Worker-Robot Cooperation and Integration into the Manufacturing Workcell via the Holonic Control Architecture

by

Ahmed Rabee Ahmed Sadik, born on the 9th of September 1984, in Cairo, Egypt

Dissertation zur
Erlangung des akademischen Grades
Doktor-Ingenieur (Dr.-Ing.)
der Fakultät für Informatik und Elektrotechnik der
Universität Rostock

Rostock, Germany
21 August, 2018

https://doi.org/10.18453/rosdok_id00002421

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

﴿قُلْ لَوْ كَانَ الْبَحْرُ مِدَاداً لِكَلِمَاتِ رَبِّي لَنَفِدَ الْبَحْرُ قَبْلَ أَنْ تَنْفَدَ كَلِمَاتُ رَبِّي وَلَوْ جِئْنَا بِمِثْلِهِ مَدَداً﴾

"Smartness is not enough, Hard Working is not enough. Smart Hard Working is the Key"

To My Parents, My Brothers, My Wife, and My Son

Examiners:

Prof. Dr.-Ing. Bodo Urban
University of Rostock, Germany

Prof. Dr.-Ing. Kerstin Thurow
University of Rostock, Germany

Prof. Dr.-Ing. Ute Dietrich
HTW Berlin University of Applied Sciences, Germany

Date of Submission:

21. August 2018

Date of Defense:

31. January 2019

Keywords:

Worker-Robot Cooperation, Cooperative Robotics, Smart Manufacturing, Holonic Control Architecture

Classification (ACM CCS2012):

Information systems ➔ Information systems applications ➔ Decision support systems ➔ Expert systems

Abstract

There is no doubt that the rapid development in robotics technology has dramatically changed the interaction model between the robot and the worker. The current robotics technology affords very reliable means to guarantee the physical safety of the worker during a close proximity interaction with the industrial robot. Therefore, new forms of cooperation between the worker and the industrial robot can now be achieved. Cooperative and collaborative robotics are the new fields in industrial robotics, which empowers the idea of close human-robot interaction in manufacturing. The two fields involve the use of a collaborative robot (cobot). The cobot is a social lightweight industrial robot that can cooperate safely with the human co-worker. This is in contrast with the conventional industrial robot that is dangerous to operate in a direct contact with the worker, therefore it often operates in isolation from the worker. The difference between the cooperative and collaborative manufacturing is that in cooperative manufacturing, both the worker and the cobot are sequentially performing separate tasks over the same product in the same shared workspace. However, in collaborative manufacturing, they simultaneously perform the shared task. Cooperative manufacturing is the main focus of study in this dissertation.

Cooperative manufacturing adds a new dimension to the production system, which promotes the agility and the flexibility of the production model. The fast success of cooperative manufacturing is a natural result of the varying production demands, which requires high level of customizability. Gathering the worker and the cobot in the same manufacturing workcell can provide this production customizability. This is because the worker does not only add the high flexibility of taking the proper actions based on the production demands, but also the worker is able to use his natural senses intuitively to form complex solutions during the real-time of production. Simultaneously, the cobot is a reliable resource in terms of speed, accuracy, and weight lifting. In other words, cooperative manufacturing supports the use of the cobot as a smart tool by the worker, to increase the efficiency and accelerate the productivity of the manufacturing.

Cooperative manufacturing is a new field of research, which addresses new challenges beyond the physical safety of the worker. Those new challenges appear due to the need to connect the worker and the cobot from the informatics point of view in one cooperative workcell. This requires developing an appropriate manufacturing control system, which fits the nature of both the worker and the cobot. Furthermore, the manufacturing control system must be able to understand the production variations, to guide the cooperation between worker and the cobot and adapt with the production variations.

Designing a manufacturing control solution that enables the cooperation between the worker and the cobot is the main purpose of this dissertation. The design of this manufacturing control solution has been done over three levels. The first level is the control software component. In this level, an autonomous three layers software component is developed to link the worker and the cobot to the control solution. The three layers of the software component are physical, communication, and reasoning. The second level of the solution is the cooperative workcell where other sources of information are represented along with the worker and the cobot such as the product and the manufacturing operations and tasks. Finally, the last level of the solution is the industrial enterprise where more than one cooperative workcell must coordinate together. Ultimately, three case studies have been introduced to test the viability and the feasibility of the proposed control solution.

Kurzfassung

Es besteht kein Zweifel, dass die rasante Entwicklung in der Robotertechnologie das Interaktionsmodell zwischen Roboter und Arbeiter dramatisch verändert hat. Die aktuelle Robotik-Technologie bietet sehr zuverlässige Mittel, um die physische Sicherheit des Arbeiters während einer nahen Interaktion mit dem Industrieroboter zu gewährleisten. Damit können neue Formen der Zusammenarbeit zwischen Arbeiter und Industrieroboter erreicht werden. Kooperative und kollaborative Robotik sind die neuen Bereiche der Industrierobotik, die die Idee einer engen Mensch-Roboter-Interaktion in der Fertigung ermöglichen. Die beiden Bereiche beinhalten den Einsatz eines kollaborativen Roboters (Cobot). Der Cobot ist ein sozialer, leichter Industrieroboter, der sicher mit dem menschlichen Mitarbeiter zusammenarbeiten kann. Der Cobot steht im Gegensatz zum herkömmlichen Industrieroboter, welcher in direkter Zusammenarbeit mit dem Arbeiter für diesen gefährlich ist und daher oft isoliert vom Arbeiter arbeitet. Der Unterschied zwischen der kooperativen und der kollaborativen Fertigung besteht darin, dass in der kooperativen Fertigung sowohl der Arbeiter als auch der Cobot sequenziell getrennt Aufgaben am gleichen Produkt im gleichen gemeinsamen Arbeitsbereich ausführen. In der kollaborativen Fertigung erfüllen sie jedoch gleichzeitig die gemeinsame Aufgabe. Der Schwerpunkt der Dissertation liegt in der kooperativen Fertigung.

Die kooperative Fertigung verleiht dem Produktionssystem eine neue Dimension, die die Agilität und Flexibilität des Produktionsmodells fördert. Der schnelle Erfolg der kooperativen Fertigung ist eine natürliche Folge der unterschiedlichen Produktionsanforderungen, die ein hohes Maß an Anpassungsfähigkeit erfordert. Das Zusammenarbeiten von Arbeiter und Cobots im gleichen Arbeitsbereich, kann diese Produktionsanpassung ermöglichen. Denn der Arbeiter hat nicht nur die nötige, hohe Flexibilität, auf Grundlage der Produktionsanforderungen die richtigen Maßnahmen zu ergreifen, sondern er ist auch in der Lage, seine natürlichen Sinne intuitiv zu nutzen, um komplexe Lösungen während der Produktion in Echtzeit zu bilden. Gleichzeitig ist der Cobot eine zuverlässige Ressource in Bezug auf Geschwindigkeit, Genauigkeit und Gewichtheben. Mit anderen Worten, die kooperative Fertigung unterstützt den Einsatz des Cobots als intelligentes Werkzeug des Arbeiters, um die Effizienz zu steigern und die Produktivität der Fertigung zu beschleunigen.

Die kooperative Fertigung ist ein neues Forschungsgebiet, dass sich neuen Herausforderungen jenseits der physischen Sicherheit des Arbeitnehmers stellt. Diese neuen Herausforderungen ergeben sich aus der Notwendigkeit, den Arbeiter und den Cobot aus der Sicht der Informatik in einem kooperativen Arbeitsplatz zu verbinden. Dies erfordert die Entwicklung eines geeigneten Produktionskontrollsystems, das sowohl der Natur des

Arbeiters als auch der des Cobots entspricht. Darüber hinaus muss die Fertigungssteuerung in der Lage sein, die Produktionsschwankungen zu verstehen, um die Zusammenarbeit zwischen Arbeiter und Cobot zu steuern und sich an die Produktionsvarianten anzupassen.

Das Hauptziel dieser Dissertation ist die Entwicklung einer Fertigungssteuerungslösung, die die Zusammenarbeit zwischen dem Arbeiter und dem Cobot ermöglicht. Das Design dieser Fertigungssteuerungslösung wurde über drei Ebenen realisiert. Die erste Ebene ist die Kontrollsoftware Komponente. Hier wird eine autonome dreischichtige Softwarekomponente entwickelt, die den Arbeiter und den Cobot mit der Steuerungslösung verbindet. Die drei Schichten der Softwarekomponente sind physisch, kommunikativ und logisch. Die zweite Ebene der Lösung ist der kooperative Arbeitsplatz, in der neben dem Arbeiter und dem Cobot auch andere Informationsquellen wie das Produkt und die Fertigungsvorgänge und -aufgaben dargestellt werden. Die letzte Ebene der Lösung ist schließlich das Industrieunternehmen, in dem mehr als ein kooperativer Arbeitsplatz koordiniert werden muss. Schließlich wurden drei Fallstudien vorgestellt, um die Durchführbarkeit und Machbarkeit der vorgeschlagenen Kontrolllösung zu testen.

Acknowledgment

I always believed that we are all reasons for something bigger than us and so am I. Hence, the completion of this dissertation would never be achieved without the support and the contribution of my supervisor, my colleagues, and my family.

The actual work in this research started in December 2014, the implementation of this research has been done at Fraunhofer Institute for Computer Graphic Research IGD. The research has been supported by the German Federal State of Macklenburg-Western Pomerania and the European Social Fund under grant ESF/IV-BM-B35-0006/12 and by grants from the University of Rostock and Fraunhofer IGD.

First, I would like to express my gratitude for Professor Urban. Simply, this dissertation will never exist without him. The main idea of the dissertation to bring the worker and the robot together at the same work environment has been originally proposed by him. During my dissertation work, I was always impressed by Professor Urban's vision of the future of the industry in general. His vision drew a clear line for me to follow during my research and inspired me with many new ideas.

Second, I would like to thank all the members of the Department of Visual Assistance Technologies in Fraunhofer-IGD-Rostock, who helped my work with all the available means. I would love to especially thank MSc. Marian Haescher, MSc. Omar Adel, MSc. Andrei Taramov, PhD. Gerald Bieber, PhD. Jörg Voskamp, PhD. Mario Aehnelt, Henry Kraemer, the IT staff, and the secretariat of the department. Furthermore, I would love to admit I was so lucky to get my doctoral studies in Deutschland. After the time I lived here, I can feel a part of me is belonging to the country.

Finally, I dedicate this dissertation for the people who loved me unconditionally. For my mother Fatama who blessed my life with her prayers and kindness. For my father Rabee who did all what he could to make me the man who I am today. For my wife Natalia who immersed me with her love, passion, and care. For my brothers Ayman, Eslam, and Osama who always supported my back. For my son Rayan who was just born before defending this dissertation.

Glossary of Terms

Manufacturing System: a collection of machines, workers, tools, and material handlers, which operate together to establish a specific manufacturing sequence that results as in an end product.

Cellular Manufacturing: is a manufacturing technique where a bundle of workcells are routed together to produce a specific product, as every workcell is preforming a certain process. Different workcell routes can be defined to produce different products over the same production line.

Manufacturing Workcell: a small, self-contained modular manufacturing unit that contains several operational resources and manufacturing operations.

Flexible Manufacturing Workcell: a workcell that affords a high level of product customizability as it can manufacture a product from scratch.

Reconfigurable Manufacturing Workcell: a workcell that affords a moderate level of product customizability as it can manufacture a family of similar products.

Operational Resource: a physical entity in the workcell, which performs a specific production operation. An operational resource can be a machine, a robot, a worker, etc.

Production Scheduling: an optimal allocation of the operational resources to their assigned jobs over the production time.

Cobot: a light-weight robot which is capable of operating safely with a human co-worker in a shared work environment.

Robot Degrees of Freedom: the number of independent directions or joints of the robot, which allow the robot to move its end effector through the required sequence of motions.

Robot Accuracy: the measurement of the deviation between the command characteristic and the attained characteristic.

Robot Payload: the maximum mass that the robot can manipulate at a specified speed, acceleration/deceleration, centre of gravity location, and repeatability.

Cooperative Workcell: a workcell where at least one worker is cooperating with one cobot and the other operational resources to achieve the production process.

Shop Floor: the area of the factory where the production takes place.

Cooperative Enterprise: an industrial enterprise whose shop floor composes of many cooperative workcells.

Manufacturing Control: the decision-making process that is required to transform the customer orders into products. This is done by monitoring and planning the manufacturing disturbance.

Manufacturing Disturbance: an intentional or unintentional alternation in the manufacturing conditions.

Control System Architecture: a conceptual model that defines the structured layers of the control solution and the function of every layer.

Control Software Component: is a software building block that is used to build the control system architecture.

Control System Modularity: the control solution is developed as modules or components that can be easily connected to or separated from each other.

Control System Customizability: the control solution is flexible to be changed and edited during the operation.

Control System Reusability: the control solution is generic enough to be reused in the same or in another similar case without or with the minimum customization.

Control System Interoperability: the ability of the control solution to be integrated and to communicate with the other control solutions.

Control System Scalability: the ability of the control solution to be scaled to fit the size of the application. A system can be scaled up by adding more modules that are similar, or scaled down by removing replicated modules.

Control System Extensibility: the ability of the control solution to be extended by including more functions and integrating more control modules (i.e., by plug and play).

Control System Fault-diagnosis: the ability of the control solution to automatically recognize the current state of the controlled system, so it detects and diagnoses the disturbance from the production and the operations, to subsequently correct them quickly.

Control System Robustness: the ability of the control solution to continuously and correctly compensate for the manufacturing disturbance.

Control System Reliability: the ability of the control solution to continuously accomplish its function without failure.

Control System Agility: the ability of the control solution to rapidly respond to the manufacturing disturbance.

Control System Fault-tolerance: the ability of the control solution to tolerate a failure of one or more of its components and to continue to accomplish its function correctly - perhaps in degenerated state.

Contents

List of Figures

List of Tables

List of Abbreviations

3D	Three Dimensional
ADACOR	ADAptive-holonic-COntrol-aRchitecture
AI	Artificial Intelligent
AID	Agent Identifier
AMS	Agent Management System
AP	Agent Platform
AR	Augmented Reality
CAD/CAM	Computer-Aided Design/Computer-Aided Manufacturing
CH	Customer Holon
CNC	Computer Numerical Control
Cobot	Collaborative robot
DAG	Directed Acyclic Graph
DCS	Distributed Control System
DF	Directory Facilitator
DMS	Dedicated Manufacturing System
DOF	Degree of Freedom
EH	Energy Holon
ERP	Enterprise Resource Planning.
FB	Function Block
FBDK	Function Block Development Kit
FCFS	First Come First Serve
FIPA	Foundation for Intelligent Physical Agent
FIPA-ACL	FIPA-Agent Communication Language
FMS	Flexible Manufacturing System
GASP	Graph-based Assembly Sequence Planner
GUI	Graphical User Interface
H2A	Human to Application
HCA	Holonic Control Architecture
HMI	Human-Machine Interaction
HRC	Human-Robot Cooperation
HRI	Human-Robot Interaction
HTTP	Hyper Text Transport Protocol
I/O	Input/Output
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IH	Inventory Holon
IMS	Intelligent Manufacturing System
IR	Industrial Robot
ISA	International Society of Automation
ISO	International Organisation for Standardization
JADE	Java Agent DEvelopment
KPI	Key Production Indicator
MAS	Multi-Agent System

MES	Manufacturing Execution System
MOM	Manufacturing Operations Management
MTS	Message Transport Service
NIST	National Institute of Standards and Technology
OH	Order Holon
OOP	Object-Oriented Programming
OPC	Open Platform Communications
ORH	Operational Resource Holon
OS	Operating System
OWL	Web Ontology Language
P2P	Peer-to-Peer
PAC	Programmable Automation Controller
PH	Product Holon
PLC	Programmable Logic Controller
PROSA	Resource-Order-Staff-Architecture
RDF	Resource Description Framework
RE	Reasoning Engine
RH	Robot Holon
RMS	Reconfigurable Manufacturing System
ROS	Robot Operating System
SCADA	Supervisory Control and Data Acquisition
SDK	Software Development Kit
SH	Staff/Supervisor Holon
SME	Small and Medium Enterprise
SMLC	Smart Manufacturing Leadership Coalition
SMS	Smart Manufacturing System
SOAP	Simple Object Access Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TS	Technical Specification
UDDI	Description, Discovery and Integration
UML	Unified Modeling Language
UX	User eXperience
WH	Worker Holon
WLAN	Wireless Local Area Network

List of Symbols

J_i	Production Job Number i
n_i	Number of Units in Production Order Number i
O_i	Production Order Number i
RE_A	Robot Allocation Efficiency
RE_O	Robot Operation Efficiency
RE_T	Robot Technical Efficiency
RE_U	Robot Utility Efficiency
RR_P	Robot Production Rate
RT_{AO}	Robot Actual Operation Time
RT_B	Robot Busy Time
RTC	Robot Time Constant
RT_D	Robot Delay Time
RT_{OO}	Robot Overall Operation Time
RT_{Pi}	Robot Processing Time for Production Order Number i
ST	Setup Time
T_{AOi}	Actual Operation Time Number i
T_{Bi}	Busy Time Number i
T_{Di}	Down Time Number i
T_{Ii}	Idle Time Number i
T_{OOi}	Overall Operation Time Number i
T_{Pi}	Processing Time Number i
T_{Si}	Setup Time Number i
WE_A	Worker Allocation Efficiency
WE_O	Worker Operation Efficiency
WE_T	Worker Technical Efficiency
WE_U	Worker Utility Efficiency
WMTV	Worker Mean Time Value
WR_P	Worker Production Rate
WT_{AO}	Worker Actual Operation Time
WT_B	Worker Busy Time
WT_D	Worker Delay Time
WT_{OO}	Worker Overall Operation Time
WT_{Pi}	Worker Processing Time for Production Order Number i

Chapter 1 - Introduction and Challenges

"Logic will get you from A to B. Imagination will take you everywhere."

Albert Einstein

1.1 Introduction

From the beginning of humanity, the development of the human race is very dependent on the available tools. The human is not only inventing or looking for new helping tools in everyday activities, but also he is actively searching for new ways to use the existing tools. Machines are the best example of a variety of tools that define our current present and future, and drew our past. Robots in particular are forms of machines that have been designed to mimic the human capabilities. The robot in its modern forms as we know today is a relatively recent invention. However, the idea of a machine in the form of human or other creatures is very eminent through history. The first modern robotic arm has been invented in 1952 by G. Devol and named as "Unimate". The mechanical structure of this robotic arm is not so different from the ones that exist nowadays. However, it took around ten years to convince the contemporary industry at that time to implement this robotic arm in the production. 1962 was the actual involvement of Unimate in the production lines of General Motors, which was the bounce to the second industrial revolution. After that, the fields and the applications of the robots have been vastly increased. This created new types and categories of robots that can be seen in everyday life. Nevertheless, the industry, in particular the manufacturing was and still is the main inertia to accelerate the progress of robotics. Hence, the progress in industrial robotics evolved from the first generation of **I**ndustrial **R**obots (**IR**s), which were dump robots, to the second generation that are known as clever robots, to finally the third generation that are smart enough to be called **co**llaborative ro**bot**s (**cobot**s).

Before the existence of machines and robots, the human worker was the main engine of industry. The invention of the IR was an attempt to replace the worker, who performs repeatable jobs that require physical strength. When the industrial robotics became mature enough, the industry tended to replace the workers with robots in fully automated production lines. However, at that time the market was hungry for basic standard products with very little customization level. This manufacturing technique is precisely known as the mass production. Mass production never succeeded to completely replace the worker for two main reasons. First, the availability of the technology, as in many cases the available technology could not provide the same level of intelligence that the human worker can afford. Second, the cost of the technology, as many complex manufacturing tasks can be theoretically automated. Nevertheless, the cost of the automation would be dramatically huge, yet inefficient in comparison with the human worker.

At the beginning of the nineties, continuous mass production paradigm could not cope with the saturated market demands, which led to the appearance of new paradigms such as just in time and lean production. The main idea of these production paradigms is to divide the continuous rigid production line into a group of manufacturing workcells. This technique is known in industry as cellular manufacturing. Cellular manufacturing tends to produce a variety of similar products and adjust the production rhythm as quick as possible to adapt with the market needs. Simultaneously, it aims to maximize the use of all the available operational resources within the workcell, and reduces the waste of time, effort, cost, and space. Toyota was the first pioneer to implement the cellular manufacturing concept, which was the main reason to transform the company from a small automotive manufacturer to one of the market leaders. Common terms to hear nowadays such as **S**mart **M**anufacturing **S**ystem (**SMS**) and Industry 4.0 are a natural evolution of cellular manufacturing. SMS can be seen as a bundle of cooperative workcells that deploy a variety of growing technologies in advanced robotics, big data analysis, and industrial information technology. Industry 4.0, the fourth manufacturing revolution, and the factory of the future are all direct implications of the smart manufacturing paradigm.

In smart manufacturing, the role of the IR is different from traditional manufacturing. In traditional manufacturing, either the IR is a replacement of the worker, or it operates separately from the worker due to the safety regulations. Separating the worker and the IR reduces the value of both. This is why in smart manufacturing both the worker and the IR must cooperate in the same workcell. This cooperation is no longer impossible thanks to the appearance of the cobots. Even that the cobots have been recently introduced into the commercial robotics market, they obtained a remarkable attention. Currently, the sales of cobots represent 5% of the overall industrial robotics market, with strong growth expectations in the future. Some cobots' manufacturer such as Universal Robots has been founded in 2005, but gained fast success and fame in a short time period. In 2015, Universal Robots achieved 91% growth compared to 2014 and 223% increase compared to 2013. These growth rates indicate that the industry is now in a transient shift towards cooperative manufacturing.

However, the field of cooperative manufacturing is still in progress, which proposes new challenges beyond the worker physical safety. Those challenges are mainly focused on connecting the cobot and the worker from the informatics point of view in one cooperative workcell. This requires a manufacturing control solution, which plans the information flow at the cooperative workcell, and constructs meaningful knowledge that can be reasoned by both the worker and the cobot. Therefore, the cooperative workcell is able to achieve a form of self-organisation to adapt with the production variations. Furthermore, the cooperative workcell must schedule the cooperative tasks to obtain the highest possible production efficiency. Finally, the cooperative workcell can be integrated with other workcells to form the cooperative enterprise structure.

