitch (in degree) and yaw (in degree). The diameter, elevation and basic rotation are related to the work area to a robot arm, whereas, roll, pitch and yaw are related to rotational angles of the robot wrist about the three principal axes.	Combination of Degrees of Freedom and Space Required
Maintainability and Safety Features	Maintainability	Combination of Maintainability and Safety
Manipulator	Mechanism, usually consisting of a series of segments, or links, jointed or sliding relative to one another, for grasping and moving objects, usually in several degrees of freedom. It is remotely controlled by a human (manual manipulator) or a computer (programmable manipulator). The term refers mainly to the mechanical aspect of a robot.	Combination of many attributes.
Manipulator Reach	Manipulator reach is the maximum distance that can be covered by the robotic manipulator so as to grasp objects for the given pick-n-place operation.	Combination of Horizontal and Vertical Reach
Precision	A general concept reflecting the robot's accuracy, repeatability, and resolution.	Combination of accuracy, repeatability and resolution.
Productivity	Number of actions that can be performed in a time unit.	Combination of maximum tip speed, traversability.
Working Environment		Combination of Ambient Temperature and other environmental factors

The following 13 attributes are excluded since they are a component instead of an output description:

Table 43 - Attributes excluded because they are describing a component

Attribute	Description	Reason for excluding
Cable layout		Component description, instead of output description.
Configuration of robot		Component description, instead of output description.
Control Technologies		Component description, instead of output description.
Favorable Appearance and Proper Structure		Component description, instead of output description.
Integrated with a PC		Component description, instead of output description.
Link Masses and inertia properties		Component description, instead of output description.
Online Program correction		Component description, instead of output description.
Program steps		Component description, instead of output description.
Robot arm configuration		Component description, instead of output description.
Simulation Software		Component description, instead of output description.
Teach-in play-back programming technology		Component description, instead of output description.
Type of basic robot configurations		Component description, instead of output description.
Type of gears used for power transmission		Component description, instead of output description.

The following 21 attributes could not be excluded based on another criteria, but could also not be matched with a task since the attributes are not defined:

Table 44 - Attributes excluded because they are not defined

Attribute	Description	Reason for excluding
Classification by control method		No definition
Classification by type of mechanism		No definition
Coordinate System		No definition
Defect Prevention handling Parts		No definition
DH parameters		No definition
Environmental Performance		No definition
Exposure to operator Unrest		No definition
Gripper Control		No definition
Inconsistency with infrastructure		No definition
Manipulability measure		No definition
Motion transformation from actuator to link		No definition
Natural Frequency		No definition
Scheduling Utilization		No definition
Sensing Distribution		No definition
Sensing Quantity		No definition
Sensitivity		No definition
Singularities present		No definition
Spline Interpolation		No definition
Stroke		No definition
Supporting Channel Partner's performance		No definition
Type of gear train used		No definition
User Support		No definition

Appendix D – Attributes excluded using the Matching Procedure

Table 45 - Attributes and corresponding definitions excluded by the matching procedure

Attribute	Description
Communication Bandwidth	Range of bandwidth on which the robot can communicate
Communication Range	Range of the communication system of the robot
Computational Capacity	
Convenience of use	How easy to use?
Convertibility (design for functionality changes)	How easy to change robot for different functionalities
Customization product delivery	How customized the robot can be delivered
Delivery Time	Delivery Time of the robot
Dexterity	The ability to change position and orientation of the end effector, between two different configurations of the robotic structure.
Drive Systems	Electric/Hydraulic
Implementation easiness	How easy to implement the robot
Internal state sensors	The ability of the robot to detect the current status of its internal operation.
Joint Orientations	Joints
Joint Sequence	Joints
Learning	
Life expectancy	Expected Life range
Maintainability/regular maintenance	Ease and cost of maintenance.
Man-machine interface	Same as user interface. The interface between the robot and the operator through devices such as a teach pendant or PC. It provides the operator with the means to create programs, jog the robot, teach positions, and diagnose problems.
Means used for rotary to linear motion conversion	Joints
Means used for rotary to rotary motion conversion	Joints
Memory Capacity	Memory capacity of a robot is measured in terms of number of points or steps that it can store in its memory while traversing along a predefined path.
Number of in and output channels of the controller	Controller
Path Measuring Systems	A part of the mechanical construction of each axis which provides the position coordinate for the axis. Typically, for translational axes, potentiometers or ultrasound are used for path measuring systems. But for rotational axes, resolvers, absolute optical encoders, or incremental encoders are used. A path measuring system may be located directly on a robot axis or included with the drive system.
Power Source Requirement	The type of energy consumed by the robot.