1.2 Challenges and Problem Statement

Cooperative manufacturing is a direct application of the **H**uman-**R**obot **I**nteraction (**HRI**) Field. HRI focuses on understanding, designing and evaluating the robotics systems that involve the human as an essential element of these systems. HRI originates from literature roots, as the fundamentals of the HRI have been stated by the author Isaac Asimov in his novel "I, Robot" [Pin99]. The first two fundamental rules are as following:

- *A robot may not injure a human being or, through interaction, allow a human to come to harm.*
- *A robot must obey the orders, which are given to it by the human beings.*

Scientifically speaking, the first HRI fundamental rule explicitly addresses the problem of the physical safety of the human around the robot. The problem of the human physical safety has gained a remarkable focus of research during the last decades. Simply, because there is no interaction if the first fundamental rule was not fulfilled. The result of this research focus is many social robots that are available at the commercial market. However, the full leverage of HRI can never be reached as long as the second fundamental rule is not fulfilled. The second HRI fundamental rule addresses the problem of the information exchange and the communication between the human and the robot. In general, this problem did not get the fair amount of research focus yet, as the safe robots are relatively new in our daily life. Accordingly, the problem of this dissertation can be summarized in the following statement, *the cobot is a safe IR which needs a manufacturing control system that enables it to communicate with the worker and to be integrated into the manufacturing workcell. Otherwise, it will lose its value as a cooperative smart tool.* This problem statement derives many related challenges. Those challenges are the subjects of investigation in the dissertation as shown in Figure 1.1 and discussed follows:

- **Worker Physical and Psychological Safety**: Even that the worker safety is not the main topic of this dissertation, it will be included in the literature review. This is to provide an overall view of the advances in safety techniques in robotics, which is the initial point that leads to the other challenges in cooperative manufacturing. The current progress in safety techniques is high enough to develop well-known safety standards such as the **I**nternational **O**rganization for **S**tandardization (**ISO**)-10218, and **T**echnical **S**pecification (**TS**) such as ISO/TS 15066. These standards will be highlighted along with other safety techniques in section 3.1.1.

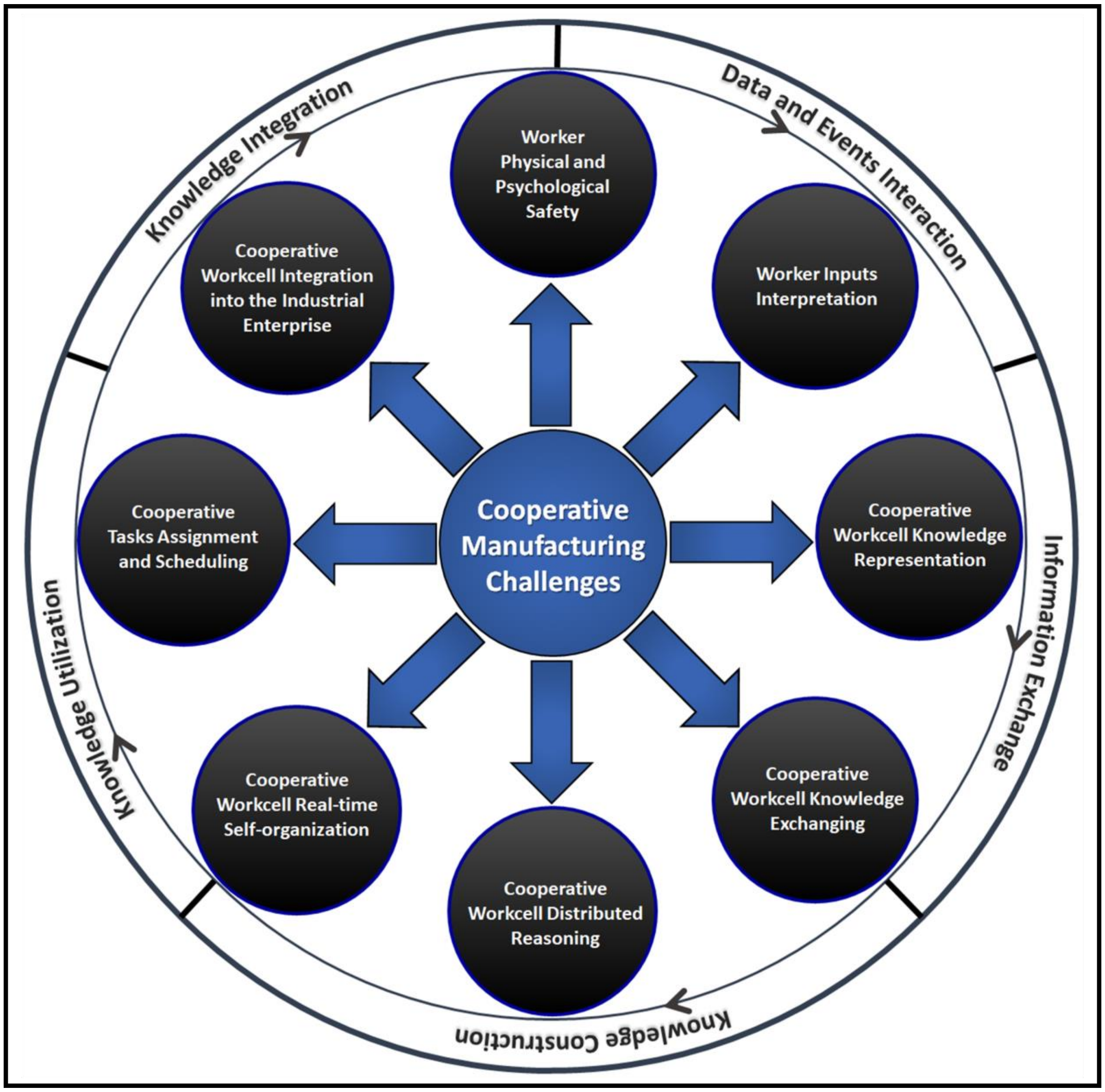

Knowledge Integration
Data and Events Interaction
Information Exchange
Knowledge Construction
Knowledge Utilization
Worker Physical and Psychological Safety
Worker Inputs Interpretation
Cooperative Workcell Knowledge Representation
Cooperative Workcell Knowledge Exchanging
Cooperative Workcell Distributed Reasoning
Cooperative Workcell Real-time Self-organization
Cooperative Tasks Assignment and Scheduling
Cooperative Workcell Integration into the Industrial Enterprise
Cooperative Manufacturing Challenges

Figure 1.1: Cooperative manufacturing challenges.

- **Worker Inputs Interpretation**: to integrate the worker to the manufacturing control system, a proper mean must exist to interpret his input to data and events that can be understood by the manufacturing control. In the reverse direction, a proper mean must exist to interpret the output from the manufacturing control system to the appropriate format that can be understood by the worker. Putting into consideration that the worker input during the cooperative manufacturing can be explicit when giving direct commands to the manufacturing control system, and implicit when the manufacturing control system automatically recognizes some of the worker's tasks as a part of the manufacturing scenario. For instance, an implicit worker input can be detected during performing a pick and place operation. In other words, this challenge aims to find the appropriate combination of sensory devices and **G**raphical **U**ser **I**nterfaces (**GUI**s) that link the worker to the manufacturing control system.

- **Cooperative Workcell Knowledge Representation:** the abstract representation of the facts and the objects within the cooperative workcell is an essential necessity to establish a successful cooperation. Thus, in order to accomplish the cooperative manufacturing concept, a proper approach is required to describe the shared environment between the worker and the cobot including themselves. In addition, the other production components such as the product parts and the manufacturing tools. Descriptive meta-data representation is essential to give a meaning of the objects, tasks, and operations within the cooperative workcell. Structural meta-data are required to structure bonds between the objects to compound new descriptive meta-objects, this also can be done by associating new attributes to an existing object. In addition, the structure meta-data can be used to define the parent-child relations among the objects. Finally, administrative meta-data are required to control the cooperative task assignment and the cooperation planning, management, and execution.
- **Cooperative Workcell Knowledge Exchange:** the knowledge is useless, unless it is exchanged and shared. This leads that the manufacturing control is the nervous system of the cooperative workcell, which provides a communication framework to connect the cobot, the worker, and the manufacturing components together in the same body. In the absence of the proper communication framework, the maximum usability of the cobot can never be reached, because it loses its real value as a smart tool. The challenge from the research point of view is to provide a communication language that can express the relations among the cooperative workcell components. Moreover, the actions and operations that can be performed by these components. Putting into consideration that the communication language should be human readable and at the same time can be processed by the machine (i.e., the cobot). The challenge from the technical point of view is to provide a communication mean that guarantees the industrial connectivity (i.e., the software interoperability).
- **Cooperative Workcell Distributed Reasoning:** the distributed intelligence is an essential requirement in the cooperative workcell. The main privilege of the cooperative manufacturing is to leverage the capabilities of the workcell by combining the skills of the cobot and the worker together at the same time. Accordingly, a collective decision-making is the main control theme. Distributing the reasoning among the cooperative workcell components is not only efficient, but also so fast from the processing point of view. Thus, every component in the cooperative workcell must have specific responsibilities and behaviours to fulfil. Furthermore, the cooperative workcell must be smart enough to build new facts based on this distributed reasoning.

- **Cooperative Workcell Real-time Self-organizing:** the main purpose of the cooperative workcell is to adapt with the variations in production demands and in the operational resources status. This means that the manufacturing control system must comprehend these variations, and hence self-organize the cooperative workcell. The notion of self-organization in cooperative workcell simply means that the workcell can fulfil the production customization with the present operational resources. The variation in the production customization is a direct impact of the customer needs. While the variation in the operational resources comes from changing the number of operational resources during the production. This can be caused by production shifts, time breaks, or machines maintenance.
- **Cooperative Task Planning and Scheduling:** the nature of the cooperative tasks can be different due to the manufacturing scenario. However, there is no doubt that a cooperative task requires to assign the task to at least one cobot and one worker. Therefore, the task planning can be an ambiguous decision in case of many workers exist at the cooperative workcell. Cooperative task assignment influences the overall performance of the workcell. Therefore, it is important to find the right criteria to assign a cooperative task upon it. Cooperative tasks scheduling is another crucial challenge that affects the cooperative workcell performance. Cooperative workcell scheduling is a form of a flow shop problem that is well-known in production systems. The problem appears when a group of jobs shares the same processing sequence on two or more processing units sequentially. The main goal of workcell scheduling is to minimize the amount of delay, and hence decrease the overall makespan. Those delays are the result of a task waiting to be processed by a free operational resource, or a free operational resource waiting for a task to process. The challenge comes from the fact that the flow shop scheduling algorithms are based on knowing in advance the **P**rocessing **T**ime (**T_P**), which is not the case in the cooperative workcell. As the time needed from the worker to complete a task is always varying. Thus, the workcell manufacturing control system must be able to continuously record the **W**orker **P**rocessing **T**ime (**WT_P**) and predict the next WT_P, in order to schedule the cooperative tasks.
- **Cooperative Workcell Integration into the Industrial Enterprise:** the workcell is the smallest building unit of the industrial enterprise. Therefore, it is important to define a manufacturing system model that horizontally connects all the cooperative workcells over the shop floor, and vertically connects the different layers of the industrial enterprise. This model must guarantee the scalability and extensibility of the enterprise. Moreover, it must supervise the enterprise **K**ey **P**erformance **I**ndicators (**KPI**s), as the main terms to measure the enterprise performance.

1.3 *Roadmap and Focus*

The challenges in cooperative workcell are not only implying this research questions, but also they define the dissertation roadmap and focus as illustrated in Figure 1.2.

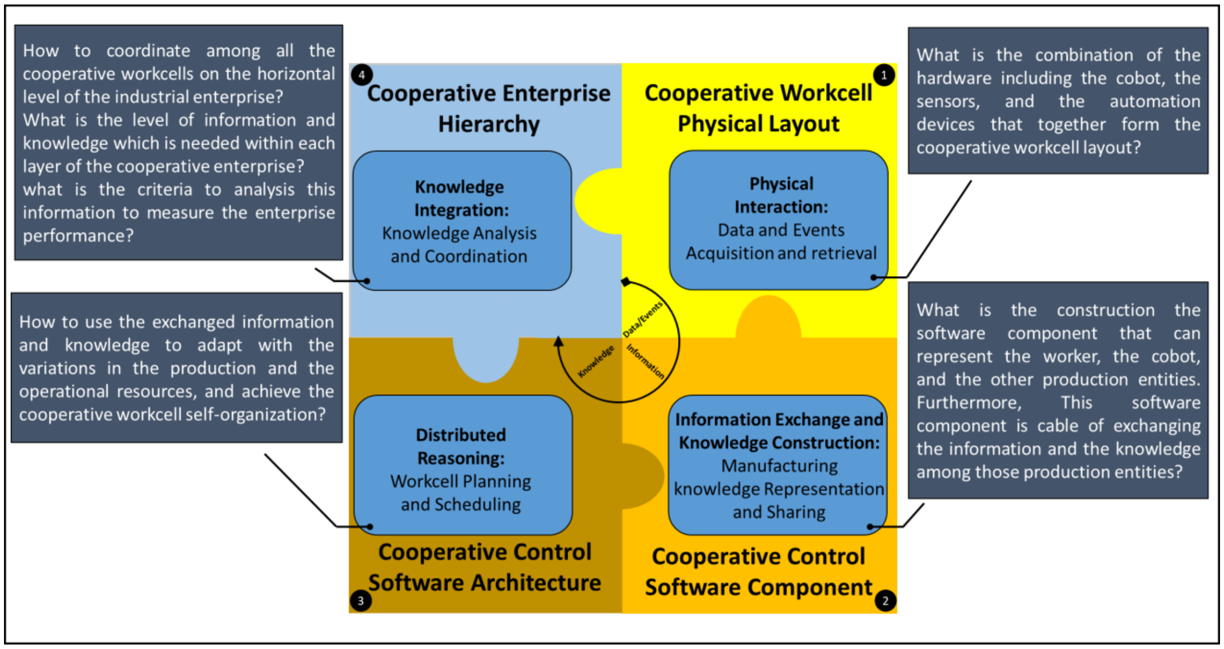

How to coordinate among all the cooperative workcells on the horizontal level of the industrial enterprise? What is the level of information and knowledge which is needed within each layer of the cooperative enterprise? what is the criteria to analysis this information to measure the enterprise performance?
How to use the exchanged information and knowledge to adapt with the variations in the production and the operational resources, and achieve the cooperative workcell self-organization?
4
Cooperative Enterprise Hierarchy
Knowledge Integration: Knowledge Analysis and Coordination
Cooperative Workcell Physical Layout
1
Physical Interaction: Data and Events Acquisition and retrieval
What is the combination of the hardware including the cobot, the sensors, and the automation devices that together form the cooperative workcell layout?
Data/Events
Information
Knowledge
Distributed Reasoning: Workcell Planning and Scheduling
Cooperative Control Software Architecture
3
Information Exchange and Knowledge Construction: Manufacturing knowledge Representation and Sharing
Cooperative Control Software Component
2
What is the construction the software component that can represent the worker, the cobot, and the other production entities. Furthermore, This software component is cable of exchanging the information and the knowledge among those production entities?

Figure 1.2: Dissertation roadmap and the research focus.

The first step during this research is to build the physical layout of the cooperative workcell. In order to accomplish that, it is important to investigate the available cobots in the commercial market and know their specifications, capabilities, and limitations. Selection of the appropriate cobot is an important factor to consider during the cooperative workcell design and it differs from one application to another. It is also important to consider the mean that will be used by the worker to physically interact within the cooperative workcell. This simply means how the worker inserts his input to the manufacturing control system, and how the manufacturing control system conducts its output to the worker. In addition to the cobot and the worker at the cooperative workcell structure, there might be machines such as parts feeders and conveyor belts. The automation devices control the **I**nput/**O**utput (**I/O**) of these machines. Therefore, the distributability nature of the cooperative workcell must be also considered while selecting the automation devices. As they have to communicate with one another and with the cobot controller during any cooperative manufacturing scenario. Studying the nature of the cooperative workcell physical layout is discussed in chapter 2 of this dissertation.

The second step after the cooperative workcell hardware is the software component [Su07]. Component-based software engineering uses the term software component to express a modular set of generic functions that are encapsulated in the same code. This technique is often used in service-oriented and event-driven architectures to develop a distributed manufacturing control system. Manufacturing control system design based on a generic software component guarantees many privileges such as the modularity, the reusability, the customizability, the scalability, and the extensibility. Component-based software development technique fits the nature of the cooperative workcell. This is because every entity with the cooperative workcell can be represented as a generic component. Thus, this generic component can be reused or customized to fit different cooperative workcell layouts or different production demands. The mission in this dissertation is to build the software conceptual and implementation models, which can be used as a guidance for further related researches. Accordingly, chapter 4 is focused on developing an autonomous and cooperative software component, which fits the nature of the cooperative workcell.

The third step in the dissertation roadmap is the cooperative workcell control architecture. The workcell control architecture is constructed in section 5.2 upon the previously mentioned software component to guarantee the flexibility from the software perspective. However, other aspects of flexibility must be considered while designing the cooperative workcell control architecture as well, such as the production and operational flexibility. Therefore, the control architecture must be able to compensate the variations in both the production and the operational conditions. This compensation must be accomplished without any interference that leads to the interruption of the production. Furthermore, the control architecture must consider optimizing the workcell performance while compensating those variations. The cooperative workcell flexibility, adaptability, and performance optimization are studies in details in section 5.3.

The last step is to provide a hierarchal model, which scales up the control architecture all over the industrial enterprise. A cooperative workcell can be seen as the building unit of the cooperative enterprise. Therefore, the model must define a method to coordinate the different cooperative workcells over the shop floor. Moreover, the model must consider the other software that might exist in a large-scale enterprise, such as the **M**anufacturing **O**perations **M**anagement (**MOM**) and the **E**nterprise **R**esource **P**lanning (**ERP**) software. Thus, the parameters that are passed by the manufacturing control solution to these software and vice-versa could be defined. Finally, the cooperative enterprise model must consider monitoring the production indicators, and then it is able to evaluate the success of the enterprise in qualitative and qualitative terms. As the cooperative enterprise size is beyond the implementation limits of this dissertation, the cooperative workcell concept is extended to the enterprise level in section 5.4. Nevertheless, chapter 6 is only focused on the implementation of the cooperative workcell, as it is within this research capability.

1.4 Layout and Contribution

Based on the previously discussed roadmap, the dissertation is organized in seven chapters. Chapter 1 has introduced the idea of cooperative manufacturing and addressed the potential challenges from the informatics point of view. The idea and the challenges in cooperative manufacturing have been considered as a scientific contribution in [SB16a, SB16b, and SB17d]. Chapter 2 tackles the main problem by dividing the research into two connected halves. The first half is the cooperative manufacturing, while the second half is the **H**olonic **C**ontrol **A**rchitecture (**HCA**). Accordingly, the grounds of both the cooperative manufacturing and the HCA are introduced to derive the motivation of the dissertation. The dissertation background and motivation have been an essential part of all the published articles during the dissertation time. In particular, [SB17b] has emphasised the motives behind the cooperative manufacturing.

Chapter 3 conducts the literature review of the related research, and it goes in the same rhythm of chapter 2 to ensure the consistency of the dissertation. Therefore, the first half of chapter 3 is a general investigation of the current advances in the factory of the future, and then it goes in more details into related research topics such as the safety and the manufacturing techniques in cooperative manufacturing. The gap between the cooperative workcell manufacturing control concept and implementation is emphasised during this half of chapter 3. Therefore, the second half of chapter 3 focuses on the HCA, which is a common solution for similar cases with the cooperative manufacturing such as the flexible and reconfigurable manufacturing systems.

Chapter 4 constructs the control software component. This is done by introducing the holon structure and transforms it to an implementation model, which fits in the cooperative manufacturing context. Thus, a generic three-layered software component has been introduced via this chapter. The first layer is the physical layer that is directly connected to the cooperative workcell I/O. Then, comes the communication layer that enables complex cooperation scenarios that might happen during the cooperative manufacturing. These complex interaction scenarios are mainly based on the ontology concept that is also discussed in details in this chapter. Finally, the last layer of the software component is the reasoning layer that guarantees the distributed intelligence of the cooperative workcell. Various technologies have been explained during this chapter, such as the **I**nternational **E**lectrotechnical **C**ommission (**IEC**) 61499, **R**obot **O**perating **S**ystem (**ROS**), **M**ulti-**A**gent **S**ystem (**MAS**), and Drools **R**easoning **E**ngine (**RE**) technologies. These technologies represent the preliminaries of the manufacturing control software architecture in chapter 5 as well. The contribution of chapter 4 has been published during developing the holonic software component in [SB16a, SB16b].

Chapter 5 builds the cooperative manufacturing control concept. This is done over four different levels. First by providing the basic definitions of specific terminologies that are related to this dissertation in general and to this chapter in particular. The second level of chapter 5 provides detailed and generic object-oriented models of the different types of the software component (i.e. holons), that are involved in the cooperative manufacturing. Those models will be the guidance to implement the solution concept in chapter 6. The third level of chapter 5 focuses on the different aspects to consider during the design of a cooperative workcell, such as the workcell layout, the sources of flexibility, and the optimal scheduling of the workcell. The final level of chapter 5 proposes the cooperative enterprise structure based on the **I**nternational **S**ociety of **A**utomation (**ISA**)-95 standard model. The sources of disturbance in the cooperative enterprise have been considered in developing this hierarchal model. In addition to the production's KPIs that can be monitored by the proposed manufacturing control system. The evolving of the solution concept has gone through different phases starting from [SB17a, SB17b], and passing by [SB17c, SB17e, and SB18b], until [SB18a, SBA18].

Chapter 6 shows three case studies to validate and test the concept that was developed in chapter 4 and chapter 5. Therefore, the first case study focuses on the feasibility of the proposed HCA to be implemented over the cooperative workcell hardware. Therefore, a case study of dual-arm cobot in cooperation with one worker has been introduced. Where, Baxter from rethink robotics is the cobot. While, the Leap Motion sensor is used as an input device to the worker. Thus, the worker used a group of explicit and implicit hand gestures, which are detected by the Leap Motion to interact with the HCA. The contribution as well as the case study has been published in [SBA17, SB17a]. The second case study aims to provide a solution for a complex manufacturing scenario, which requires more than a simple message communication. Thus, the case study proposes cooperative workcell that contains two workers in cooperation with one cobot. The goal of the case study is to show a cooperative assembly scenario of a family of products, which are either a centrifugal pump or a screw compressor. The contribution of this case study was published in [SB17d, SB17e, and SB18a]. The last case study shows an application of the proposed HCA. This application is to schedule the cooperative tasks between one cobot in cooperation with one worker via Johnson's scheduling rules. The goal of this case study is to show the importance of the HCA as it can leverage the cooperative workcell productivity. The details of this case study have been discussed and published in [SB17c, STB17].

Chapter 7 discusses and summarizes the dissertation to wrap it up with the outcomes and the challenges in the future work.

Chapter 2 - Background and Motivation

"You just can't differentiate between a robot and the very best of humans."

Isaac Asimov- I, Robot

2.1 *Cooperative Manufacturing*

Cooperative manufacturing is a multidisciplinary field that results 5 of research as shown in Figure 2.1.

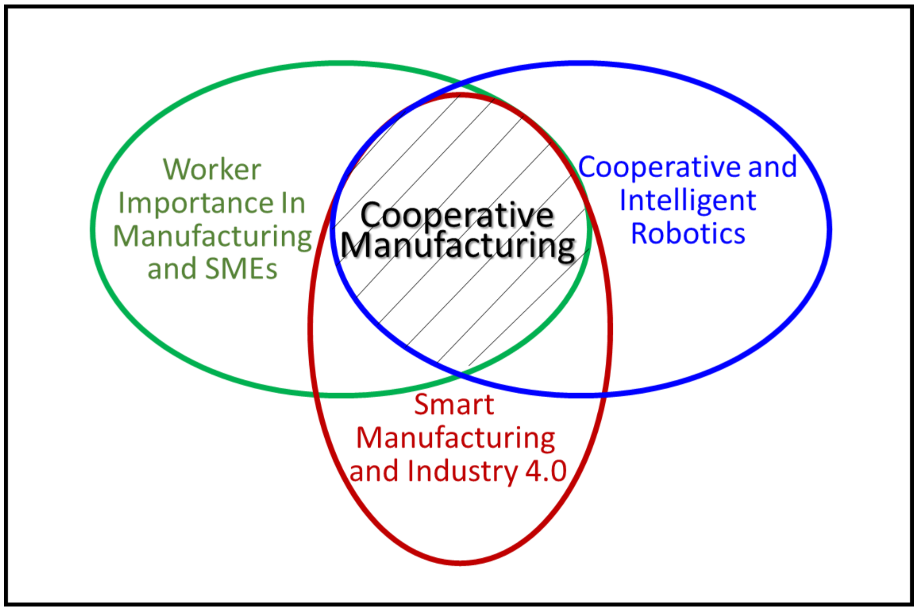

Worker Importance In Manufacturing and SMEs
Cooperative Manufacturing
Cooperative and Intelligent Robotics
Smart Manufacturing and Industry 4.0

Figure 2.1: Cooperative manufacturing as a multidisciplinary field.

One of industry 4.0 goals is to develop a cooperative factory platform [aca14]. The idea of a smart cooperative factory is to connect all the factory elements in one manufacturing control architecture. Thus, the factory is smart enough to understand the market needs and reshape its organisation to fulfil these needs with maximum performance. This idea requires a flexible shop floor structure, which is able to adapt as fast as possible to the production changes. The flexible shop floor is one of the main privileges that can be achieved via the cooperative manufacturing. The cooperative manufacturing offers this flexibility by gathering the advantages of both the cobot and the worker. The comparison in Table 2.1 shows how the worker's pros can eliminate the cobot's cons and vice-versa.

Table 2.1: Comparison between the worker and the cobot in manufacturing.

	Worker	Cobot
Pros	• Assimilates new tasks rapidly. • Capable of providing a situation judgment. • Able to move freely. • Able to tolerate compensation. • Can sense and locate things. • Can handle flexible parts. • Able to innovate. • Flexibly available.	• Consistent quality thanks to the automatic control. • Highly endurance. • Able to take unreasonable and monotonous tasks. • Can handle heavy loads. • Can operate in hazard environment.
Cons	• Limited quality and process control. • Mental and physical performance limits (e.g. fatigue). • High cost of training. • Needs a workshop jig to operate accurately.	• Limited movement. • Requires a predefined programming. • High cost of implementation, programming, and certification. • Function-oriented rather than goal-oriented. • Fixed cost even if the production load fluctuates.

During this chapter, the three intersected subjects that form the cooperative manufacturing will be discussed in details for better understanding of the dissertation topic.

2.1.1 Smart Manufacturing and Industry 4.0

The term smart manufacturing has been commonly used lately. Due to the **N**ational **I**nstitute of **S**tandard and **T**echnology (**NIST**), smart manufacturing is a group of systems that are fully-integrated, cooperative, and real-time adaptive to the change in the demands and conditions of the factory, the supply network, and the customer needs. The **S**mart **M**anufacturing **L**eadership **C**oalition (**SMLC**) defines the smart manufacturing as the ability of the manufacturing system to solve the existing and future problems via an open architecture that allows solutions to run at the speed of business while creating advantaged value [McK17b]. Smart manufacturing imposes three important fields into the industrial enterprises, which are data collection and analysis, advanced and autonomous robotics, and connectivity of the industrial devices and services [IBa17]. Due to the broadness of the smart manufacturing subjects, a subsequent term has recently appeared as Industry 4.0. Industry 4.0 is also known as the fourth industrial revolution as explained in Figure 2.2. Industry 4.0 provides a view of the factory of the future that is based on the smart manufacturing objectives [GTA14]. The most important aspects of this view can be summarized in the following points:

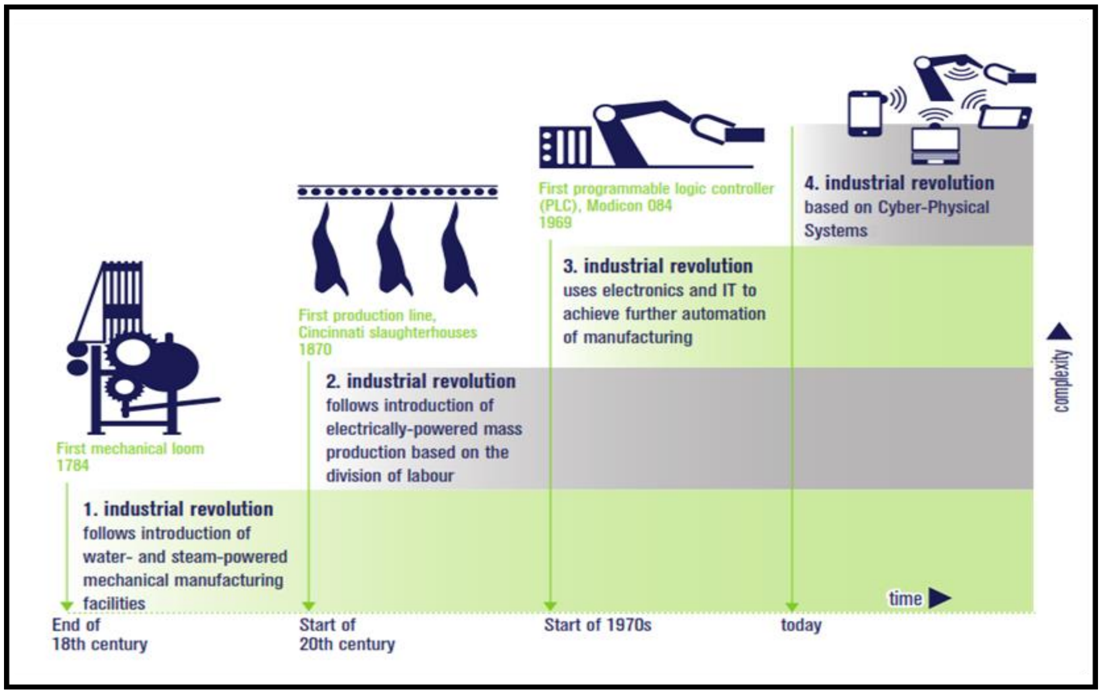

First mechanical loom
1784
1. industrial revolution
follows introduction of water- and steam-powered mechanical manufacturing facilities
First production line, Cincinnati slaughterhouses
1870
2. industrial revolution
follows introduction of electrically-powered mass production based on the division of labour
First programmable logic controller (PLC), Modicon 084
1969
3. industrial revolution
uses electronics and IT to achieve further automation of manufacturing
4. industrial revolution
based on Cyber-Physical Systems
complexity
time
End of 18th century
Start of 20th century
Start of 1970s
today

Figure 2.2: The four stages of the industrial revolution - source [aca14].

- **Cyber-physical systems and the internet of things**: this point supports the idea of the communication among all the existing entities within the industrial enterprise including the machines, the workers, the products, and the customers. Furthermore, gathering the information that is invoked during the communication to analysis and manage the enterprise.
- **Service oriented architecture**: as an aspect of the smart factory, every entity within the shop floor must be able to provide a group of manufacturing services and hence acquire other services. Therefore, every entity needs its own reasoning core.
- **Resources efficiency:** reducing the waste in time and effort due to the miscoordination among the resources. For instance, a free operational resource is waiting for a job while it is available, wrong scheduling of a group of jobs which are standing in a production order, or mismanaging the capabilities and the services that every resource can afford (i.e., who do what and where is it locates). The concept of resource efficiency has been originally established as a part of the lean manufacturing theory.
- **Advanced Human-Machine Interaction (HMI)**: using advanced techniques in HMI such as active vision techniques, capacitive and touch screen inputs, and **A**ugmented **R**eality (**AR**). Those techniques should afford the human natural and spontaneous means to interact with the machine.

- **Cooperative factory platform**: this point is branched into two directions. The first direction aims at providing an open and dynamic manufacturing control system architecture [GB04], which provides cooperation and coordination services among the resources, especially between the workers and the machines (in particularly the cooperative robots). Open manufacturing control architecture is meant to provide a generic conceptualization of the objects, relations, and actions that exit in the industrial enterprise. The purpose of this conceptualization is the flexibility in expanding the architecture with the minimum effort. The second direction is to establish a cooperative shop floor virtual services, this direction aims to provide a complete simulation and virtual testing of the production processes.
- **Standardizing and referencing the industrial enterprises**: standardizing and referencing are one of the crucial goals of Industry 4.0. This is because Industry 4.0 aims to solve the problem of software interoperability and hardware compatibility on the shop floor among the different automation manufacturers. Furthermore, it aims to provide a reference architecture that describes the control structure style of the industrial enterprise, which links the shop floor to the MOM, and the ERP.

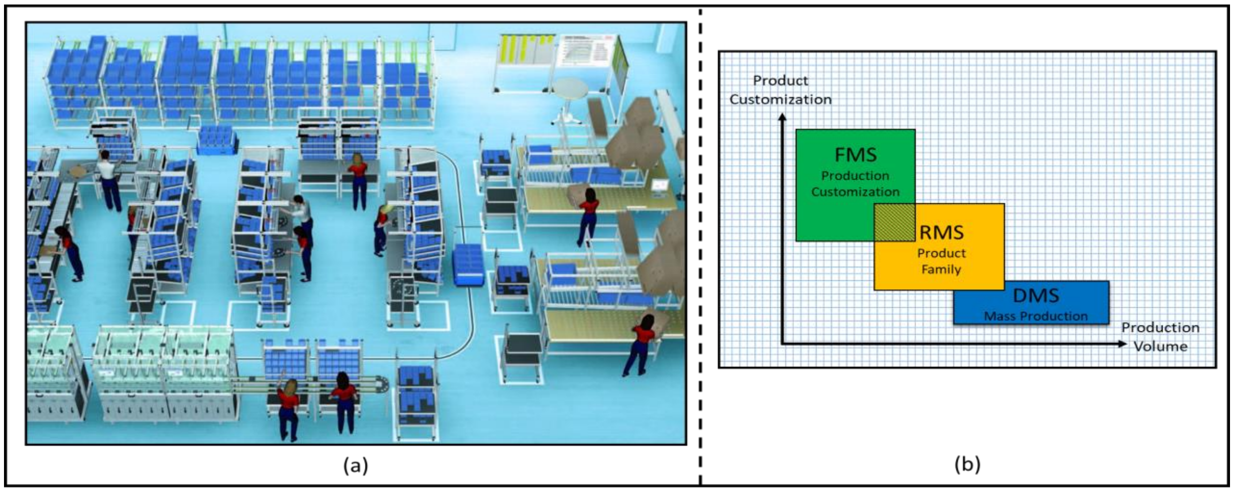


Figure 2.3: (a) Cellular manufacturing shop floor layout - schematic courtesy Bosch Rexroth - source [ass17] – (b) Manufacturing system paradigms - a hypothesis [Elm05].

Smart manufacturing and Industry 4.0 are based on the top of the cellular manufacturing ground. Cellular manufacturing divides the factory shop floor into workcells as shown in Figure 2.3a. A workcell is a self-contained modular unit that contains several operational resources and manufacturing operations. A workcell accomplishes a production process that is composed of a set of operations, as every operational resource within the workcell is responsible for performing a specific operation. Three manufacturing systems can be applied based on the cellular manufacturing that are **D**edicated **M**anufacturing **S**ystem (**DMS**), **F**lexible **M**anufacturing **S**ystem (**FMS**), and

Reconfigurable **M**anufacturing **S**ystem (**RMS**). The difference between the three systems is shown in Figure 2.3b. On the one hand, DMS is a manufacturing system with a fixed machine structure and static production schedules that aims to produce a specific product with high volume [Hu13]. DMS goal is the mass production that is only efficient in case of high market demand, which is not a feature of nowadays market. On the other hand, FMS is an integrated system of machine modules and material handling units that together offer a high degree of flexibility. FMS goal is the production customization, which aims to manufacture a product from scratch. Subsequently, the FMS production volume is low [LTL+94]. In the middle between the DMS and the FMS stands the RMS, where a family of products with shared physical aspects and the same operational sequence is manufactured. Therefore, RMS can afford a moderate production volume and a moderate customization [KJM+99, Kor06].

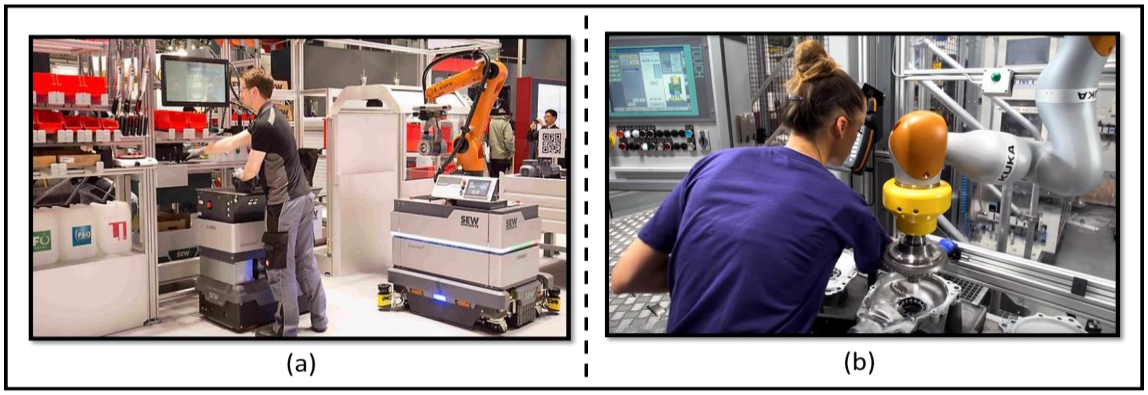


Figure 2.4: (a) SEW eurodrive cooperative workcell - source [You16b, Sew16] – (b) KUKA lightweight cobot in a flexible cooperative workcell- source [You16c, KUK16].

The concept of reconfigurable or flexible workcell empowers the idea of cooperative manufacturing. Therefore, a cooperative workcell is either flexible or reconfigurable, which depends on many factors such as the application and the size of the enterprise. Figure 2.4 shows two real case studies where a cooperative workcell is implemented. The first case study in Figure 2.4a shows a reconfigurable cooperative workcell where one worker is in cooperation with one customized autonomous IR. This reconfigurable cooperative workcell has been implemented by SEW eurodrive [Sew16] to assemble a family of geared electrical motors. In this example, the robot handles the worker the parts that are needed to assemble a customized electrical motor. The second case study that is shown in Figure 2.4b, is an example of a flexible cooperative workcell which is implemented by KUKA robotics [KUK16]. In this example a lightweight KUKA cobot is helping the worker to assemble a vehicle gearbox. The cobot function is to accurately mesh two helical gears, while the worker performs the rest of the assembly process.

2.1.2 Evolution of Industrial Robotics

The cobot is a natural evolution of the IR. Thus, it worth to briefly summarize the history of robotics, for better understand of the cobot purpose and to distinguish between the robot and the IR. The concept of mechanical creatures and beings can be seen in very old civilizations such as ancient Egypt, Greece, and China. Passing by the era of Leonardo Da Vinci who sketched a mechanical man around 1495, to Nicola Tesla who first demonstrated a radio-controlled boat. However, the term robotic has been mentioned for the first time in 1941 by the science fiction writer Issac Asimov to emphasis an idea of a new generation of intelligent machines that can take control of their actions and act autonomously. The term robot comes from the word robota in the Slavic language, the writer Karel Capek in his fiction "Rossunis Universal Robot" has introduced this term for the first time in Czechoslovakia in 1920. Robota means a forced labourer, and it has been used during the fiction to describe an artificial human-like being who has been built as an inexpensive worker. The fiction behind inventing the word robot has drawn an image that is still perceived in people's minds until now. This image is a human-like being that is following the human orders to perform a variety of tasks such as walking, talking, and objects manipulation. Although, this image is merely describing one specific type of robots which is the humanoid.

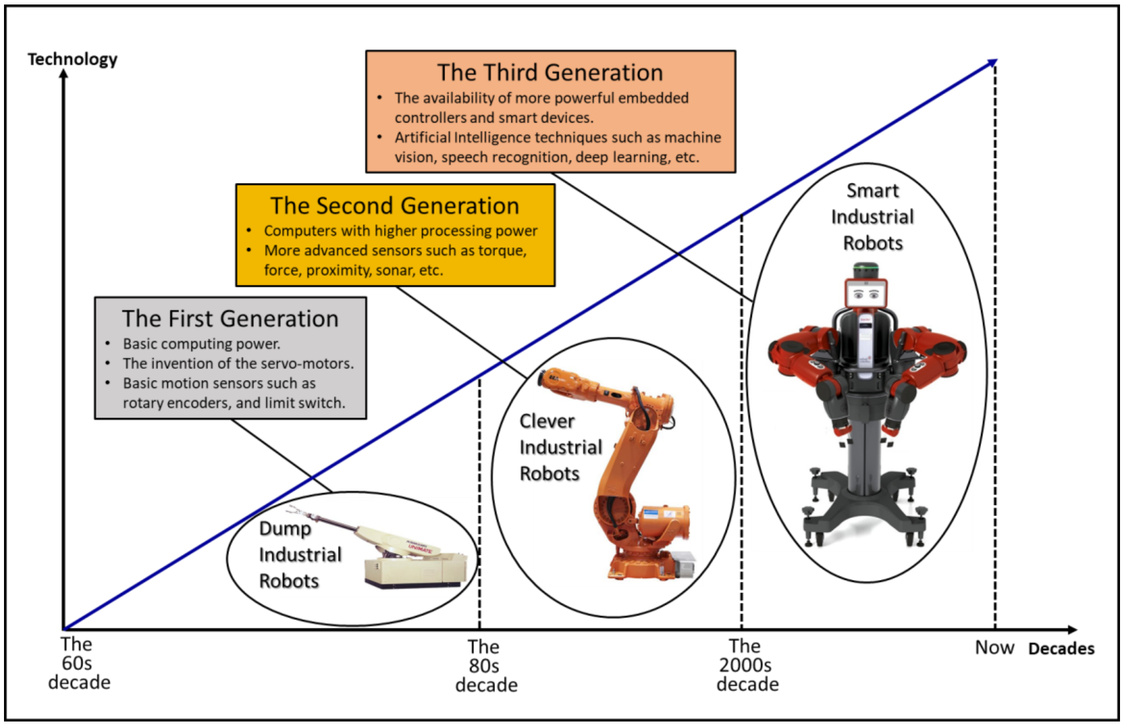

Technology
The Third Generation
• The availability of more powerful embedded controllers and smart devices.
• Artificial Intelligence techniques such as machine vision, speech recognition, deep learning, etc.
The Second Generation
• Computers with higher processing power
• More advanced sensors such as torque, force, proximity, sonar, etc.
The First Generation
• Basic computing power.
• The invention of the servo-motors.
• Basic motion sensors such as rotary encoders, and limit switch.
Smart Industrial Robots
Clever Industrial Robots
Dump Industrial Robots
The 60s decade
The 80s decade
The 2000s decade
Now
Decades

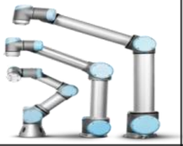
Figure 2.5: Industrial robots evolution.

A more generic definition of the robot can be found in the dictionary as a machine that is capable of automatically carrying out a complex series of actions, especially one programmable by a computer. An IR extends the generic definition of the robot by specifying its functionality. An IR is an automatically controlled, reprogrammable, multipurpose manipulator with at least three **D**egree **o**f **F**reedom (**DoF**). An IR is either fixed or autonomous manipulator that is used in material and parts handling, or in additive manufacturing (e.g., assembly, welding, gluing, painting, etc.) or in subtractive manufacturing (e.g., milling, cutting, grinding, polishing, etc.) [SSK13].

The history of IR is much related to the creation of the robotics concept itself [Nof99, Kut07]. In 1952, G. Devol invented the first programmable IR that was called Unimate. Ten years later, Unimate IR has been operated in General Motors factory to be considered the first IR generation as shown in Figure 2.5. The first generation of the IRs was an extension of the present technology at that time. Therefore, they were simple mechanical manipulators which are capable of performing precise motion profiles with high speed, according to a fixed sequence of events. Nevertheless, they were dump IRs. This is because they were very restricted to a specific sequence regardless the changes in the work environment, thus they could not alter their actions. The second IR generation showed better signs of intelligence, due to the progress in the sensory technologies and the computing power by the beginning of the eightieth. This generation is the clever robots, as they could determine the status of the outside world based on the feedback loops of their control system. Therefore, they could adjust their operation parameters such as velocity, acceleration, torque, and force.