Attribute	Description
Processor	Processor
RAM	RAM
ROM	ROM
Service	Service provided by producer
Simplicity	How easy is the robot
Training Delivery Period	Delivery time of the training
Type of Cables used	Types Of Cables
Type of Control System	Control System
Type of flexible drive system used	Drive system used
Type of Joints	Which Type of Joints used
Type of memory	Type of memory used
Type of Robot	Robot Type
Vendor's Service (Contract)	Vendor’s service quality refers to the level and variety of services offered by a robot vendor
Vendor's training	Training from producer
Versions of robot Installations	Version of software
Warranty	Warranty Period
Weight Of the Robot	Weight of the Robot

Table 46 - Level of Robot Autonomy Framework

LORA	Sense	Plan	Act	Description	Examples from Literature
Manual	H	H	H	The human performs all aspects of the task including sensing the environment, generating plans/options/goals, and implementing processes.	"Manual Control" Endsley & Kaber, 1999
Tele-operation	H/R	H	H/R	The robot assists the human with action implementation. However, sensing and planning is allocated to the human. For example, a human may teleoperate a robot, but the human may choose to prompt the robot to assist with some aspects of a task (e.g., gripping objects).	"Action Support" Endsley & Kaber, 1999; Kaber et al., 2000; "Manual Teleoperation" Milgram, 1995; "Tele Mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005
Assisted Tele-operation	H/R	H	H/R	The human assists with all aspects of the task. However, the robot senses the environment and chooses to intervene with task. For example, if the user navigates the robot too close to an obstacle, the robot will automatically steer to avoid collision.	"Assisted Teleoperation" Takayama et al., 2011; "Safe Mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005
Batch Processing	H/R	H	R	Both the human and robot monitor and sense the environment. The human, however, determines the goals and plans of the task. The robot then implements the task.	"Batch Processing" Endsley & Kaber, 1999; Kaber et al., 2000
Decision Support	H/R	H/R	R	Both the human and robot sense the environment and generate a task plan. However, the human chooses the task plan and commands the robot to implement actions.	"Decision Support" Endlsey & Kaber, 1999; Kaber et al., 2000
Shared Control With Human Initiative	H/R	H/R	R	The robot autonomously senses the environment, develops plans and goals, and implements actions. However, the human monitors the robot's progress and may intervene and influence the robot with new goals and plans if the robot is having difficulty.	"Shared Mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005; "Mixed Initiative" Sellner et al., 2006; "Control Sharing" Tarn et al., 1995
Shared Control With Robot Initiative	H/R	H/R	R	The robot performs all aspects of the task (sense, plan, act). If the robot encounters difficulty, it can prompt the human for assistance in setting new goals and plans.	"System-Initiative" Sellner et al., 2006; "Fixed-Subtask Mixed-Initiative" Hearst, 1999
Executive Control	R	H/R	R	The human may give an abstract high-level goal (e.g., navigate in environment to a specified location). The robot autonomously senses environment, sets the plan, and implements action.	"Seamless Autonomy" Few et al., 2008; "Autonomous mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005
Supervisory Control	H/R	R	R	The robot performs all aspects of task, but the human continuously monitors the robot, environment, and task. The human has override capability and may set a new goal and plan. In this case, the autonomy would shift to executive control, shared control, or decision support.	"Supervisory Control" Endsley & Kaber, 1999; Kaber et al., 2000
Full Autonomy	R	R	R	The robot performs all aspects of a task autonomously without human intervention with sensing, planning, or implementing action.	"Full Automation" Endsley & Kaber, 1999

*Note: H = Human, R = Robot. Manual represents a situation where no robot is involved in performing the task; this level is included for a complete taxonomy continuum.