During the nineties, an intense research focused on the safety in industrial robotics, which led to new control techniques that allowed a collision free cooperation between the IR and the worker. The progress in the safety techniques together with the advances in the **A**rtificial **I**ntelligence (**AI**) reinforced the appearance of the third IR generation by the beginning of the current century, the third IR generation are commonly known as cobots. Examples of these cobots are KUKA lightweight [KUK16], Rethink Baxter [Ret16], YuMi ABB [ABB16], Universal Robots [Uni16], and others [Fra17, Tmr16 , and Fan16], as shown in Table 2.2. These cobots are not only safe to cooperate with the worker, but also they afford different aspects of smartness. The cobot can be programmed by a demonstrative teaching, which means that a worker without any programming knowledge guides the cobot arm in a specific path with a specific speed, then the cobot repeats the same motion profile. Moreover, most of the cobots are equipped with many sensors and cameras. Those sensors can be flexibly coded based on the manufacturing scenario. For instance, a cobot may detect different products via a customized image processing algorithm. Additionally, the cobot operating system is highly developed to apply deep learning methods in case of sophisticated tasks.

Table 2.2: Examples of the most well-known cobots.

Cooperative Robot	Specifications	Applications
UR3, UR5, UR10 (Universal Robots, Denmark) 	• Degree of Freedom: 6 DOF • Payload: 3 Kg, 5 Kg, 10 Kg • Weight: 11 Kg, 18.5 Kg, 29 Kg • Reachability: 0.5 m, 0.85 m, 1.3 m	• Material handling • Machine tending • Quality inspection • Machining and polishing
KUKA LBR7, KUKA LBR14 (KUKA Robotics, Germany)	• Degree of Freedom: 7 DOF • Payload: 7 Kg, 14 Kg • Weight: 24 Kg, 30 Kg • Reachability: 0.8 m, 0.85 m	• Parts assembly • Screw driving • Machining and coating
Baxter, Sawyer (Rethink Robotics ,USA)	• Degree of Freedom: 7 DOF Dual-arm • Payload: 2.5 Kg per arm, 4 Kg • Weight: 75 Kg, 19 Kg • Reachability: 1.0 m, 1.3 m	• Pick and place • Machine tending • Packaging • Material handling • Quality inspection
ABB-YUMI (ABB Robotics, Switzerland)	• Degree of Freedom: 7 DOF Dual-arm Payload: 0.5 Kg per arm • Weight: 38 Kg • Reachability: 0.5 m	• Electronics assembly • Small goods packaging
Franka-Panda (Franka-Emka, Germany)	• Degree of Freedom: 7 DOF • Payload: 3 Kg • Weight: 19 Kg • Reachability: 0.9 m	• Electronics assembly • Material handling • Quality inspection • packaging
TM5-700, TM5-900 (Quanta inc., Taiwan)	• Degree of Freedom: 6 DOF • Payload: 6 Kg, 4 Kg • Weight: 22 Kg, 22.2 Kg • Reachability: 0.7 m, 0.9 m	• Electronics assembly • Material handling • Quality inspection • packaging
Fanuc CR4, CR7, CR35 (Fanuc, Japan)	• Degree of Freedom: 6 DOF • Payload: 6 Kg, 7 Kg, 35 Kg • Weight: 48 Kg, 53 Kg, 990 Kg • Reachability: 5.5 m, 0.7 m, 1.8 m	• Automotive machining and assembly • Machine tending and part inspection, • Packaging and palletizing, Dispensing

However, the full potential of the cobot is not realized yet for two main reasons. First, the cooperative robotics technology is still in progress, which means that the current cobots are not mature enough in terms of speed, accuracy and weight lifting comparing to the second generation who is fully matured. This immaturity limited the current cobots to some light applications such as material handling, and machine tending. Nevertheless, these limitations are rapidly vanishing. For example, Fanuc Corp. has announced lately its cobot with a 35 kg payload, which is a breakthrough that will evolve the cobots applications. The second reason is the transient delay that usually happens during an upgrade from one technology to another. This means that the industrial firms are willing to change their shop floor from a conventional to a cooperative structure, after the uncertainty factor in this new approach is already known. Large enterprises such as BMW and Audi have already started to use cobots in their production lines [MIT14, Gre16], but it will take time until majority of the other companies do the same. Even though, the growth in the cobots' sales is very promising, as shown in Figure 2.6. Due to the study that is conducted by ABI Research in 2015, it is estimated that the cobot market will exceed $1 billion by 2020 [ria17].

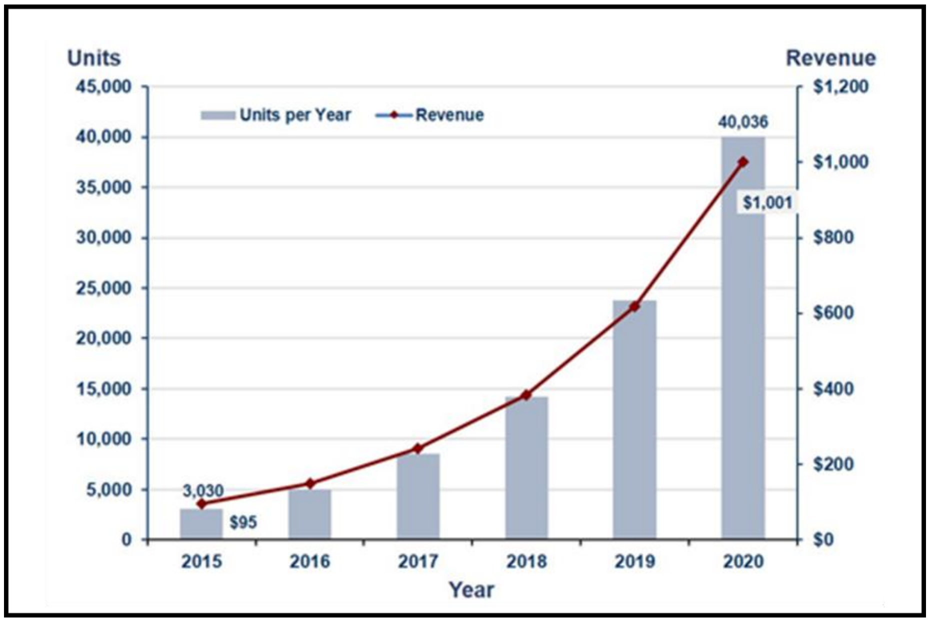


Figure 2.6: Cobots sales by 2020 due to ABI research [ria17].

The cobots' sales predication in Figure 2.6 represents another important motive for this dissertation. Considering the high increase in the cobots' sales, an appropriate mean to integrate these cobots with the workers in the workcell and hence in the whole enterprise affects the real value of these cobots. In other words, in absence of a conceptual approach to integrate the cobot and the worker from the information point of view, the cobot will lose its advantage of being cooperative, and it will turn to be just a safe robot.

2.1.3 Worker Importance in Manufacturing and SMEs

There is a common misconception among the public towards the actual potential of the automation in general and the robotics in particular. The misconception results from the presumption that all the jobs and occupation can be completely automated. However, the latest statistics study from the US bureau of labor and McKsiney Global Institute reflects a more accurate view [McK17a]. According to the study that is shown Figure 2.7a, 18% of the consumed time to accomplish a job is wasted in predicated activities. Those predicated activities can be automated with a high success rate of 78%. Moreover, 12% of the consumed time is wasted in unpredicted activities. Those unpredicted activities can be automated with a low success rate of 25%. In manufacturing, the success rate to automate the predicting actives is even decreasing to 60% as shown in Figure 2.7b. This simply can be interpreted to the following, in manufacturing there are predicated tasks that are efficient to be automated, simultaneously, there are unpredicted tasks that are efficient to be performed manually. The reasons that the unpredicted manufacturing tasks are more efficient to be done by the worker are discussed as follows:

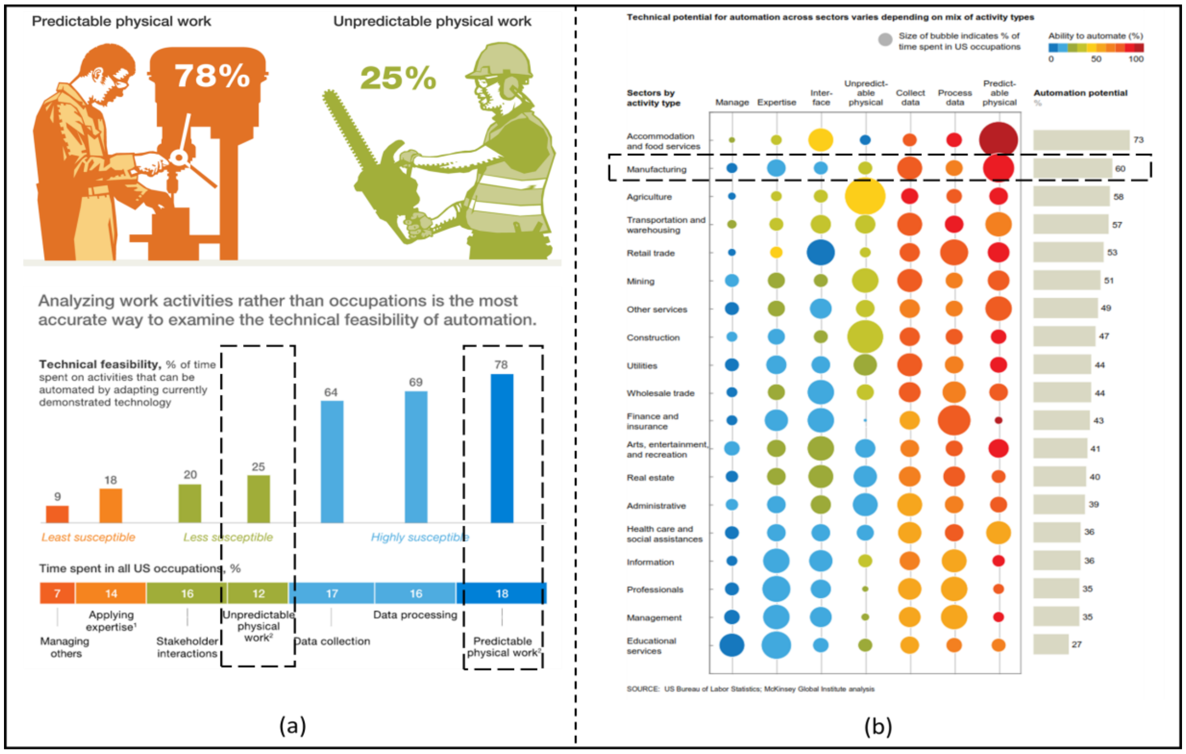


Figure 2.7: (a) Technical feasibility of automating various jobs in USA in 2017– (b) Technical feasibility of automation in manufacturing in USA in 2017 (US bureau of labor statistics & McKinsey global institute analysis) - source [McK16].

- The ability of the human to dynamically adapt to the high level of uncertainty in the manufacturing. As been mentioned above, the nature of the tasks could be unpredicted, which from the technical point of view makes the robot unable to perform these tasks alone. For instance, in an aircraft assembly scenario as shown in Figure 2.8a, the nature of the environment is chaotic, which is the main reason that most of the tasks are done manually. Building the interiors inside an airplane vessel is more time, effort, and cost efficient to be done with workers than robots. In this context, the robot can work in its own, or fully automated. Using the robot in a cooperative shop floor scenario as a smart tool will be without a doubt the most efficient method.

- The ability of the human to use his natural senses intuitively to form very complicated yet instant solutions. For example, in electronic devices assembly, some connections or parts could be very tiny and there is no standard way to assemble them, as it shown in Figure 2.8b. Performing those tasks by the robot are extremely difficult or not possible.
- The accumulated experience of the human worker and the need of the human logical judgement. For this reason, many of the electronic devices manufacturing are still using the worker for quality inspection. As an example, Figure 2.8c shows the sound quality checking operation that is often done by the worker.

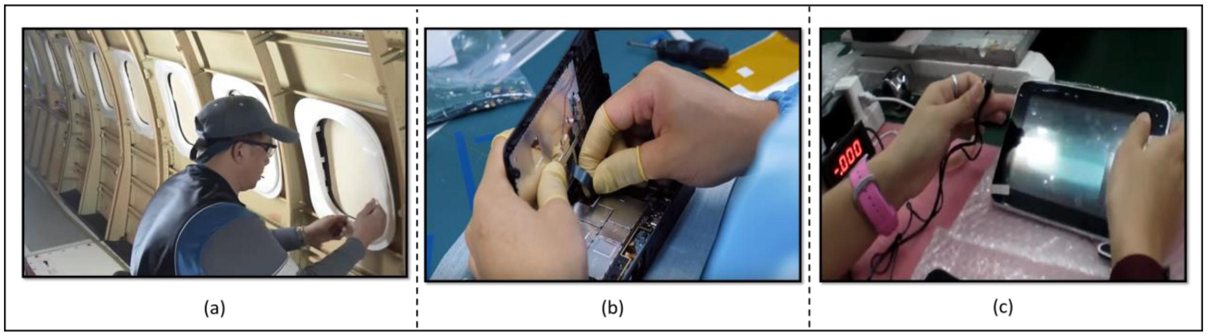


Figure 2.8: Examples of worker importance in manufacturing (a) Worker in airplane assembly - source [You17a] – (b) Worker in electronic devices assembly - source [You16a] – (c) Worker in electronic devices quality inspection - source [You17b].

Therefore, the worker is still a better choice in many cases than the robot due to many technical reasons that can be summarized as:

- **The nature of the task:** if the uncertainty level is very high in a manufacturing task, this means that it is better to be done by the worker.
- **The cost of automation:** some tasks are visible to be automated from the technical point of view. Nevertheless, the cost of the automation would be very expensive. As those tasks can be easily done by a human, yet very hard to program them for a robot.
- **The limitations of technology and Artificial Intelligent (AI):** some tasks still cannot be done by the robot, due to the limitations of the current technology and the AI.

Figure 2.7 also shows that there is a huge amount of wasted time in the data collection and processing activities. This amount reaches to 33% in total during the job execution. This wasted time can be automated with a high success rate that is 67% in average. In cooperative manufacturing, the manufacturing control system can automatically collect and process the data, which will save huge amount of time consummation during the manufacturing. This also means that the worker is more focused on one task, which will not only shorten the task time but will increase the quality of the work.

Table 2.3: Number of enterprises and workers in EU28 in 2015 (source: Eurostat, National Statistical Offices and DIW Econ) [Ec16, MDJ+16].

	Micro < 10 worker	Small < 50 worker	Medium < 250 worker	SMEs	Large > 250 worker	Total
Number of Enterprises						
Number	21,356,252	1,378,7022	224,647	**22,959,600**	44,458	23,004,059
Percentage	92.8%	6.0%	1.0%	**99.8%**	0.2%	100%
Number of Workers						
Number	40,057,408	27,503,428	23,170,352	**90,731,192**	45,168,732	135,899,904
Percentage	29.5%	20.2%	17.0%	**66.8%**	33.2%	100%

From the economical perspective, the worker power still represents an important influence over the country economy, especially if we consider the Micro, **S**mall and **M**edium-Sized **E**nterprises (**SME**s). According to the annual report on European SMEs 2015/2016, there are more than 20 million SMEs that represent the greatest sector either in the number of employees or in the value of the market [MDJ+16]. The comparison between the SMEs, Micro, and large enterprises in 2015/2016 can be seen via Table 2.3. In manufacturing, the number of workers in SMEs represents 20% of the overall number of employment, as shown in Figure 2.9. This number is more than 18 million workers who added 44% of the total manufacturing value in Europe, which is a great economic inertia that must be always considered. Taking Germany as an example for the world's leading manufacturing supplier. 22 of Germany's top 100 SMEs are machinery and plant manufacturers, with three of them featuring in the top ten. Machinery and planet manufacturing is ranked as one of the Germany's main exports alongside cars and chemicals [aca14]. In Asia, the South Korean government has raised 57.5 billion USD investment to fund 1240 Korean smart SMEs in 2016. This decision has been taken as the result that those SMEs have showed 29.2% decrease in the production cost, 7.1% time reduction, and 27.6% decrease in overall production defection [Bus16].

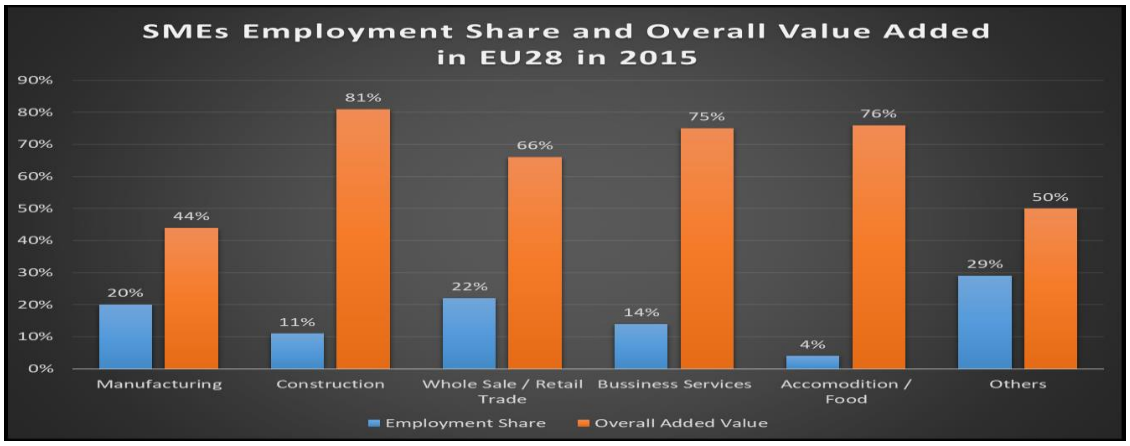


Figure 2.9: SMEs employment share and value added in EU28 in 2015 (Source: Eurostat, National Statistical Offices and DIW Econ) [Ec16, MDJ+16].

The cooperative manufacturing is a potential solution for the SMEs, for two reasons:

1- The nature of the SMEs market is very different from the large enterprises. The SMEs market is completely depending on the customization with low production volume. Furthermore, the SMEs need to minimize their manufacturing cost and time in order to survive against the large enterprises. Simultaneously, they cannot afford a sensible defection rate. In other words, the competence of the SMEs is defined by delivering whatever the customer needs as fast and efficient as possible.

2- The financial and operational capabilities of SMEs are low in comparison with large enterprises, thus SMEs cannot afford fully automated production lines. Because they do not fit their production strategy as the automation cost is very high comparing to the production profit. Therefore, the worker plays a very important role in this scenario, as he is a very valuable resource of flexibility in production with very low cost. Simultaneously, the cobots are also very important resource in SMEs. Because it is much easier and efficient to perform many small easy tasks such as material handling, packaging, or machine tending by a cobot. Yet, the workers can perform complex tasks such as electronic device assembling.

2.2 *Holonic Control Architecture*

2.2.1 Manufacturing System Control Evolution

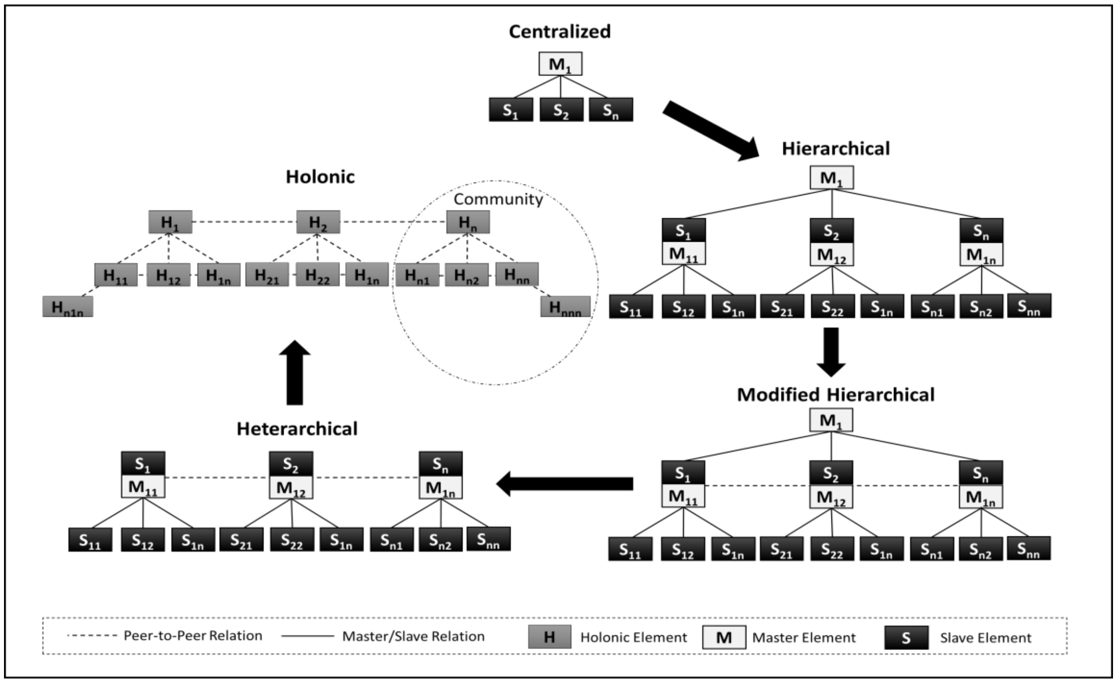


Figure 2.10: Manufacturing control system styles – adapted from [DBW91, Chr94].

Perhaps the best approach to understand the meaning of the holon is to go through the evolution of the manufacturing system control. Flexibility of the control system is the key purpose in its evolving. The evolution of manufacturing system control can be resembled by the evolution of the country ruling system. Starting from a central authoritarian monarchy, where one individual (i.e., the king) makes the decisions. To a federal parliamentary republic, where local decisions are made by every individual in each federal (i.e., the people), then a global decision is taken upon all the local decisions. The success of one ruling system style depends on fitting the ruling style with the nature of the country. An analogy in manufacturing can be applied if we see the manufacturing system as the country and the control solution is the ruling system. With the same chronological order of the ruling system evolution, the manufacturing control system has been evolved as well. Starting from a central control structure to a fully autonomous structure. This evolution can be seen via Figure 2.10 and is discussed in details as follows:

1- **Centralized control architecture:** the oldest known control structure, where two levels of control can exist. On the top level, there is a single master central controller. On the bottom level, there are many slave controllers. All the slave controllers are monitored and controlled via the central controller. The relation between the central control element and the controlled element is a master/slave relation, where all the slave elements are commanded by the master central element and they report their feedback. Thus, the central master control takes the final decision that is translated into commands to the slave controllers. Centralized control has a simple structure and enables global information sharing. Nevertheless, it has a rigid structure where a single master controller has to process all the information. Moreover, having a single master controller reduces the system reliability. Because if the central controller fails, the whole control systems fails with it. Furthermore, centralized control restrains the system customizability, flexibility, scalability, and extensibility. Thus, the system can be extended only in the horizontal direction of the bottom layer.

2- **Hierarchical control architecture:** is a modification of the central control, the purpose of this modification is to add more flexibility to the central control solution. This has been done by adding an intermediate control level that acts as master and slave at the same time. Therefore, Hierarchical control style can be seen as a top-bottom pyramid hierarchy, where master-slave relationships between the control elements are predefined and restricted, and a bottom-top feedback mechanism is followed. The decision-making process occurs according to a descending hierarchy (i.e., top to bottom). The information perception process occurs according to an ascending hierarchy (i.e., bottom to top). This control style provides responsive and more efficient performance than the centralized control. However, the slave controllers at the bottom of the hierarchy are helpless without their master instructions. Thus, it is hard to achieve the fault-tolerance. This affects the robustness and the agility of the control system negatively. Furthermore, the extensibility of the control system can be only achieved in the bottom and the intermediate levels.

3- **Modified hierarchical control architecture:** is an enhancement of the prior control style by defining **P**eer-**to**-**P**eer (**P2P**) relations among the intermediate control elements. P2P relation means an equality in the hierarchal position between the control elements. P2P relation affords the cooperation concept (i.e., win-win relation) instead of the command concept in a master/slave relation. The cooperation among the intermediate control elements is followed to obtain local decisions. However, the main system commands are originated from the main master controller. Modified hierarchical control improves the fault-tolerance and diagnosis, synchronization among the control elements, the extensibility, and the reliability. Nevertheless, the system still inherits the cons of the hierarchical control.

4- **Heterarchical control architecture:** is quasi-autonomous style, where the master control element of the central and hierarchical style is eliminated. Therefore, the decision-making process is done autonomously at the top level of the heterarchical control architecture. Thus, the control elements at the bottom layer are commanded based on the decisions that are made at the upper layer. Heterarchical control affords a moderate level of flexibility. However, it does not achieve a full system scalability or extendibility. This is because the expansion of the system can be done at the two control layers in the horizontal direction. Nevertheless, the extension of the system in the vertical direction is still impossible. In addition, it shows a great level of complexity to implement the control solution, as the system designer is required to think in a semi-autonomous approach to solve the control problem.

5- **Holonic control architecture:** the idea of a control element which acts as a master and slave at the same time is the main inspiration of the holon concept. A holon is a self-controlled and self-contained element, which forms mutually accepted control plans based on the cooperation with the other holons. A group of holons with the same type is called a community. A collective decision-making is always taken upon a negotiation procedure that happens via direct or indirect interaction among the holon communities. This collective decision is based on the responsibilities and functions that are distributed among the holon communities. The information perception is obtained via all the control holons, this information is later used in the collective decision-making process. This control style is very agile and robust [Lei04]. However, it can be more unpredictable compare to the other control styles due to the negotiation process.

2.2.2 Holonic Control Reference Model

The nature of the cooperative manufacturing is distributed with high level of manufacturing disturbance, which promotes an autonomous control solution to adapt with this nature. Due to the research in [CC07], three main autonomous control solutions are commonly applied to the smart manufacturing. Those solutions are Fractal, Bionic, and Holonic. The three systems provide a set of concepts and terminologies to design an adaptive workcell. Some of those concepts are self-origination, self-similarity, and self-optimization. Those concepts vary from one solution to another in their names but not in their meaning. However, the HCA is the most well-known approach in research. In the late of the sixties, the term holon has been introduced for the first time by philosopher Koestler [Koe67]. Koestler developed the term as a basic unit in his explanation of the evolution of the biological and social structures. Based on his observations that organisms (e.g., biological cells) are autonomous self-reliance units, which have a certain degree of independent control of their actions, yet they still subject to higher level of control instructions. His conclusion was that any organism is a whole "holos" and a part "on" in the same time, which derived the term holon [BB10]. The holon concept has been adopted in the early of the nineties by the **I**ntelligent **M**anufacturing **S**ystems (**IMS**) consortium, to define a new paradigm for the factory of the future. The following terminologies are defined by the IMS to provide a better understanding of the HCA:

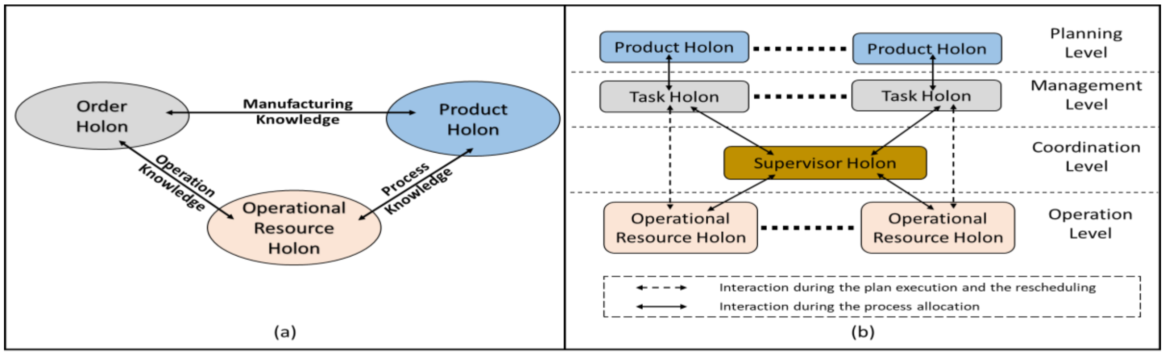


Figure 2.11: HCA reference models (a) PROSA [BWV+98] – (b) ADACOR [Lei04].

- **Holon:** an autonomous cooperative software component that represents the building block of the manufacturing control system. It is used to transform, transport, store and/or validate the information and the physical signals [BC06].
- **Autonomy**: the capability of the holon to create and control the execution of its own plans and/or strategies.
- **Cooperation**: a process whereby a set of holons develop mutually acceptable plans and execute these plans together.

- **Holarchy**: a system of holons that cooperates to achieve a global goal or objective. The holarchy states the basic rules of cooperation among the holons, and thereby defines their autonomy.

The HCA answers many questions that are proposed in autonomous manufacturing [Mec15]. The HCA is a distributed control and communication topology that divides the manufacturing tasks and responsibilities over different holon categories. Two well-known reference models define the tasks and responsibilities of the holons in a manufacturing system. The first reference model which is the oldest and the most commonly used is called the **P**roduct-**R**esource-**O**rder-**S**taff-**A**rchitecture (**PROSA**) [BWV+98]. PROSA model defines three basic holon types as shown in Figure 2.11a and discussed as follows:

1- Operational Resource Holon (ORH): is a physical entity within the manufacturing system, which can represent a robot, a machine, a conveyor belt, etc. The ORH offers the production capacity and functionality to the other holons.

2- Order Holon (OH): manages the production orders by assigning the manufacturing tasks to the present ORHs and monitors the execution status of those tasks.

3- Product Holon (PH): processes, stores and updates the different production plans required to insure the correct manufacturing of a certain product.

Besides PROSA three basic holon types, another holon type may exist when it is needed. This holon is called the staff holon. The staff holon operates as an expert holon who provides advices to the other holons if required. However, the other holons are free to refuse those advices, hence they are completely responsible for their final decision. The second reference model that can be seen as a modification of the prior model is the **ADA**ptive-holonic **CO**ntrol-a**R**chitecture (**ADACOR**) [Lei04]. ADACOR model focuses on adapting the disturbance of the manufacturing system. Thus, it implements the three PROSA basic holons except it renames the OH to the task holon as shown in Figure 2.11b. However, all the roles of the holons are still the same. The clear difference between PROSA and ADACOR architectures is that the second emphasises the role of a new holon which is called the **S**upervisor **H**olon (**SH**). The SH extends the function of the staff holon by providing coordination services when it is needed to cooperate outside the boundaries of the workcell [LR08]. Also, the SH is responsible for managing the evolution of the other holons in the model according to the environment context.

During the dissertation, PROSA reference model will be followed as long the concept is within a workcell, where only the basic holons are needed. Nevertheless, in case of the cooperative enterprise, where many cooperative workcells need to be coordinated, an SH will be added to the solution.

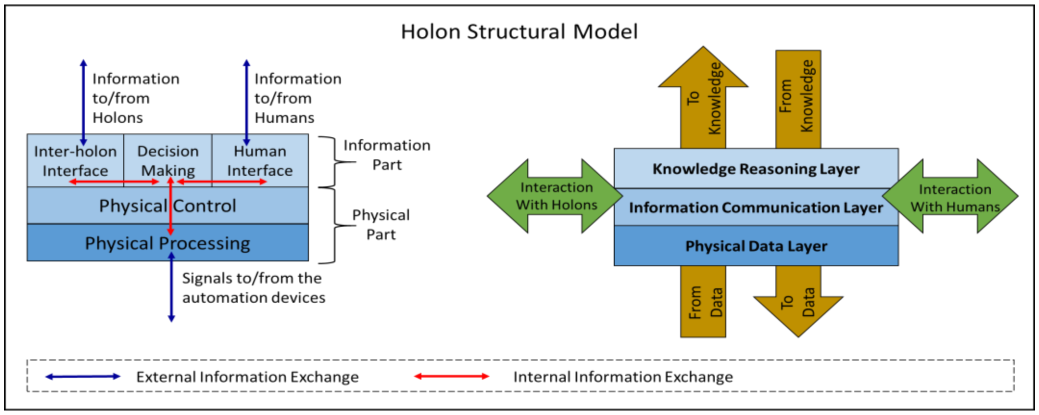


Figure 2.12: Holon structural model – a modern perspective based on [Chr94, Bus98].

The holon structural model which is shown at the left side of Figure 2.12, has been originally introduced by J. Christensen at [Chr94] and illustrated by S. Bussmann at [Bus98]. This model is composed of two components that are physical and information. The physical component is linked directly to the automation devices. The holon's physical component is responsible for conducting the physical signals from the automation devices to the holon and vice-versa. The second component is the kernel of the holon as it is responsible for processing the incoming signals from/to the physical component, to fulfil the decision-making process. Accordingly, the holon interacts with the humans and the other holons to take control of its actions based on this interaction. A modern perspective of the holon model has been followed during this dissertation, as shown at the right side of Figure 2.12. The modern perspective explicitly distinguishes between three different layers of the holon, which are as follows:

1- **Physical data layer:** physical data and events are the low-level signals that represent the manufacturing control I/O. Thus, the main responsibility of the holon's physical layer is to link the physical I/Os to the HCA. For example, in the context of cooperative manufacturing, when one of the controller outputs equals to five volts, this means a busy status of this operation resources. This layer will be studied in deep details in section 4.1.1.

2- **Information communication layer:** data are meaningless as long as they are not processed and placed in the right context. For example, cobot-1 is busy, is a clear information regarding cobot-1 status. However, the real value of the information is to exchange and share it. Therefore, the second layer in the holon structure is mainly responsible for interacting and communicating with the other holons and humans, to share and acquire new information. Furthermore, the information communication must guarantee the interoperability of the software component in case of integrating the HCA with another manufacturing control solution. This layer will be studied in deep details in section 4.1.2.

3- **Knowledge reasoning layer:** knowledge is a structured information. For example, cobot-1 is busy and it handles product-1 to worker-1. Knowledge construction requires to reason and manage the information, which is the main responsibility of the third layer in the modified holon model, which will be studied in details in sections 4.1.3 and 4.2.

In contrast with the PROSA and ADACOR models, which use very generic holon model, the dissertation provides a clear and a modern perspective of the holon. This modern model will be used to design an implementation model of the holon in chapter 4. In section 5.2, the holon implementation model will be converted to object-oriented models, which represent the different holon categories in the context of cooperative manufacturing. Finally, these models will be used to implement the HCA in chapter 6.

2.3 *Conclusion*

The conclusion of this chapter can be summarized as follows:

- Cooperative manufacturing is one of the most important pillars that supports Industry 4.0 goals. Due to the novelty of the subject, there is no focused research on this subject yet, which plays a great motive for this dissertation.
- The research in industrial robotics safety is rich enough to afford a new generation of safe and cooperative IRs. Those cobots have achieved high acceptance in the industrial market. However, the cobot value is less than what it is worth if it is not integrated within the cooperative workcell. In other words, *the cobots are designed to be safe but they cannot be cooperative without the appropriate manufacturing control architecture*.
- The worker is an important asset in nowadays manufacturing. From the technical point of view, *the worker is an affordable resource of flexibility and adaptability, and hence the worker can compensate the shortages in the present technologies and AI*. From the economical point of view, *the total replacement of the workers by robots is not a feasible solution. Because the workers represent a great economical inertia especially for SMEs*. The total replacement of the workers would have a negative social and economic impact. Therefore, combining the worker and the robot together in the same manufacturing paradigm becomes a necessity to cope with the current industrial challenges.
- The evolution of the manufacturing system control shows that the HCA is the most appropriate solution to integrate the worker and the robot together in one manufacturing workcell. This is due to the correspondence between the autonomous characteristics of the solution and the distributed nature of the cooperative workcell.
- The HCA reference models provide a very generic description of the holon structure as a software component, and functions as a manufacturing control solution. The mission of this dissertation is to develop these models in the context of cooperative manufacturing.

Chapter 3 - Literature Review

"The Principle Mark of Genius is not Perfection but Originality, the Opening of new Frontiers."

Arthur Koestler

3.1 Overview

This chapter investigates two different subjects of research. The first subject is the cooperative manufacturing and the second subject is the holonic control. The goal of the chapter is to study the latest research progress and the open issues in both subjects. Therefore, the conclusion of the investigation will be considered during the design and the implementation phases of this dissertation.

Three related topics will be investigated in cooperative manufacturing, which are the safety in cooperative robotics, the cooperative factory of the future, and the cooperative manufacturing control solution. The literature review investigates the advances in worker safety that already have been achieved in prior research. Then, it moves towards the idea of cooperative manufacturing as a solution for the factory of the future. The opinion of the workers as a part of the cooperative manufacturing will be considered, to understand the challenge in this subject from the point of view of an experienced worker. Furthermore, other related studies will locate the real potential of the cooperative manufacturing in practice. The implementation of a cooperative workcell is the third topic to investigate. Thus, a study of the proposed cooperative manufacturing control solution in research will be conducted, to understand the existing conceptual approaches and the technologies that are used to achieve these approaches.

As the cooperative manufacturing is a special case of the RMS and the FMS, the second half of the literature review investigates the HCA. This is because the HCA is the most common solution for the reconfigurable and the flexible workcells. Different case studies that are similar to the cooperative workcell will be studied. Therefore, the benefits and the drawbacks of their solution will be considered during the design of the dissertation concept.

3.2 *Cooperative Manufacturing*

3.2.1 Safety in Cooperative Robotics

The safety in cooperative robotics is addressed at this part of the literature review, as safety is the first step towards the cooperative manufacturing. Safety in industrial robotics is regulated by a variety of standards. One of the most known safety standard is ISO 10218 that entitled as "robots and robotic devices – safety requirements for industrial robots". ISO 10218 outlines the potential methods of safety in cooperative manipulation such as speed limits, power and force limits, and separation monitoring. Those methods are discussed in details in Table 3.1. Another standard is ISO/TS 15066 which entitles as "robots and robotic devices – collaborative robots" extends ISO 10218 by specifying the technical requirements to achieve the recommendations of ISO 10218 [Mat14]. These standards are still developing, as shown in [BPS17, VWE17]. For example, [BPS17] builds new strategies upon these safety standards to cooperate with heavy-load IRs. Those strategies are reactive approaches in response to the IR actions. While, [VWE17] uses a camera-based system to define three zones of safe cooperation.

The research in [LFS17, LRS14, and MMT+15] is extensively investigating the different safety methods in cooperative manufacturing. The safety in cooperative manufacturing can be defined in terms of physical or psychological safety. On the one hand, the physical safety immunizes any unintentional or unwanted contact that might occur between the worker and the robot. Furthermore, during the physical cooperation, the forces exerted upon the worker from the robot must remain below a specific threshold, in order to prevent any possible injury. On the other hand, the psychological safety is the prevention of any psychological discomfort that might stress the worker who is in a close cooperation with the robot. The psychological discomfort might be the result of the worker's fear to be injured during the physical cooperation, or it might be caused from the violations of the social conventions and norms during the interaction.

Table 3.1: Safety Methods due to ISO 102108 and ISO/TS 15066.

Safety Method	Method Description	Applications
Safety-rated Monitoring Stop	The robot stops and remains stopped without removal of its electrical power input, while the coworker is still in the collaborative workspace. This is different from traditional robot which uses safety-rated control system to shut down the robot while the operator is in robot workspace.	• Direct part loading or unloading to end-effector. • Work-in-process inspections.
Hand Guided Operation	The robot is directly and manually controlled by the coworker.	• Worker assistance in applications which require weight lifting or accuracy.
Speed & Separation Monitoring	The robot speed controlled based on how close is the coworker.	• Simultaneous tasks.
Power & Force Limits	The robot forces and torque are controlled so no physical impact can occur. Physical contact between the robot and the coworker can occur intentionally or unintentionally during the operation.	• Small or highly variable applications. • Applications needs frequent coworker presence.

The first common techniques in physical safety are control-based. These techniques are taking advantage of the prior knowledge of the cooperative environment and tasks, to formulate cooperation models and hence react upon that. Control-based safety techniques may use pre-collision methods or post-collision methods, as shown in Figure 3.1. Pre-collision safety is a prevention method that tends to monitor the worker, the robot, or both and modify the robot control parameters before any collision. This can be done via quantitative limits, where the robot parameters such as velocity and forces are compensated to limited values during the cooperation, thus no injury might occur to the worker. Speed and separation monitoring method is another pre-collision approach that has been discussed in Table 3.1. Potential fields help to develop more complex safety behaviours by defining a risk criterion, which guides the robot motion profile based on the changes in the environment. For example, moving around the worker's head can be taken as an indication of the risk criteria.

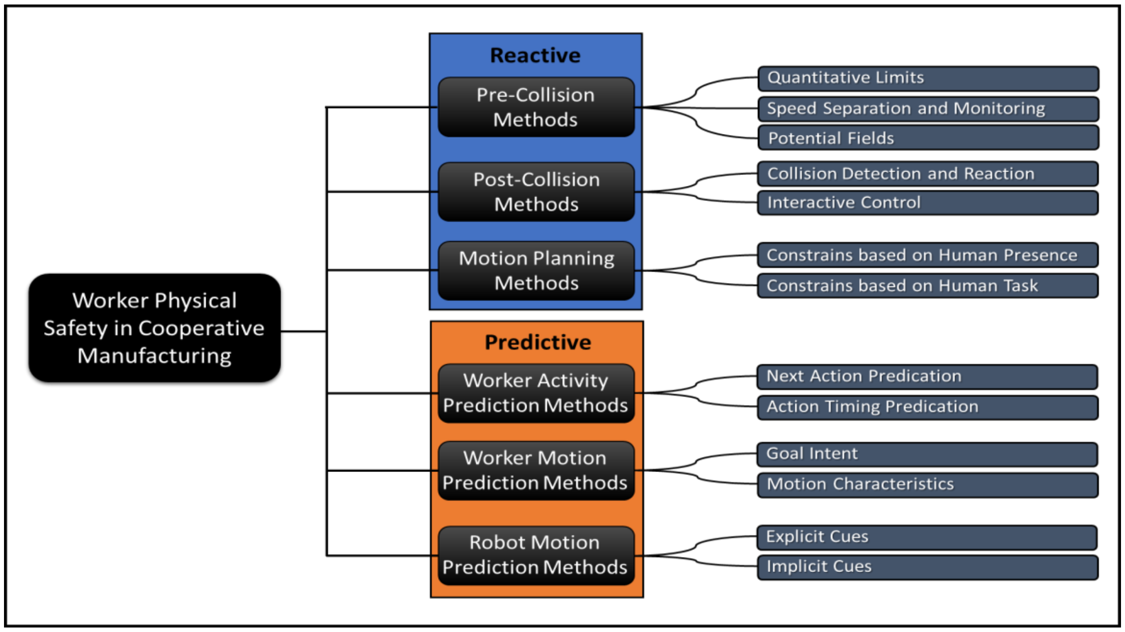

Worker Physical Safety in Cooperative Manufacturing
Reactive
Pre-Collision Methods
Post-Collision Methods
Motion Planning Methods
Predictive
Worker Activity Prediction Methods
Worker Motion Prediction Methods
Robot Motion Prediction Methods
Quantitative Limits
Speed Separation and Monitoring
Potential Fields
Collision Detection and Reaction
Interactive Control
Constrains based on Human Presence
Constrains based on Human Task
Next Action Predication
Action Timing Predication
Goal Intent
Motion Characteristics
Explicit Cues
Implicit Cues

Figure 3.1: Physical safety in cooperative manufacturing - source [LFS17].

Post-collision method is not only required after the collision has already happened, but also during the cooperative tasks, where the physical contact is a part of the process. Collision detection and reaction approach aims to discriminate between intentional or unintentional collision via different means such as machine learning, modelling of the physical collision, and using variable stiffness actuators. According to the type of collision, the control system would have different reaction strategies. Post-collision control strategies might be a function in the estimated collision force or torque. These strategies are mainly tending to either compensate the values of the robot parameters or changing the path of the robot to avoid the collision. An industrial example that follows this technique can be seen via Bosch APAS cooperative assistance system [Bos15]. Bosch APAS is a smart skin that is worn by the robot to sense any post-collision situation, and thereby compensates the robot motion profile. Another safety approach is the interactive control, where an explicit interaction between the worker and the robot takes place to distinguish between cooperative and non-cooperative mode. This can be based on gesture, speech, or GUI interaction to apply the post collision safety during the cooperative mode.