Appendix F – Evaluation Forms

Appendix F1 – Robot Evaluation Form

Dear participant,

Thanks in advance for participating to the validation of the created model by filling in this form. The objective of this research is to create a model to describe robots, for the purpose of allocating robots to manufacturing tasks. The model comprises of 35 attributes retrieved from scientific literature. To check if a robot can be described as complete as possible using only those attributes, we want you to try and describe a robot using this form. The robots will be described to be able to decide if a robot can perform a certain task, based on the task requirements. When you have any comments regarding missing attributes, unclear attributes or the form, please let us know by writing this down in the comments section on page 2. With those comments, the model to describe a robot can be improved.

To do:

First, fill in the table below. Next, go through all the attributes, starting from page 3 and try to fill in those attributes of this robot. Lastly, fill in the three questions on page 2. Comments regarding this form can also be filled in on page 2.

Robot Name:	
Robot Vendor:	
Filled In by:	

Please answer the following three questions when you finished filling in the attributes:

Completeness: All aspects of the robot are covered by the attributes.	Strongly Agree			Weakly Disagree	
	5	4	3	2	1
Clarity: The attributes are clearly described and it is clear what the attributes mean.	Strongly Agree			Weakly Disagree	
	5	4	3	2	1
Simplicity: The model is easy to use.	Strongly Agree			Weakly Disagree	
	5	4	3	2	1
Comments:					

Cost-Attributes

Operation Cost Total cost of running the robot.	€/s
Power Consumption How much electric energy is consumed during operation, expressed as	kWh
Maintenance Cost Cost for maintaining the robot. This excludes cost of lost opportunity.	€/year
Environmental Impact The effect of the robot operation on the environment.	CO_2/Day

Output-Attributes

Operation Time Duration of continual operation without required rest.	Min
Noise Delivery The maximum noise output of the robot during runtime.	dB
Rest Time Time required by the robot to reset between operations.	sec
Maximum Tip Speed Maximum tip speed is the speed at which a robot can move in an inertial reference frame.	Mm/s
Force Output Amount of force a robot can generate.	N
Terrain Traversability How fast does the robot travel on the certain surface. (Smooth, rough, inclined)	Terrain: Speed(m/s):
Load Capacity Load capacity is the maximum load that a manipulator can carry without affecting its performance.	kg
Overload capacity of the robot Load capacity beyond the recommended specification.	kg
Reliability Such as reliability can be expressed by Mean Time Between Failure (MTBF) or Mean Time To Repair (MTTR)	min

Structure-Attributes

Type of end Effectors List of End effectors that can be used by the robot.	
Number of end effectors Number of tips the robot has.	
Slip sensors Does the robot have slip sensors?	Y / N
Mounting Position Possibility to attach the robot to surfaces in different orientations. (for example ground, wall, ceiling and mobile platforms).	
Tactile Sensors Does the robot have tactile sensors?	Y / N
Material Of Robot What material does the outer surface of the robot consist of.	
Degree of Protection Degree of protection denotes the protection provided by an enclosure against access to hazardous parts, ingress of solid foreign objects and water. (IP-Code)	

External Requirements - Attributes

Ambient Temperature Range of temperature within which the robot functions as expected.	^{o}C - ^{o}C
Space Requirements How much space does the robot need to be able to function correctly.	Length: mm Width: mm Height: mm
Recommended Operating Humidity Range of humidity within which the robot functions as expected.	% - %

Automation - Attributes

Adaptability Average duration to change the configuration of the robot to perform different operations or at different locations.	sec
Autonomy The capability of an entity to create and control the execution of its own plans and/or strategies. For instance, in mobile robots the ability of the robot for determining the trajectory to reach a specific location or pose. See **table 1** for the definitions and the different levels of autonomy.	

Precision - Attributes

Sensing Range Distance to recognise a predefined shape, image or event.	mm
Accuracy Accuracy is the measure of closeness between the robot end effectors and the target point, and can usually be defined as the distance between the target point and the centre of all points to which the robot goes on repeated trials.	mm
Repeatability Repeatability is the measure of the ability of a robot to return to the same position and orientation over and over again.	mm
Resolution The smallest incremental motion which can be produced by the manipulator. Serves as one indication of the manipulator accuracy. Three factors determine the resolution: mechanical resolution, control resolution, and programming resolution.	mm
Stability The measure of absence of oscillations at termination of arm movement.	mm
Compliance The measure of displacement of the end effector in response to force (or torque) applied to it.	mm

Reach and Distances - Attributes

Degrees of Freedom	axes
A term used to describe a robot's freedom of motion in three-dimensional space, specifically the ability to move forward and backward, up and down, and to the left and to the right. • 1-axis robot: Linear guide system for transferring parts in a single line of motion. • 2-axis robot: Typically in an XY or YZ configuration, these are often in the form of two adjoining linear guides. • 3-axis robot: Typically in an XYZ configuration, these tend to be in the form of two adjoining linear guides and a third axis guide or cylinder. • 4-axis robot: A more conventional arm that is typically used in palletizing applications in which the face plate is always parallel with the ground. Has the ability to rotate the object it is picking. • 5-axis robot: Similar to a conventional four-axis robot but adds the ability to rotate the object it is picking. • 6-axis robot: Offers the most flexibility with six axes all the way from the base axis for full robot rotation to the sixth axis for rotating the "wrist" or faceplate. • 7-axis robot: A six-axis robot which is placed on a rail or some means to move it from one place to another in a linear direction.	
Horizontal Reach	mm
Reach of the robot of the horizontal axis.	
Vertical Reach	mm
Reach of the robot of the vertical axis.	
Wrist Reach Distance	mm
Maximum distance in which the end effector of the robot arm reaches in the work envelop	

To describe the degree of protection of a robot, a IP code can be created, and should look like IPxx. To fill in the x's, use the following tables.

1st numeral: Protection against contact and against solid foreign objects.	IP numeral
0 Non-protected	IP 0X(X)
1 Protected against solid foreign objects of ≥ 50 mm ø	IP 1X(X)
2 Protected against solid foreign objects of ≥ 12.5 mm ø	IP 2X(X)
3 Protected against solid foreign objects of ≥ 2.5 mm ø	IP 3X(X)
4 Protected against solid foreign objects of ≥ 1.0 mm ø	IP 4X(X)
5 Protected against ingress of dust (dust-protected)	IP 5X(X)
6 Protected against ingress of dust by underpressure (dust-tight)	IP 6X(X)

2nd numeral: Protection against ingress of water.	IP numeral
0 Non-protected	IP X0(X)
1 Protected against vertically falling water drops	IP X1(X)
2 Protected against vertically falling water drops when enclosure tilted up to 15°	IP X2(X)
3 Protected against spraying water	IP X3(X)
4 Protected against splashing water	IP X4(X)
5 Protected against water jets	IP X5(X)
6 Protected against powerful water jets	IP X6(X)
7 Protected against the effects of temporary immersion in water	IPX7(X)
8 Protected against the effects of continuous immersion in water	IPX8(X)