Control based safety techniques do not require complex models of the environment, which is a big advantage. However, they are purely reactive techniques that may cause inefficient cooperation. This reason urged the researchers to find other means of physical safety. For example, [LRS14] proposes a motion planner implementation that consider the worker's presence and movement while computing the robot paths. This motion planner is also capable of reasoning constrains that are based on the worker geometry to support a real-time motion re-planning. In this research, a standard non-cooperative IR (ABB-IRB-120) has been used during the implementation, as shown in Figure 3.2a. The research uses a commercial motion capture system (PhaseSpace) to recognise the worker's motion. Thus, it represents both the worker and the robot in a three dimensional (3D) virtual simulation environment (Open-Rave), as shown in Figure 3.2b. Based on the separation distance between the robot and the worker, one of three possible speed modes will be considered by the robot's controller, as shown in Figure 3.2c. This method is very similar to speed monitoring safety. Nevertheless, it considers the planning of other robot parameters such as inertia, acceleration, or jerk to promote the worker comfort during the cooperation. Neural network is another technology to implement the motion planner instead of the real-time simulation, as stated in [MOE17]. Other motion planning approaches consider the WTP to define constrains of the robot path re-planning as in [PMP+16]. In this approach, the motion planner estimates the WTP to calculate a safety zone of the robot motion. Combining task-based planning and the worker presence and geometry provides a higher task planning accuracy.

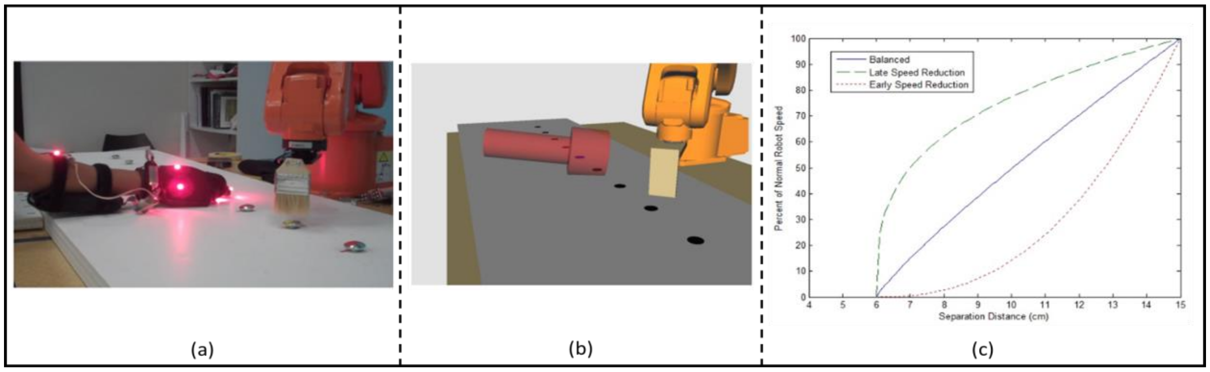


Figure 3.2: [LRS14] (a) Motion planner implementation– (b) Motion planner virtual representation – (c) Three possible modes of speed reduction as a function of the separation distance.

Motion planning safety techniques are useful in a quasi-static environment. However, in a highly dynamic environment, adding the motion prediction with the motion planning gives more reliance. The motion prediction can be based on predicting the worker activities and motion, predicting the robot motion, or both. Predicting the worker activity is done via the low-level sensor data or the goal objective. Low-level sensors data can be based on measuring the pressure, magnetic field, or flow, to understand the current worker activity and hence the next activity is estimated. Goal-oriented predication depends on image or video processing to understand the current worker activity. For example, the item that the worker is grasping might lead to estimating the next activity. The worker activity prediction can be also based on the timing, this can be done by calculating the possible time of every task and build the next prediction based on that. Nevertheless, this method of worker activity prediction can be very demanding because the WT_P is always variable.

The worker activity predication does not give information regarding the worker motion profile in his workspace. However, this is possible via the worker motion prediction. In case of a goal-oriented activity predication, the motion predictor estimates the worker path based on his location with respect to the final goal. Another method to predict the worker motion is through observing the motion characteristics of every individual to predict the next move upon that. Nevertheless, the previous method needs deep structured learning techniques along with image processing, which makes it quite complex, as stated in [LW17]. [LW17] uses Hidden Markov model with body gestures recognition to build a motion predicator. The last physical safety in cooperative manufacturing can be also achieved via the robot motion predication. Explicit and implicit cues from the worker are used to predict the next move of the robot. The explicit cues occur through a direct vocal or visual communication. While in implicit cues, an analysis must be held to understand the implicit non-verbal communication that can be detected within the worker motions. Although, the HMI is not the main theme of this dissertation, the dissertation proposes to combine the explicit and implicit interaction together to provide the worker with an adequate approach to interact with the proposed control solution during the cooperative manufacturing, this will be seen in details during the implementation phase in section 6.2.

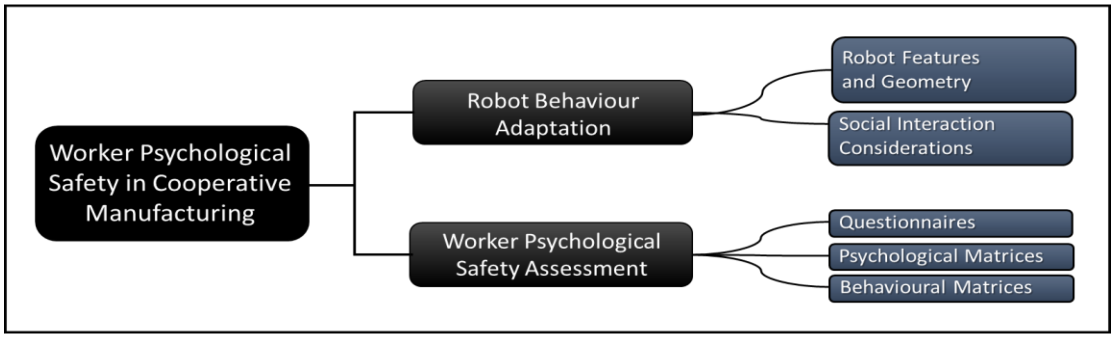

Worker Psychological Safety in Cooperative Manufacturing
Robot Behaviour Adaptation
Robot Features and Geometry
Social Interaction Considerations
Worker Psychological Safety Assessment
Questionnaires
Psychological Matrices
Behavioural Matrices

Figure 3.3: Psychological safety in cooperative manufacturing - source [LFS17].

The psychological safety in cooperative manufacturing is an important matter of research (refer to [MK16]). The absence of psychological safety would cause harm to the worker on long-term bases. The worker psychological safety can be obtained through two different approaches. The first approach is to adapt the robot's behaviour. This can be done by adjusting the robot features such as velocity, acceleration, and proximity distance to avoid any discomfort during the cooperation. Another method is to consider the social features of the robot by designing it in a way that smooths the cooperation. This aspect is obvious in some of the new social cobots such as Baxter. Even that Baxter is a dual-arm IR (refer to Table 2.2 in section 2.1.2), but it looks like a hominid robot to get the acceptance of the worker. Another social factor could be the height of the robot that is recommended to be in the same range of the human being height. Additionally, the interaction method with the robot is an important social consideration. Some methods in HMI such as body gestures are successful to achieve the interaction purpose, however they are uncomfortable and inappropriate in the context of cooperative manufacturing. The second approach to achieve the worker psychological safety is to consider the worker periodic feedback. Questionnaires and psychological and behaviour matrix tests can be used to assess the worker feedback.

3.2.2 Cooperative Factory

Inasmuch as the cooperative manufacturing is a new subject in industry, it is important to review the conducted **U**ser e**X**perience (**UX**) studies that addressed the subject. The UX studies reflect the worker's general opinion and feedback to get involved in a close cooperation with the cobot. Therefore, the real problems and challenges in this field can be screened. The UX studies in [WH16] and [WHM16] are a part of a project called AssistMe, which aims to develop an innovative haptic and optic assistance concept for the worker. The assistance concept is achieved through the cooperation with the cobot in two applications. The first is an assembly of automotive combustion engines, while the second targets the machining and polishing operations in the casting moulds. The study aimed to understand the usability and the acceptance of the close cooperation from the angle of the workers. An off-the-shelf robotic system from universal robotics has been used as a case study as shown in Figure 3.4a, a schematic of the implemented cooperative workcell can be seen in Figure 3.4b. Two cooperation scenarios have been used to conduct the UX. In scenario-a, the worker has to teach the cobot to perform a screw driving operation which is a part of the assembly process. While in scenario-b, the worker has to teach the cobot the specific positions of the polishing spots.

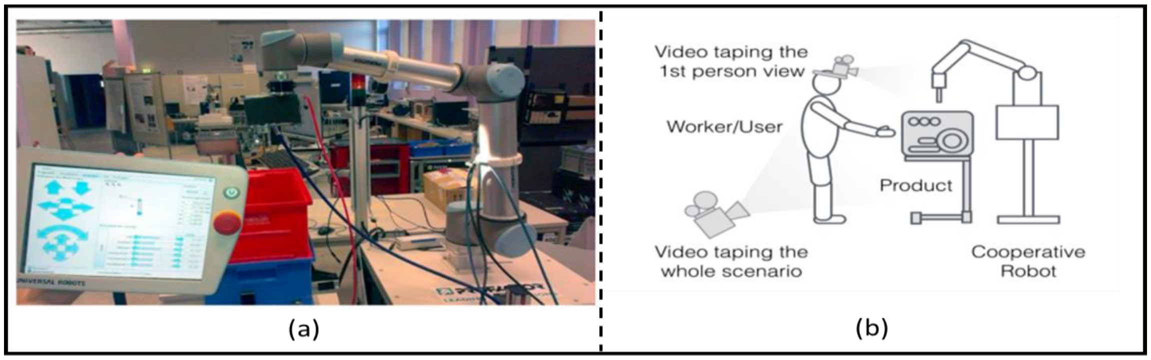

Video taping the 1st person view
Worker/User
Product
Video taping the whole scenario
Cooperative Robot
(a)
(b)

Figure 3.4: (a) An off-the-shelf use case robotic workcell from universal robotics – (b) A schematic representation of the use case set-up - source [WHM16].

The results of the study are shown in Table 3.2 and Table 3.3. The results indicate that the overall idea of cooperative system is tolerable among the workers. Apparently, the workers have high confidence that the system is trustworthy, safe from the physical aspect and saves the worker effort. However, the performance of the cooperative system is lower than their expectations. Table 3.2 and Table 3.3 support the same impression that the workers have an anxiety of being totally replaced by the cobot. This idea is a misconception of the purpose of the cobot in the manufacturing environment, and it is a result of the old trend in mass production, where machines are always seen as a replacement of the workers. Additionally, the worker cannot see the cobot as an adaptable tool even they trust in its physical safety. This opinion is a very important motive for this dissertation, because it means that the cobot cannot provide the flexibility and the adaptability unless there is a suitable control solution behind the scene that connects the cobot and the worker together. Although the UX study has been conducted only over five workers, it gives a non-biased opinion towards the implementation of the cooperative workcell over a real shop floor. The participants of this UX study were a group of experienced workers, who used to do the previously described operations manually. The answers of the questionnaire have been gathered after three working weeks with the cobot. Two of the participants have acknowledged that still after three weeks of working with the cobot, new physical and mental requirements are needed from them to get along with the cobot.

Table 3.2: Acceptance of cooperative manufacturing concept (0 = very low, 5 = very high) [WHM16].

Criterion	Use Case A (n = 5)	Use Case B (n = 5)
System Trust	3.75	2.90
Workcell Safety	4.13	3.00
Perceived System Competence	4.38	3.30
Performance Expectancy	3.00	2.40
Effort Expectancy	4.13	3.30
Total Score	79.50%	61.60%

Table 3.3: Acceptance of the cobot among the workers (0 = total disagree, 5 = total agree) [WH16].

Feedback Question	Feedback Opinion (n = 5)
The cobot will replace my job in the near future.	5
The cobot determines my work and speed.	4
The cobot should be more adaptive.	4
The cobot stops at physical contact.	4
The work with the cobot is a big transition.	3
The cobot is a helpful tool.	2
The cobot should have more anthropomorphic features	2

Another survey has been conducted by Fraunhofer institute for industrial engineering [Fra16], in order to provide a design guide to involve the lightweight cobot in manufacturing. Figure 3.5a shows the 25 applications that have been involved in this survey in the area of assembly, logistics, and part manufacturing in different sectors of industry. The result of the survey can be seen in Figure 3.5b. In terms of increasing the operational efficiency, the report recommends to consider that the higher capacity of the cobot is the key to obtain the high efficiency. This is because the cobot can be slower than the worker, if the cobot is used to assemble a product. Therefore, the justification comes by selecting the suitable task for the cobot and running it for longer operation times (e.g. more than three-shift operations). In other words, the cobot initial cost that should represent the third of all the total cost will be recovered and paid back over the long project run. In the meantime, it is safe to assume that the price of cobots are continuing to fall.

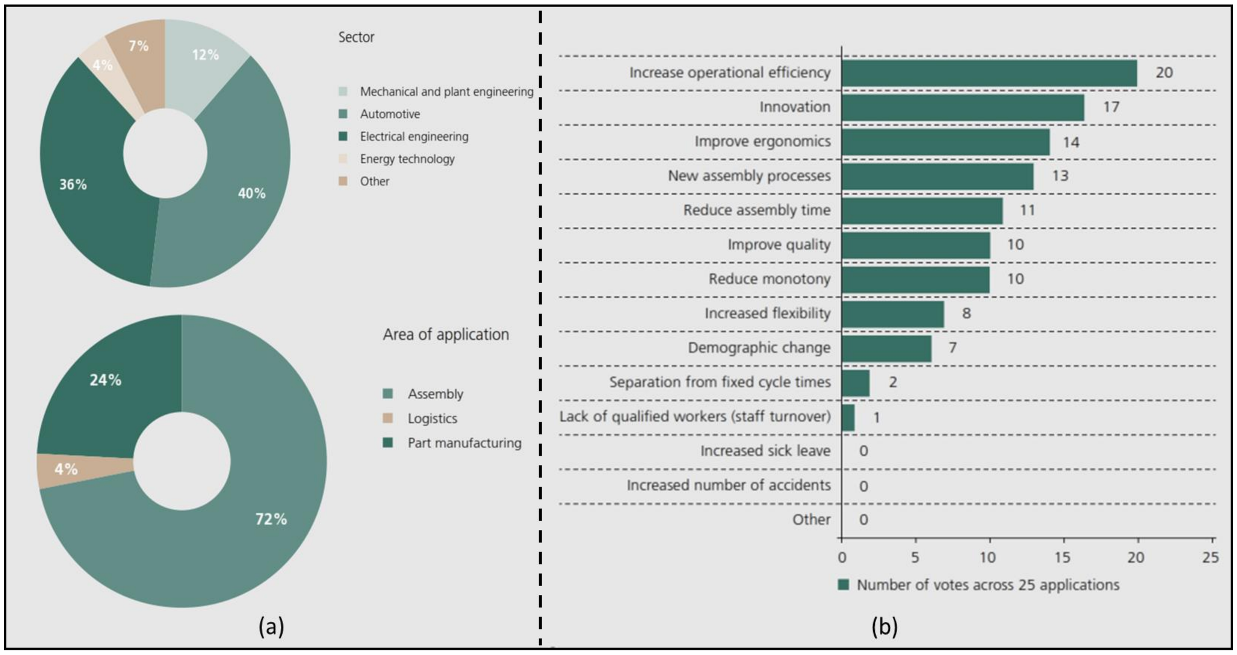


Figure 3.5: Survey to guide the immigrating from manual manufacturing to the involvement of cobot [Fra16] (a) Involved applications – (b) Survey results.

Another important result of this report is the worker ergonomics. The worker ergonomics stems from creating reasonable working conditions during his cooperation with the cobot that are as comfortable as possible. This addresses not only the physical aspects, but also involves other issues such as: the tasks content, giving the workers the leeway to define their activities, and to find the best ways to learn their tasks. Those soft factors showed a significant influence on the productivity in cooperative manufacturing. The existence of the cobot in the manufacturing context will unlock the ergonomic benefits and minimize the existing deficiencies by providing support in situations, where the workers have to perform demanding tasks such as maintaining certain postures or holding themselves in awkward positions. Also, by taking the strain from the worker when dealing with hazard environmental condition such as chemical, as well as providing the workers with the assisting power and releasing them from highly repetitive tasks.

The cooperative task planning and organization is another concern of the survey report. It is stated that the wrong planning of the cooperative tasks leads to the deterioration of production. It is also stated that there is not an available tool yet which helps the designer to check the correct cooperative task plans, and it is an open matter of research. Finally, the report discusses the aspect of accepting the cobot among the workers. Providing and sharing the information in the cooperative workcell is an important key to obtain the cobot acceptance that in turn improves the productivity. As a sum-up of this report results, there is still a big opportunity to improve the different aspects of cooperative manufacturing. Providing the flexibility in the cooperative tasks by involving the worker in the task planning process and providing the information guidance are the missing keys to accomplish the improvement.

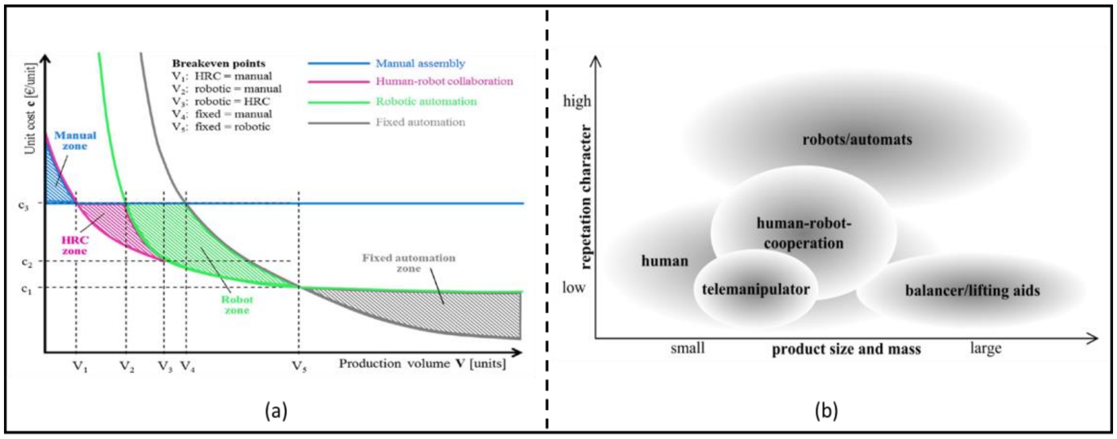


Figure 3.6: The gap in production paradigm that can be blocked by the cooperative manufacturing (a) Source [Bar15] - (a) Source [WW14].

The gap in the production paradigm has been highlighted in [Bar15]. Cooperative manufacturing has been addressed as a potential solution to block this gap. The pink coloured area in Figure 3.6a, is the suitable zone to apply the cooperative manufacturing based on the production volume and cost. The cooperative manufacturing zone is not only the interesting part in Figure 3.6a, but also the manual zone and the robot zone. The fact that the robot and the worker can exist in the same workspace, opened new possibilities of three manufacturing modes based on the production requirements. Whenever, the production volume is low, and the cost can tolerate to be high, the manual mode can be used. Otherwise, depending on the required volume and the unit price, the cooperative mode or the robot mode can be used. The same argument is the foundation of the research in [WW14], as shown in Figure 3.6b.

The research in [MK16] proposed a very interesting approach towards the social dimension of cooperative manufacturing in terms of the technical challenges. First, the research asks the most common controversial question in the field of robotics, which will the robot replace the worker. A clever answer has been presumed that without a doubt the cooperative manufacturing will change the general professional profile of the worker. In such way that the workers with low skills who are performing simple tasks will be replaced by the cobot. Nevertheless, more need of highly skilled workers would compensate this replacement. This presumption makes a lot of sense, as the recent fast growth in technology must force the evolution of the worker skills.

Furthermore, [MK16] tackles the social challenges in cooperative manufacturing from three directions. The first direction is the new methods in physical interaction with the cobot such as haptics will play an important role in developing the cooperation. As these technologies will offer better working conditions to the worker and increase the task performance. The second direction is the task complexity. It is stated that the cooperative manufacturing needs to consider new issues such as the compliance of the machine integration within the work team, the complexity of the data processing, and the new interaction and interface technologies. This will definitely require further research to develop new suitable software architectures. These architectures must be able to integrate the knowledge-based systems along with the automated and programmable machinery. Finally, the research focuses on the intuitive interfaces. At this aspect, the research addresses the need to investigate natural language and methods of programming that fit the nature of cooperative manufacturing. Software system such as **C**omputer-**A**ided **D**esign/**C**omputer-**A**ided **M**anufacturing (**CAD/CAM**) is an example of a mature manufacturing control system that has been developed after the invention of the **C**omputer **N**umerical **C**ontrol (**CNC**) machines. The CNC machine is useless without its CAD/CAM software. This is the same for the cobot, where the hardware is less than its actual value without the proper software to integrate it to the shop floor. In the conclusion, the article states the shortages in the related research, which is focused on the cooperative manufacturing control system.

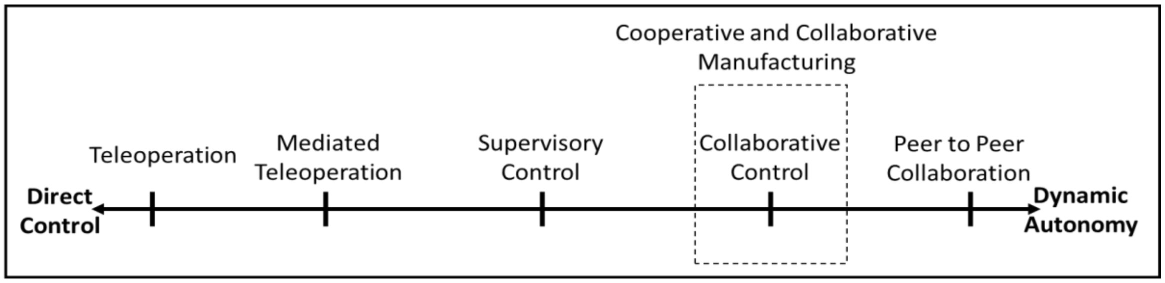


Figure 3.7: Different level of autonomy in HRC [GS07a].

The research in [GS07a] studies the subject of **H**uman-**R**obot **C**ooperation (**HRC**) from many angles. The first angle is the autonomy, which is seen as a continuum from an entirely autonomous to directly controlled by the human, as shown in Figure 3.7. These levels of autonomy describe to what extent the robot may act on its own. The research states that autonomy is not the end of the HRC, but rather a mean to support a productive cooperation. The autonomy is only useful insofar as it supports beneficial cooperation between the human and robot. The second angle that has been considered is the information exchange. In the opinion of the researcher, the information exchange can be considered the most important tool in HRC, as the amount of shared situation awareness and the common understanding grounds can be obtained via it. The third angle of the research is the system structure. The research proposed a variety of questions that by answering them the system structure can be found. Those questions can be summarized as follows: how many humans and robots will be managed by the control system, who has the authority to take a certain decision, who is issuing the instructions and the commands, who is defining the level of strategical and operational responsibilities, and whether the organizational structure is static or dynamic. The research also points that the ability of adaption and situation learning is an important attribute in cooperation, which guarantees the reliability of the long-term interaction between the human and the robot. Finally, the mutual task shaping can be obtained by the situation learning. The mutual task shaping is an essential goal in the HRC, as it is a way to perform new tasks that the human could not do alone, or to make tasks that could be done by the human alone easier and less effort demanding.

The research in [Mon13] narrows the focus of [GS07a] as it categorizes the cooperative and collaborative manufacturing as a human-cantered autonomous system that are supervised by the worker to some extent, and supported by the mean of the autonomous software agents who can provide guidance and complicated decision-making. The research emphasised the information and communication technology as the right tool to fulfil the demands of cooperative and collaborative manufacturing. Thus, the manufacturing control system must be self-steering or regulating, self-directed towards a goal, and governed by laws and strategies that direct its behaviour. Those capabilities of the proposed manufacturing control system can be achieved by three means, which are the environment perception, the information control, and the learning from the accumulative knowledge. The research in [MS10] supports the same previous ideas, as it states that flexible automation strategies must be able to support the worker and distribute the decision-making process. A decentralized organization of work with flat hierarchies and strong delegation of responsibilities, especially on the shop floor level, would be the most suitable model to integrate the worker and the cobot to the industrial enterprise. Putting into considerations the uncertainties that can be caused by the human behaviours.

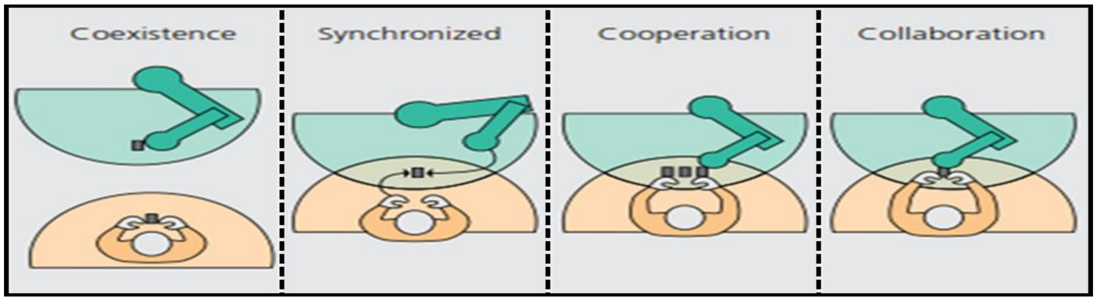


Figure 3.8: Possibilities of cooperation between the worker and the cobot – source [Fra16].

The ambiguity between the cooperative and collaborative manufacturing definitions along with other important terminologies that are always alternated in this field of research have been clarified by [Fra16]. A group of new terms have been explicitly highlighted by [Fra16], as shown in Figure 3.8 and discussed as follows:

- **Coexistence**: in this mode, there is no shared workspace between the worker and the robot. However, the robot operates without a safety cage. This could be because the robot is relatively small and cannot harm the worker in case of collision, or because it is a safe robot.
- **Synchronization**: a shared workspace does exist in this mode. However, no actual physical interaction occurs between the worker and robot.
- **Cooperation**: Both the worker and the robot are performing tasks over the same product in the same shared workspace but not simultaneously (i.e., sequential manufacturing operations).
- **Collaboration**: Both the worker and the robot are simultaneously performing tasks over the same product in the same shared workspace. (i.e., parallel manufacturing operation).

The difference between the cooperation and the collaboration in manufacturing is a very important aspect, especially when considering some problems such as the workcell planning and scheduling. The order of the manufacturing processes has a great influence on selecting the proper solution. Therefore, it is worth to emphasise that the dissertation work is mainly focused on the cooperative manufacturing.

3.2.3 Cooperative Manufacturing Control

Cooperative manufacturing research is still in its initial phase [BR17]. However, the research that explicitly addresses the cooperative manufacturing problems is focused on either the cooperative workcell safety and layout design, or the cooperative workcell implementation. The previous part of the literature survey covered the general review of cooperative manufacturing and the available safety techniques that make this idea visible. Accordingly, this part of the literature survey will review the related research that moved to the implementation part. The main goal is to check the current concepts, technologies, and techniques, which have been used during the prior research. The research in [BR17] provides a conceptual system architecture to plan the cooperative workcell that is shown in Figure 3.9a. The description and the goal of cooperation and hardware of the case study in [BR17] are not explicitly mentioned. However, due to Figure 3.9a, it is clear that KUKA lightweight robot has been used (refer to Table 2.2 - section 2.1.2). Microsoft Kinect [Mic16] and Leap Motion [Lea16] sensors are also used to recognise the worker's task as mentioned through the conclusion of the article. The article contribution is located in the conceptual system architecture that is shown in Figure 3.9b. The goal of this architecture is to represent the cooperative workcell environment, to plan the tasks between the worker and the cobot.

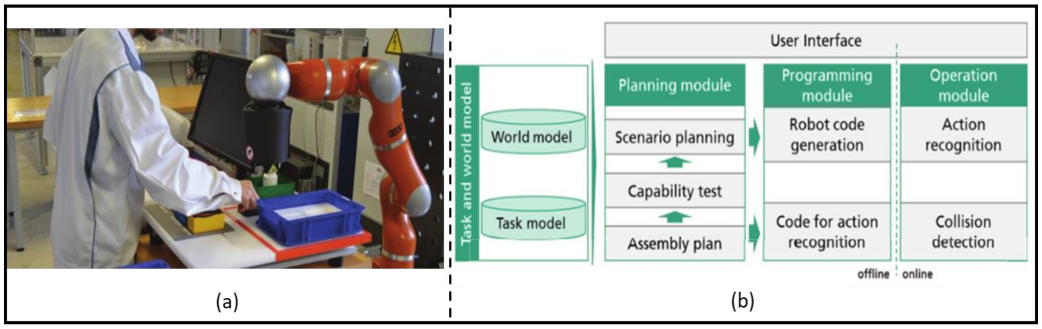


Figure 3.9: [BR17] (a) Cooperative workcell layout – (b) Control system architecture.

In [BR17], the cooperative environment is represented as CAD models that describe the workcell layout and the product structure. The planning module is generating the assembly sequence by assigning the assembly task to either the worker or the cobot, after matching their capacities with the task. Then, the task is transferred to the programming module if it is a robot task or to the operation module if it is a worker's task. The programming module is responsible for generating the robot code to move to a specific point in a specific path, and to link it with the worker's action recognition to achieve the cooperation. The operation module generates an action recognition code that fits the worker's task, and considers a collision detection strategy based on the worker's task. The research does not give any information regarding the technologies that have been used during the implementation phase. Moreover, the research does not show any sensible results and mixes between the cooperation and collaboration terms, which makes it so hard to assess the feasibility of the system architecture. Nevertheless, it reflects a general and initial overview of a proposed information architecture.

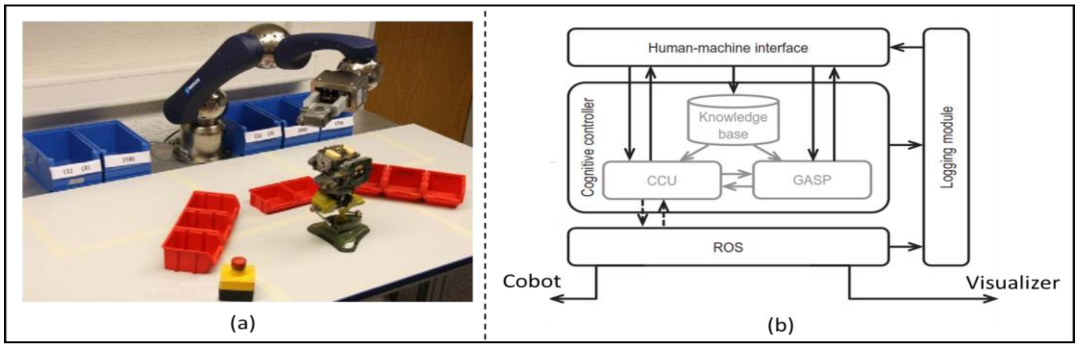


Figure 3.10: [FBS15] (a) Cooperative workcell layout – (b) Control system architecture.

The research in [FBS15, WKV+17, and MDA+17] shares many ideas with [BR17]. [FBS15] studies a cooperative assembly case study via the workcell that is shown in Figure 3.10a. The goal of this workcell is the assembly of the vehicle Stromberg carburettor. The worker assembles some parts such as the diaphragm, the springs, as these parts need the human extensive sensorimotor skills, while the cobot SCHUNK-LWA-4P [sch16] autonomously assembles the other carburettor parts. The system architecture is stated to be underdevelopment. However, Figure 3.10b shows its main idea. Over the workcell physical level, ROS has been used as the cobot automation middleware. Over the workcell control level, a cognitive controller has been proposed. The first component of the proposed controller is a **G**raph-based **A**ssembly **S**equence **P**lanner (**GASP**) that contains the sequence to assemble the carburettor. The task assignment must be done by the controller second component that is the **C**ognitive **C**ontrol **U**nit (**CCU**), the research proposed Soar cognitive RE to implement the CCU. The final component of the controller is an IF-THEN knowledge base. The implementation or the results of the concept are not shown in this research as it is in an initial stage. Nevertheless, the proposed technologies such as ROS and Soar provide a good guidance for the other researchers.

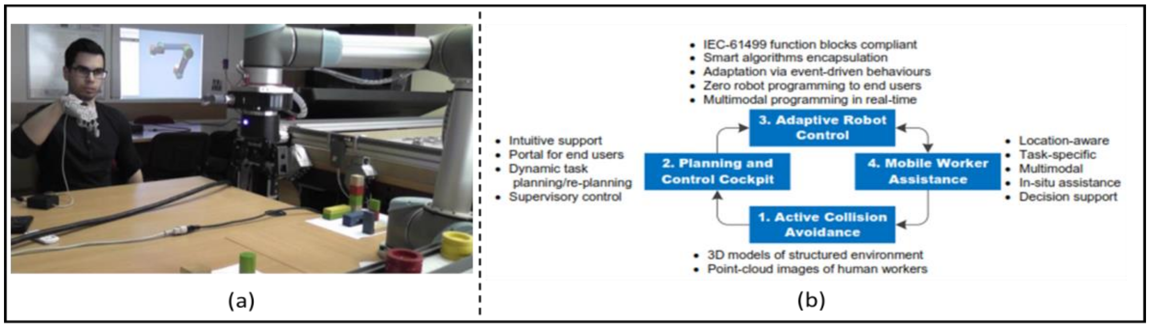

• IEC-61499 function blocks compliant
• Smart algorithms encapsulation
• Adaptation via event-driven behaviours
• Zero robot programming to end users
• Multimodal programming in real-time
3. Adaptive Robot Control
• Intuitive support
• Portal for end users
• Dynamic task planning/re-planning
• Supervisory control
2. Planning and Control Cockpit
4. Mobile Worker Assistance
• Location-aware
• Task-specific
• Multimodal
• In-situ assistance
• Decision support
1. Active Collision Avoidance
• 3D models of structured environment
• Point-cloud images of human workers
(a)
(b)

Figure 3.11: [WKV+17] (a) Cooperative workcell layout – (b) Control system architecture.

The research in [WKV+17] provides a design guidance to develop the cooperative workcell concept. The workcell in Figure 3.11a has been used to test the developed concept. The workcell contains a universal robot-UR5 (refer to Table 2.2 in section 2.1.2), along with a gesture control interface and a robot visualizer. The first step in the design guidance is to ensure the active collision avoidance by modelling the cooperative environment in 3D and tracking the human worker motion. The second step is to plan the shared tasks and supervise the dynamics at the workcell. Therefore, an adaptive control can be applied in step three. Finally, step four is accomplished by providing the worker with task-associated information. The research does not provide a solid implementation of its concept. Instead, it recommends IEC 61499 FB as a proper distributed technology to implement the concept. IEC 61499 FB is also emphasised by the research in [SAP+17] to design a cooperative manufacturing control solution.

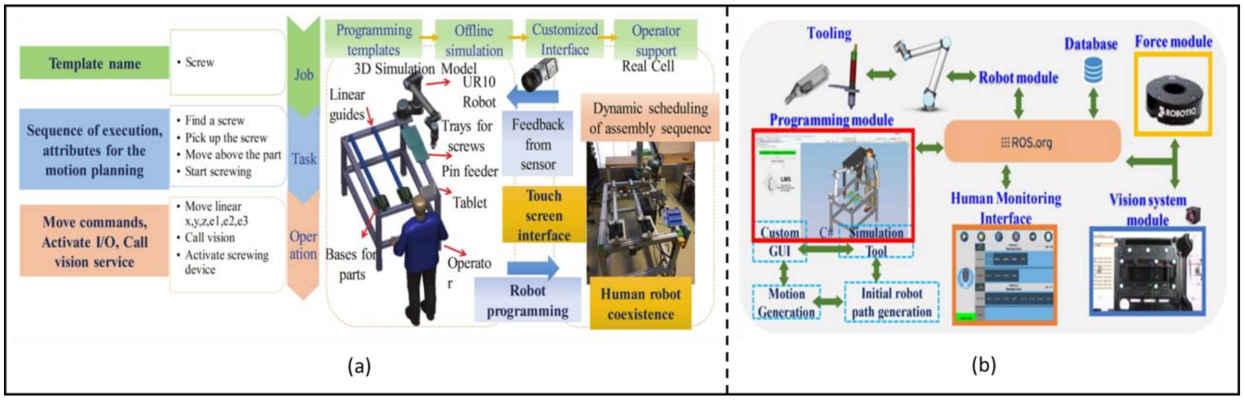


Figure 3.12: [MDA+17] (a) Cooperative workcell layout – (b) Control system architecture.

Despite that, the research in [MDA+17] is mainly addressing the coexistence instead of the cooperation. It follows the previous reviewed concepts and provides richer implementation details. The main goal of the research is to develop a flexible workcell programming and executing control. Figure 3.12a shows the structure of the proposed workcell. The implementation idea is following the task-oriented programming, where a worker or an operator who does not have any knowledge of programming, assembles a product in a 3D environment, and then the cobot performs the real assembly process. In order to do that, the manufacturing control system architecture that is shown in Figure 3.12b, converts the 3D assembly into a robot code that is executed in real-time. The workcell is composed of a universal robot-UR10 that is responsible for the product assembly. An android-based tablet provides a GUI to the operator to customize the 3D assembly by selecting the parts and connecting them in one model. Two industrial cameras monitor both the cobot and the operator, to control the execution sequencing. Other force and torque sensors were required to measure the cobot assembly forces. The integration of the sensors in addition to the cobot has been done via ROS, while the communication among them has been conducted over **T**ransmission **C**ontrol **P**rotocol/**I**nternet **P**rotocol (**TCP/IP**) protocol.

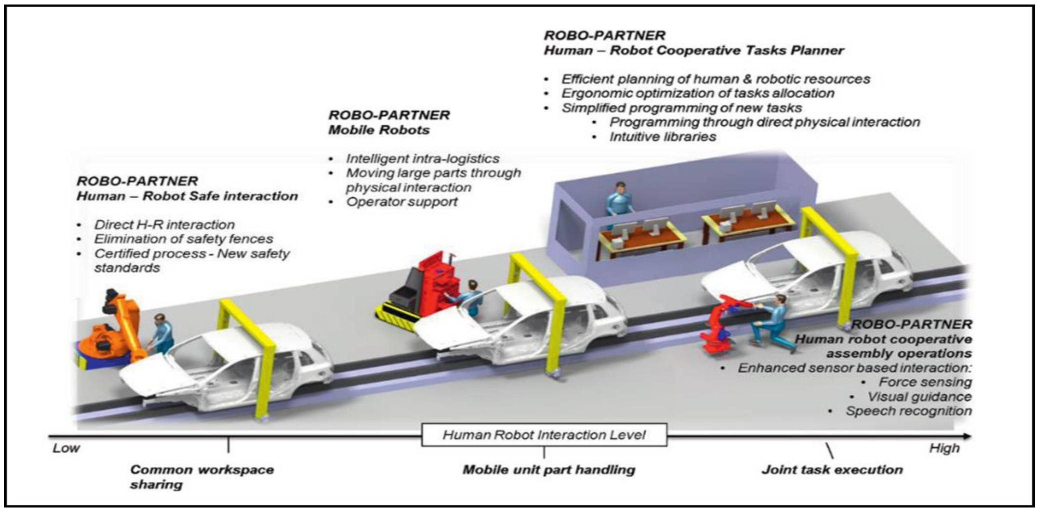

ROBO-PARTNER
Human – Robot Cooperative Tasks Planner
Efficient planning of human & robotic resources
Ergonomic optimization of tasks allocation
Simplified programming of new tasks
Programming through direct physical interaction
Intuitive libraries
ROBO-PARTNER
Mobile Robots
Intelligent intra-logistics
Moving large parts through physical interaction
Operator support
ROBO-PARTNER
Human – Robot Safe interaction
Direct H-R interaction
Elimination of safety fences
Certified process - New safety standards
ROBO-PARTNER
Human robot cooperative assembly operations
Enhanced sensor based interaction:
Force sensing
Visual guidance
Speech recognition
Human Robot Interaction Level
Low
High
Common workspace sharing
Mobile unit part handling
Joint task execution

Figure 3.13: [MMS+14] Robo-Partner production paradigm.

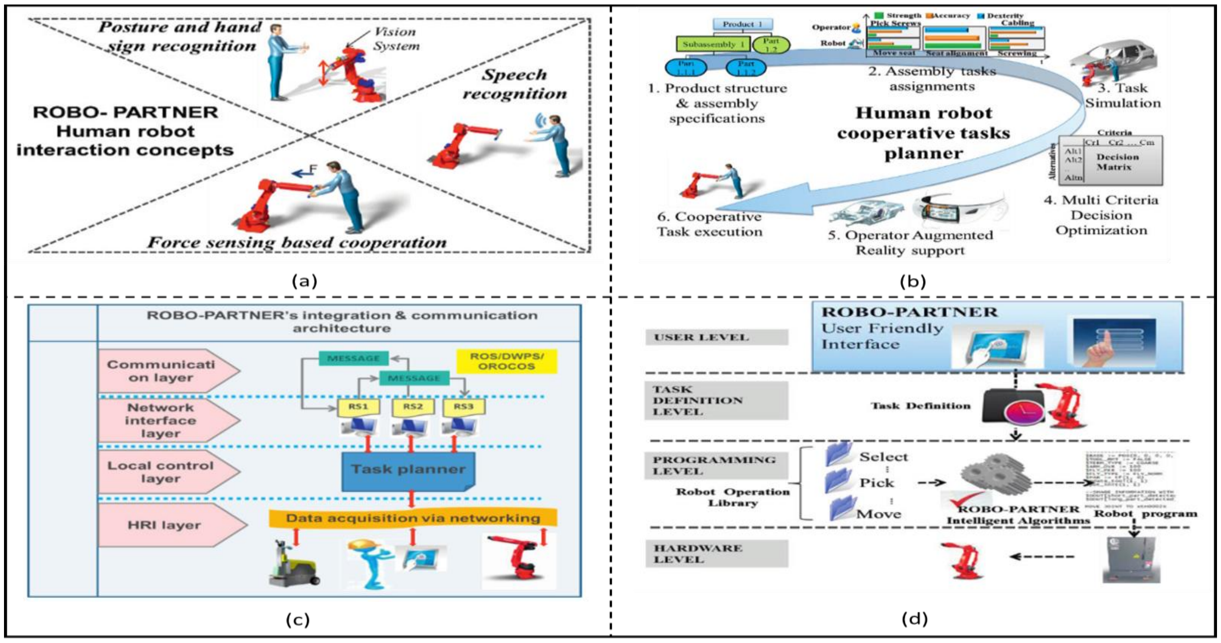

Posture and hand sign recognition
Vision System
Speech recognition
ROBO- PARTNER Human robot interaction concepts
Force sensing based cooperation
(a)
1. Product structure & assembly specifications
2. Assembly tasks assignments
3. Task Simulation
Human robot cooperative tasks planner
4. Multi Criteria Decision Optimization
5. Operator Augmented Reality support
6. Cooperative Task execution
(b)
ROBO-PARTNER's integration & communication architecture
Communication layer
Network interface layer
Local control layer
HRI layer
MESSAGE
ROS/DWPS/ OROCOS
RS1
RS2
RS3
Task planner
Data acquisition via networking
(c)
USER LEVEL
ROBO-PARTNER User Friendly Interface
TASK DEFINITION LEVEL
Task Definition
PROGRAMMING LEVEL
Robot Operation Library
Select
Pick
Move
ROBO-PARTNER Intelligent Algorithms
Robot program
HARDWARE LEVEL
(d)

Figure 3.14: [MMS+14] (a) Multimodal interaction – (b) Task planner concept – (c) Integration architecture – (d) Automation architecture.

The research in [MMS+14] proposes a variety of ideas, technologies, and a generic architecture to build the cooperative production paradigm as shown in Figure 3.13 and Figure 3.14. The research is a part of an EU project called Robo-Partner, which aims to support the automotive manufacturing assembly by integrating the worker into the process. The research gave a very generic production paradigm of the cooperative factory shop floor, which can be seen in Figure 3.13. The idea of the paradigm is to involve the worker within every operation in the vehicle assembly process. The research proposes a group of interaction techniques such as hand gesture recognition to cooperate with the robot during the assembly, as shown in Figure 3.14a. Then it discusses the cooperative task planner, which starts from the product planning and passes by the resource assignment until the execution as show in Figure 3.14b. In Figure 3.14c, this task planner represents the second layer of the integration architecture, which is connected to the data acquisition at the first layer, then the integration architecture proposes the networking and communication layers at the rest of the architecture. Finally, the research proposes another automation architecture that focuses on the tasks and the programming within the cooperative shop floor, as shown in Figure 3.14d. As a sum up, this research scratches the surface of the cooperative manufacturing challenges by giving very few details regarding the technologies or the concepts that can fulfil the proposed ideas.