LORA	Sense	Plan	Act	Description	Examples from Literature
Manual	H	H	H	The human performs all aspects of the task including sensing the environment, generating plans/options/goals, and implementing processes.	"Manual Control" Endsley & Kaber, 1999
Tele-operation	H/R	H	H/R	The robot assists the human with action implementation. However, sensing and planning is allocated to the human. For example, a human may teleoperate a robot, but the human may choose to prompt the robot to assist with some aspects of a task (e.g., gripping objects).	"Action Support" Endsley & Kaber, 1999; Kaber et al., 2000; "Manual Teleoperation" Milgram, 1995; "Tele Mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005
Assisted Tele-operation	H/R	H	H/R	The human assists with all aspects of the task. However, the robot senses the environment and chooses to intervene with task. For example, if the user navigates the robot too close to an obstacle, the robot will automatically steer to avoid collision.	"Assisted Teleoperation" Takayama et al., 2011; "Safe Mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005
Batch Processing	H/R	H	R	Both the human and robot monitor and sense the environment. The human, however, determines the goals and plans of the task. The robot then implements the task.	"Batch Processing" Endsley & Kaber, 1999; Kaber et al., 2000
Decision Support	H/R	H/R	R	Both the human and robot sense the environment and generate a task plan. However, the human chooses the task plan and commands the robot to implement actions.	"Decision Support" Endlsey & Kaber, 1999; Kaber et al., 2000
Shared Control With Human Initiative	H/R	H/R	R	The robot autonomously senses the environment, develops plans and goals, and implements actions. However, the human monitors the robot's progress and may intervene and influence the robot with new goals and plans if the robot is having difficulty.	"Shared Mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005; "Mixed Initiative" Sellner et al., 2006; "Control Sharing" Tarn et al., 1995
Shared Control With Robot Initiative	H/R	H/R	R	The robot performs all aspects of the task (sense, plan, act). If the robot encounters difficulty, it can prompt the human for assistance in setting new goals and plans.	"System-Initiative" Sellner et al., 2006; "Fixed-Subtask Mixed-Initiative" Hearst, 1999
Executive Control	R	H/R	R	The human may give an abstract high-level goal (e.g., navigate in environment to a specified location). The robot autonomously senses environment, sets the plan, and implements action.	"Seamless Autonomy" Few et al., 2008; "Autonomous mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005
Supervisory Control	H/R	R	R	The robot performs all aspects of task, but the human continuously monitors the robot, environment, and task. The human has override capability and may set a new goal and plan. In this case, the autonomy would shift to executive control, shared control, or decision support.	"Supervisory Control" Endsley & Kaber, 1999; Kaber et al., 2000
Full Autonomy	R	R	R	The robot performs all aspects of a task autonomously without human intervention with sensing, planning, or implementing action.	"Full Automation" Endsley & Kaber, 1999

*Note: H = Human, R = Robot. Manual represents a situation where no robot is involved in performing the task; this level is included for a complete taxonomy continuum.

Appendix F2 – Task Evaluation Form

Dear Participant,

Thanks in advance for participating to the validation of the created model by filling in this form. The objective of this research is to create a model to describe robots, for the purpose of allocating robots to manufacturing tasks. To check if task requirements for a robot can be formulated as complete as possible using only the attributes of the model, we want you to try and describe the task requirements of a task using this form.

To do:

You start with filling in the information about the described task. After that, you will go through the list of attributes, starting from page 3 and fill in those and only those attributes that you think are mandatory to have for a robot to be able to execute the chosen task. When you are finished with that, please go back to page 2 where you will have to answer three questions. When you have any comments regarding missing attributes, unclear attributes or the form, please let us know by writing this down in the comment section.

Task Name:	
Company:	
Filled In by:	
Description Task:	

Please answer the following three questions when you finished filling in the attributes:

Completeness: All requirements that a robot needs to perform the task are covered by the attributes in this form.	Strongly Agree				Weakly Disagree
	5	4	3	2	1
Clarity: The attributes are clearly described and it is clear what the attributes mean.	Strongly Agree				Weakly Disagree
	5	4	3	2	1
Simplicity: The form is easy to use.	Strongly Agree				Weakly Disagree
	5	4	3	2	1
Comments:					

Fill in only those attributes, that a robot needs to have to be able to execute the task.

Output-Attributes

Noise Delivery What is the maximum noise delivery that a robot can have during run-time, without negatively affecting the rest of the process.	dB
Force Output The amount of force that the robot should be able to produce.	N
Terrain Traversability What is the minimum travel speed that the robot needs on certain surfaces.(Smooth. Rough, inclined)	Terrain: Speed(m/s):
Load Capacity What is the minimum load capacity that the robot needs. Robot capacity is the maximum load that a manipulator can carry without affecting its performance.	kg
Overload capacity of the robot What is the minimum overload capacity of the robot. Overload capacity means load capacity beyond the recommended specification.	kg

Structure-Attributes

Type of end Effectors List of End effectors that the robot should be able to use.	
Number of end effectors Number of tips the robot needs.	
Slip sensors Does the robot need slip sensors?	Y / N
Mounting Position To which surfaces should the robot be able to attach to? (for example ground, wall, ceiling and mobile platforms.)	
Tactile Sensors Does the robot need tactile sensors?	Y / N
Material Of Robot The outer surface of the robot should consist of which material(s)?	
Degree of Protection To which degree of protection should the robot belong? Degree of protection denotes the protection provided by an enclosure against access to hazardous parts, ingress of solid foreign objects and water. (IP-Code) (See appendix A.)	

External Requirements – Attributes

Ambient Temperature Range of temperature of environment in which the robot should be functioning.	ºC - ºC
Space Requirements How much space does the robot has at this task?	Length: mm Width: mm Height: mm
Recommended Operating Humidity Range of humidity of the environment in which the robot should be functioning.	% - %

Automation - Attributes

Adaptability What is the maximum allowed duration to change the configuration of the robot to perform different operations or at different locations?	sec
Autonomy What level of autonomy should the robot have? Autonomy is described as the capability of an entity to create and control the execution of its own plans and/or strategies. For instance, in mobile robots the ability of the robot for determining the trajectory to reach a specific location or pose. See **Appendix** for the definitions and the different levels of autonomy.	

Precision - Attributes

Sensing Range Distance to recognise a predefined shape, image or event. What sensing range is needed?	mm
Accuracy What is the maximum accuracy the robot can have? Accuracy is the measure of closeness between the robot end effectors and the target point, and can usually be defined as the distance between the target point and the centre of all points to which the robot goes on repeated trials.	mm
Repeatability What is the maximum repeatability the robot can have? Repeatability is the measure of the ability of a robot to return to the same position and orientation over and over again.	mm
Resolution What is the maximum resolution the robot can have?The smallest incremental motion which can be produced by the manipulator. Serves as one indication of the manipulator accuracy. Three factors determine the resolution: mechanical resolution, control resolution, and programming resolution.	mm
Stability This is the measure of absence of oscillations at termination of arm movement. What is the maximum stability the robot should have?	mm

Compliance	mm
This is the measure of displacement of the end effector in response to force (or torque) applied to it. What is the maximum compliance that the robot can have?	

Reach and Distances - Attributes

Degrees of Freedom	axes
A term used to describe a robot's freedom of motion in three-dimensional space, specifically the ability to move forward and backward, up and down, and to the left and to the right. • 1-axis robot: Linear guide system for transferring parts in a single line of motion. • 2-axis robot: Typically in an XY or YZ configuration, these are often in the form of two adjoining linear guides. • 3-axis robot: Typically in an XYZ configuration, these tend to be in the form of two adjoining linear guides and a third axis guide or cylinder. • 4-axis robot: A more conventional arm that is typically used in palletizing applications in which the face plate is always parallel with the ground. Has the ability to rotate the object it is picking. • 5-axis robot: Similar to a conventional four-axis robot but adds the ability to rotate the object it is picking. • 6-axis robot: Offers the most flexibility with six axes all the way from the base axis for full robot rotation to the sixth axis for rotating the "wrist" or faceplate. • 7-axis robot: A six-axis robot which is placed on a rail or some means to move it from one place to another in a linear direction.	
Horizontal Reach	mm
What is the minimum horizontal reach the robot should have?	
Vertical Reach	mm
What is the minimum vertical reach a robot should have?	
Wrist Reach Distance	mm
Maximum distance in which the end effector of the robot arm reaches in the work envelop. What is the minimum wrist reach distance a robot should have?	

To describe the degree of protection of a robot, a IP code can be created, and should look like IPxx. To fill in the x's, use the following tables.

1st numeral: Protection against contact and against solid foreign objects.	IP numeral
0 Non-protected	IP 0X(X)
1 Protected against solid foreign objects of ≥ 50 mm ø	IP 1X(X)
2 Protected against solid foreign objects of ≥ 12.5 mm ø	IP 2X(X)
3 Protected against solid foreign objects of ≥ 2.5 mm ø	IP 3X(X)
4 Protected against solid foreign objects of ≥ 1.0 mm ø	IP 4X(X)
5 Protected against ingress of dust (dust-protected)	IP 5X(X)
6 Protected against ingress of dust by underpressure (dust-tight)	IP 6X(X)

2nd numeral: Protection against ingress of water.	IP numeral
0 Non-protected	IP X0(X)
1 Protected against vertically falling water drops	IP X1(X)
2 Protected against vertically falling water drops when enclosure tilted up to 15°	IP X2(X)
3 Protected against spraying water	IP X3(X)
4 Protected against splashing water	IP X4(X)
5 Protected against water jets	IP X5(X)
6 Protected against powerful water jets	IP X6(X)
7 Protected against the effects of temporary immersion in water	IPX7(X)
8 Protected against the effects of continuous immersion in water	IPX8(X)

LORA	Sense	Plan	Act	Description	Examples from Literature
Manual	H	H	H	The human performs all aspects of the task including sensing the environment, generating plans/options/goals, and implementing processes.	"Manual Control" Endsley & Kaber, 1999
Tele-operation	H/R	H	H/R	The robot assists the human with action implementation. However, sensing and planning is allocated to the human. For example, a human may teleoperate a robot, but the human may choose to prompt the robot to assist with some aspects of a task (e.g., gripping objects).	"Action Support" Endsley & Kaber, 1999; Kaber et al., 2000; "Manual Teleoperation" Milgram, 1995; "Tele Mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005
Assisted Tele-operation	H/R	H	H/R	The human assists with all aspects of the task. However, the robot senses the environment and chooses to intervene with task. For example, if the user navigates the robot too close to an obstacle, the robot will automatically steer to avoid collision.	"Assisted Teleoperation" Takayama et al., 2011; "Safe Mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005
Batch Processing	H/R	H	R	Both the human and robot monitor and sense the environment. The human, however, determines the goals and plans of the task. The robot then implements the task.	"Batch Processing" Endsley & Kaber, 1999; Kaber et al., 2000
Decision Support	H/R	H/R	R	Both the human and robot sense the environment and generate a task plan. However, the human chooses the task plan and commands the robot to implement actions.	"Decision Support" Endlsey & Kaber, 1999; Kaber et al., 2000
Shared Control With Human Initiative	H/R	H/R	R	The robot autonomously senses the environment, develops plans and goals, and implements actions. However, the human monitors the robot's progress and may intervene and influence the robot with new goals and plans if the robot is having difficulty.	"Shared Mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005; "Mixed Initiative" Sellner et al., 2006; "Control Sharing" Tarn et al., 1995
Shared Control With Robot Initiative	H/R	H/R	R	The robot performs all aspects of the task (sense, plan, act). If the robot encounters difficulty, it can prompt the human for assistance in setting new goals and plans.	"System-Initiative" Sellner et al., 2006; "Fixed-Subtask Mixed-Initiative" Hearst, 1999
Executive Control	R	H/R	R	The human may give an abstract high-level goal (e.g., navigate in environment to a specified location). The robot autonomously senses environment, sets the plan, and implements action.	"Seamless Autonomy" Few et al., 2008; "Autonomous mode" Baker & Yanco, 2004; Bruemmer et al., 2005; Desai & Yanco, 2005
Supervisory Control	H/R	R	R	The robot performs all aspects of task, but the human continuously monitors the robot, environment, and task. The human has override capability and may set a new goal and plan. In this case, the autonomy would shift to executive control, shared control, or decision support.	"Supervisory Control" Endsley & Kaber, 1999; Kaber et al., 2000
Full Autonomy	R	R	R	The robot performs all aspects of a task autonomously without human intervention with sensing, planning, or implementing action.	"Full Automation" Endsley & Kaber, 1999

*Note: H = Human, R = Robot. Manual represents a situation where no robot is involved in performing the task; this level is included for a complete taxonomy continuum.