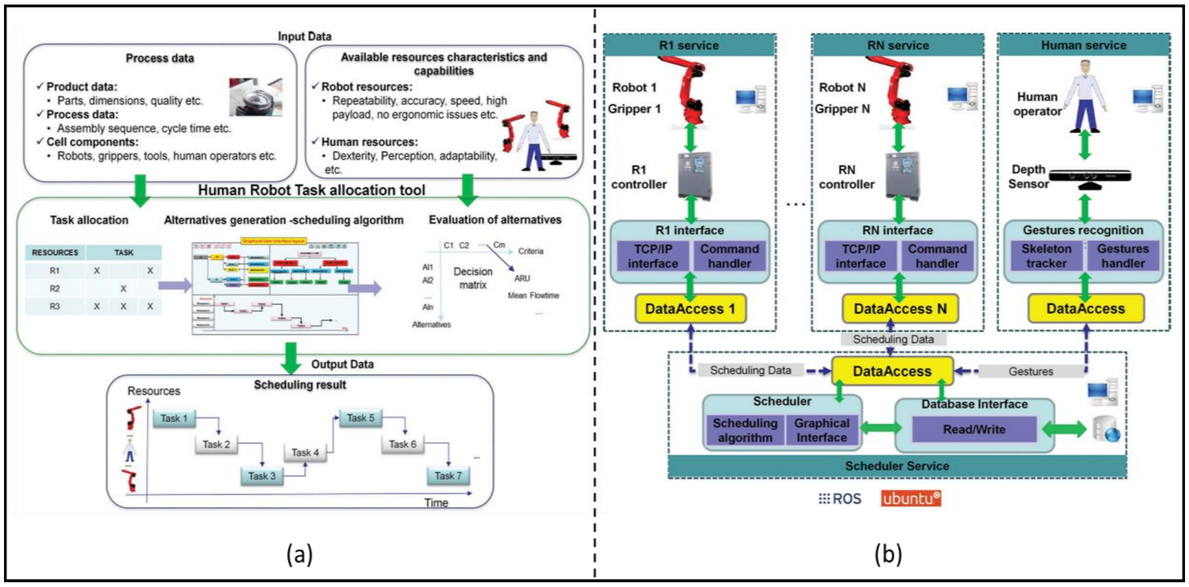


Figure 3.15: [TMM+17a, TMM+17b] (a) Task allocation method – (b) Control system architecture.

An extension of the previous research has been introduced via [TMM+17a, and TMM+17b], as shown in Figure 3.15. The task allocation method via these articles has been achieved via a set of decision-making based on three criteria, which are the resource suitability, the resource availability, and the operation time. The resources suitability denote the capability of an operational resource to carry on a specific task. The resource availability donates if the resource is available when the task is assigned. The operation time is the time needed from the resource to execute the task. Based on these criteria, the research proposed an offline simulation tool to assist the task planning based on maximizing a utilization formula. One of the parameters that has been considered during the task planning is WTP. It is not clear how this time parameter has been calculated during the offline simulation, but apparently, it is presumed in advance before deploying the simulation. If so, this makes the solution approach a bit shaky. Inasmuch as the task operation time of the worker is variable and changes due to uncountable factors. Two case studies have been used to test the research approach. During the two cases, the number of the cooperative resources is always fixed. That means that each time the number of the resources changes, the simulation must be rebuilt and redeployed again. In other words, the research does not put into consideration the adaption to the fluctuation in the number of the operational resources as a dimension of the cooperative workcell. Moreover, the main motive for the research is to distribute the tasks among the available operational resources, and it does not schedule the tasks based on the variation of the production. The idea of using WTP to plan and simulate the cooperative workcell has been used by the research in [ULT+18] as well.

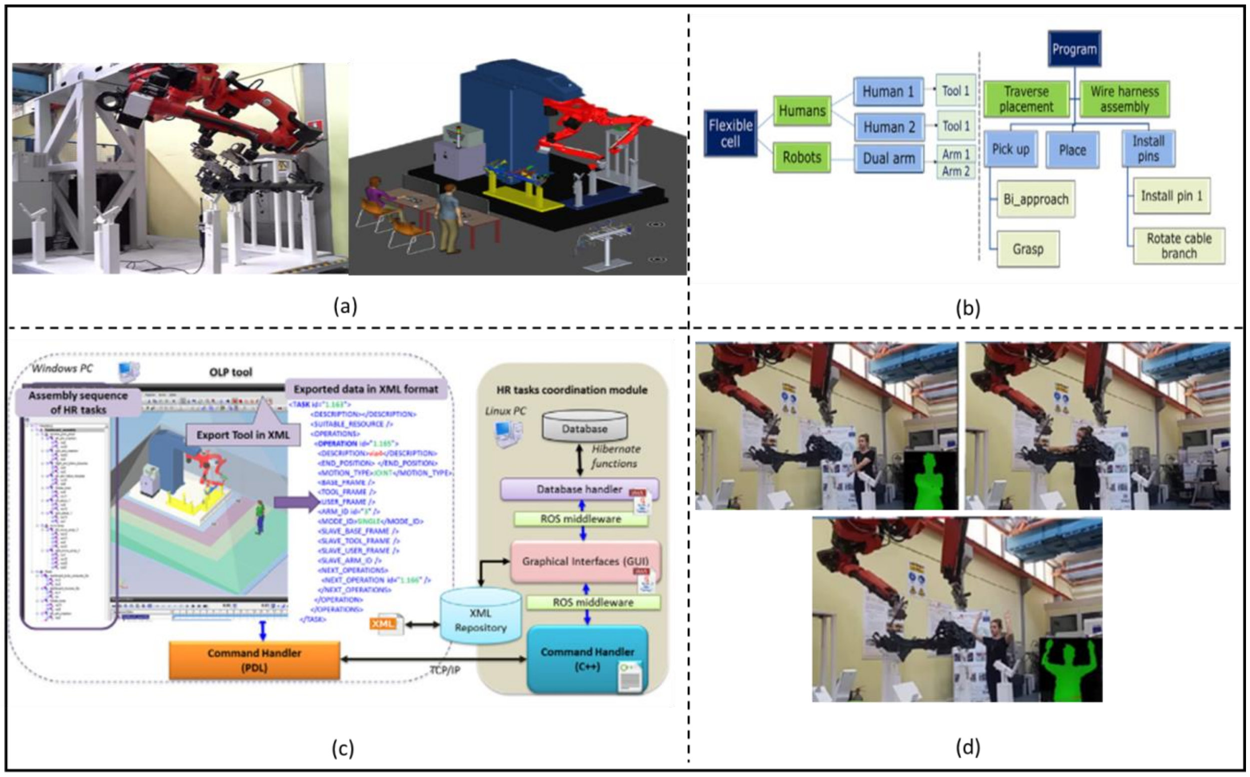


Figure 3.16: [MTM+17, TMM+15, TMC16] (a) Case study layout and simulation – (b) Resources and tasks model – (c) Offline programming architecture – (d) Interaction via body gestures.

The research in [MTM+17, TMM+15, and TMC16] uses the same offline simulation tool that has been developed in [TMM+17a, TMM+17b], to implement a case study of a dual-arm robot in cooperation with two workers during an assembly scenario. Figure 3.16a shows the layout of the case study at the left side of the figure. While at the right side of the Figure 3.16a, the case study is represented via the offline simulator. Moreover, the resources and tasks models, which are inputs for the task planner are shown in Figure 3.16b. The research connects the offline simulator to a dual-arm robot to control and plan its action during the cooperation. Therefore, it converts the simulation 3D model to an e**X**tensible **M**arkup **L**anguage (**XML**) format, which is integrated with ROS to control the dual-arm robot, as shown from Figure 3.16c. The control architecture is also connected to an industrial 3D camera system (SafetyEYE [Pil17]). the camera system is used to interact with the robot during the configuration and the operation, as shown in Figure 3.16d. Starting, stopping, or ending the human task execution are all examples of the commands that can be conducted via the body gestures at the proposed case study.

Two technical details of the previous implementation are important to analyse. First, even that ROS is a very common solution to control the latest robots, using ROS as the whole communication and networking infrastructure would cause bigger problems with respect to the system reliability and scalability. Using ROS might be adequate only in case of very small case studies, where all the automation interfaces are grouped over one device. This is because ROS communication architecture depends on one central master node, which organises all the commutation parameters within the network. If the automation device that contains ROS master node fails, the whole system fails behind. In addition, ROS uses very basic message structure, which can hold only basic data types such as Strings, Integers, or Boolean. This messaging structure can be seen as an advantage and disadvantage at the same time. The advantage comes from the fact that a simple message structure makes it fast to stream data from the shop floor. The disadvantage comes from the fact that it is impossible to ROS message to contain a complex representation of the cooperative environment. The details of ROS middleware are going to be discussed in section 4.1.1.2.

The second point to analyse is using the body gestures to interact with the manufacturing control system in the context of cooperative manufacturing. Body gestures have been often used in HRC. However, cooperative manufacturing adds new factors to the HRC equation. Some of these factors are the value of space over the shop floor. Body gesture recognition needs a huge space in comparison with hand gesture recognition as an alternative, which results in a huge waste of space. Especially if we put into consideration that the main motive for cooperative manufacturing is to fulfil the SMEs needs, whom do not have the luxury to afford huge manufacturing areas. Also, body gesture recognition is demanding, time consuming, and unnatural method for the worker to interact during the manufacturing. Furthermore, other considerations such as the cleanliness of the industrial environment will absolutely influence the accuracy of the camera recognition system. In addition to, the psychological comfort and acceptance of the worker to operate under camera surveillance system. The details and the comparison among the recognition methods are going to be discussed in section 6.2.2.2.

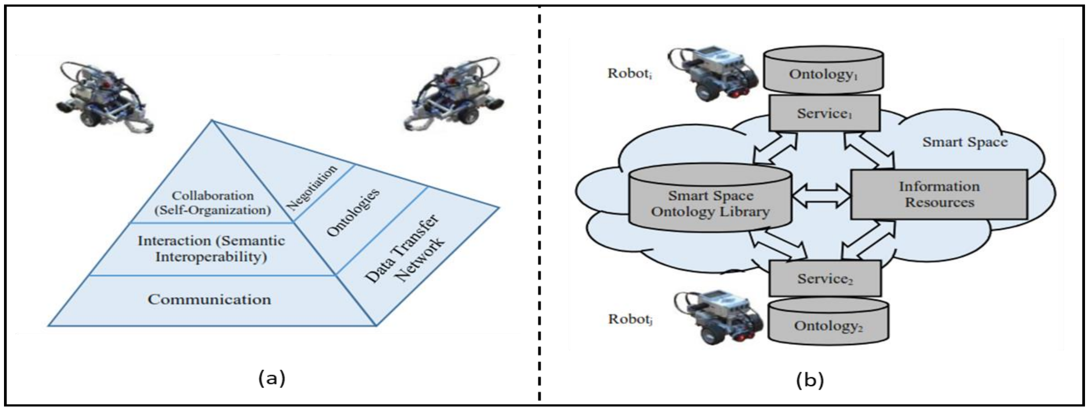


Figure 3.17: [SKM+16] (a) System architecture model – (b) System implementation model.

Although the research in [SKM+16] does not directly address the cooperative manufacturing. Instead, it aims to achieve the cooperation among a group of mobile robots, as shown in Figure 3.17a. Although, it provides an interesting concept to represent the mobile robots domain via an ontology model. The research proposes a three-layered architecture, which starts with the communication layer, followed by an ontology layer, and finally a negotiation layer. The implementation of this architecture has been achieved via an agent-like middleware which is called Smart-M3 [HLB+10]. Smart-M3 is an open source software that originally developed by Nokia to deploy the semantic web concept in communication [Nok17]. Although, the technical details of the implementation in Figure 3.17b are not clear, the result of the research shows the ability of its proposed solution to achieve the required cooperation. A similar approach in cooperative automation system development via the ontology models has been followed in [GGG+06]. [GGG+06] is also using an agent-oriented software to develop the web ontology models of its case study. The different ontology modelling languages will be discussed in more details in section 4.2. Moreover, the implementation of the ontology concept in manufacturing control will be reviewed via the next section of the literature review.

3.3 *Holonic Control Architecture*

There is a misconception among the automation control researchers between the holon and the autonomous agent [GB04]. This misconception comes from the fact that both the holon and the agent share common characteristics. However, the difference between them is that a holon is a conceptual approach to solve a distributed manufacturing problem, while an autonomous agent is a general-purpose technology, which can be used to implement the holon concept and other distributed concepts. Autonomous agent technology is commonly used to implement the holon model, but it is not the only one. IEC 61499 FB is another common technology among the researchers to implement the holon model [CB06]. Some researchers are theoretically combining between the two technologies to build the holon component. Furthermore, other technologies such as ROS or the industrial web service are not commonly used to implement the holon, but they can. In sum, there is no obligation to use a specific technology to implement the holon as it is highly depending on the application. This dissertation proposes a novel implementation model of the holon that fits the cooperative manufacturing requirement. The proposed holon implementation model is composed from three layers, which are the physical data layer, the information communication layer, and the knowledge reasoning layer. These three layers will be explained in details through chapter 4. Therefore, this part of the literature survey is focused on the technologies that have been previously used to implement the holon model. Furthermore, as the cooperative manufacturing research is still in its initial phase, similar solutions of other similar manufacturing problems will be considered. Those problems such as the reconfigurable and the agile manufacturing control systems share many common features with the cooperative manufacturing system.

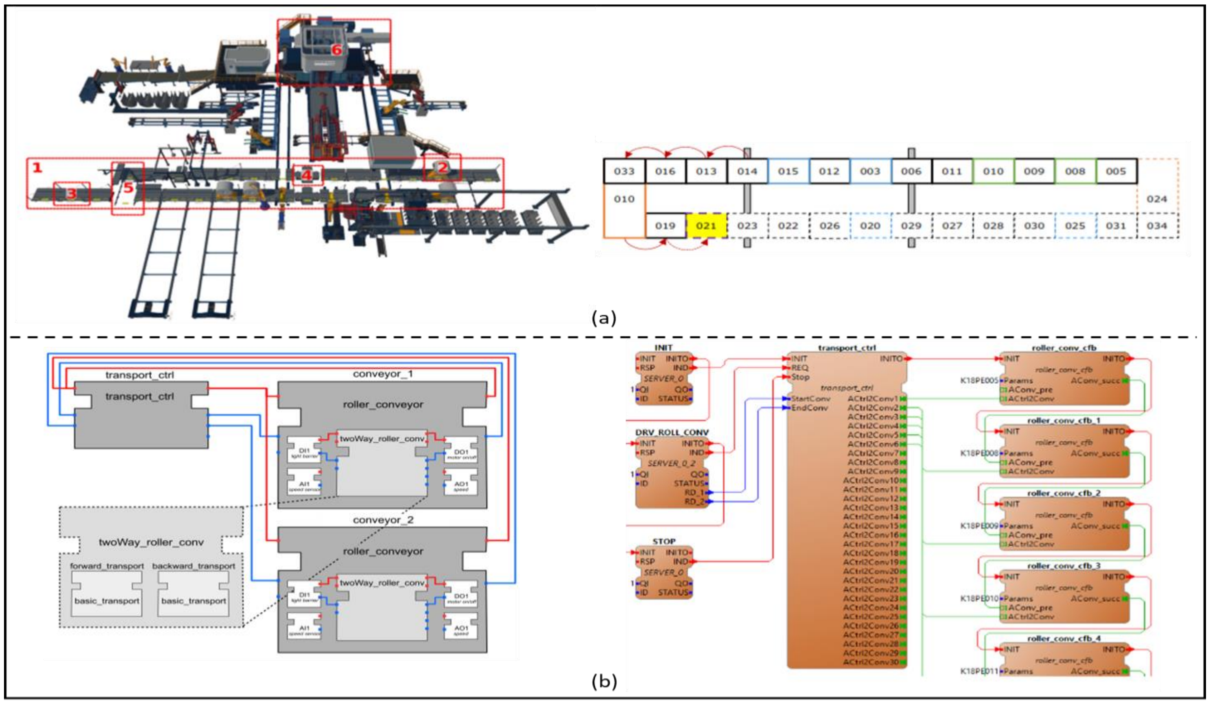


Figure 3.18: [TWZ+17] (a) Case study - a pallet transport system – (b) FB distributed control solution.

The research in [HDW+17] uses IEC 61499 FB technology in the real-time data acquisition of industrial cyber-physical system. In addition, the research demonstrates the ability of the technology to be integrated to other distributed solutions such as **D**istributed **C**ontrol **S**ystem (**DCS**) and **S**upervisory **C**ontrol **A**nd **D**ata **A**cquisition (**SCADA**) system. Furthermore, [CB06, TWZ+17] propose IEC 61499 FB as the next generation of automation software and as a potential solution for Industry 4.0. [CB06] shows the advantages in IEC

61499 standard by comparing it to its predecessor IEC 61131-3, also it reviews the different frameworks which can be used to develop IEC 61499 FB such as HOLOBLOC **F**unction **B**lock **D**evelopment **K**it (**FBDK**). [TWZ+17] provides a real case study problem that applies industry 4.0 recommendations on the shop floor. The case study is a reconfigurable pallet conveyor system that provides a flexible routing system from all the sources to all the destinations, as shown in Figure 3.18a. The system solution starts from the FB design, which represents the control I/O of a conveyor strip, as shown at the left side of Figure 3.18b. The next step is to connect the different instances of the conveyor strip FB in one control solution, which is shown at right side of Figure 3.18b.

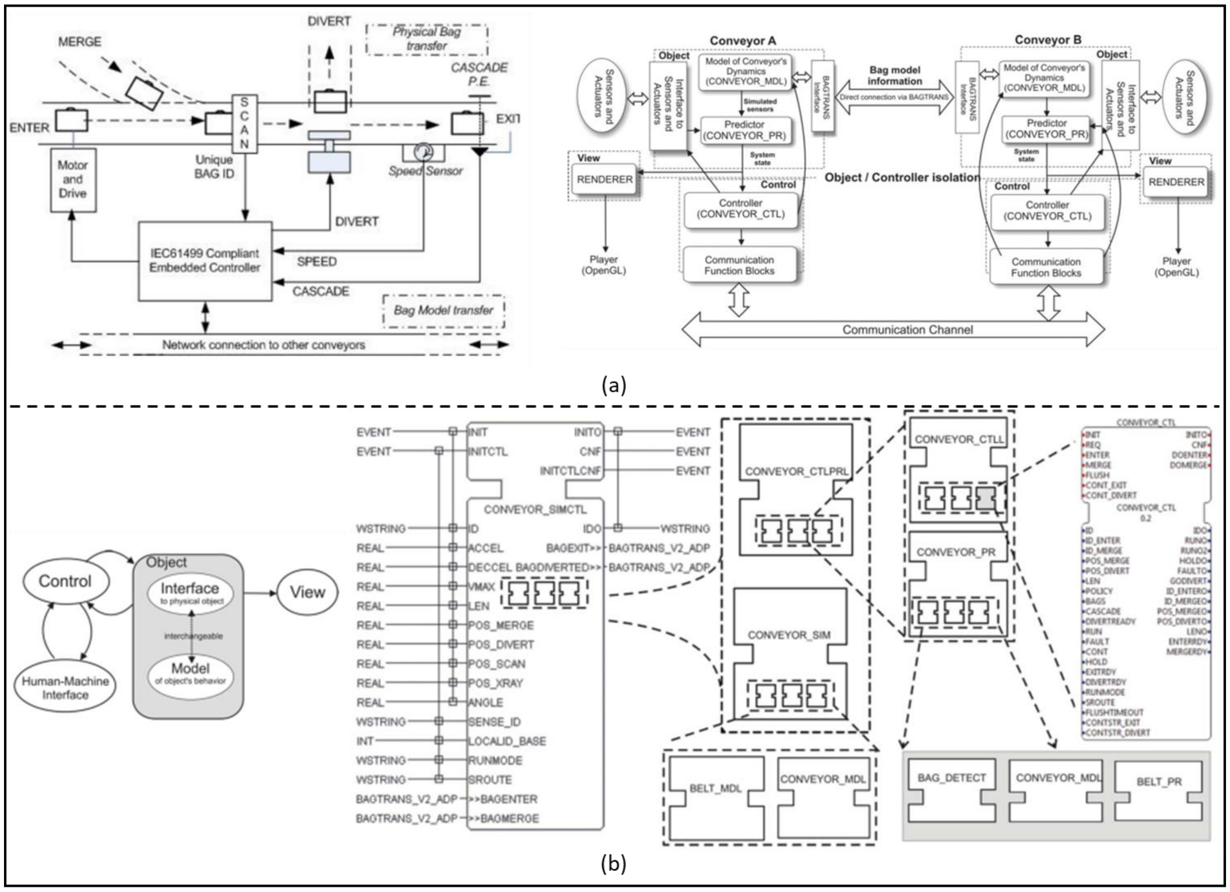


Figure 3.19: [BV09] (a) Case study - a baggage handling system – (b) FB distributed control solution.

A similar case study to [TWZ+17] has been also the focus of [BV09]. An intelligent control solution of an airport **B**aggage **H**andling **S**ystem (**BHS**) has been deeply studied, as shown in Figure 3.19a. The research provides an extendible distributed FB solution, by modelling every component in the BHS as an IEC 61499 FB. It is dividing the whole conveyor transportation system into many segments, as shown in Figure 3.19b. Every conveyor segment is presented as a customized FB. Moreover, it models every bag over the BHS as another autonomous FB. All the instances of the conveyor segment FB and the bag FB are composing together the MAS, which provides a distributed decision-making process. Two important remarks distinguish this research. The first is the reusability, extensibility, and scalability of the proposed solution, as it uses two different customized FBs to present all the instances of the conveyor segments and the bags. The second remark is the viability and the feasibility of the solution to be practically implemented, as the research provides important technical details to deploy the solution over the compliant controllers.

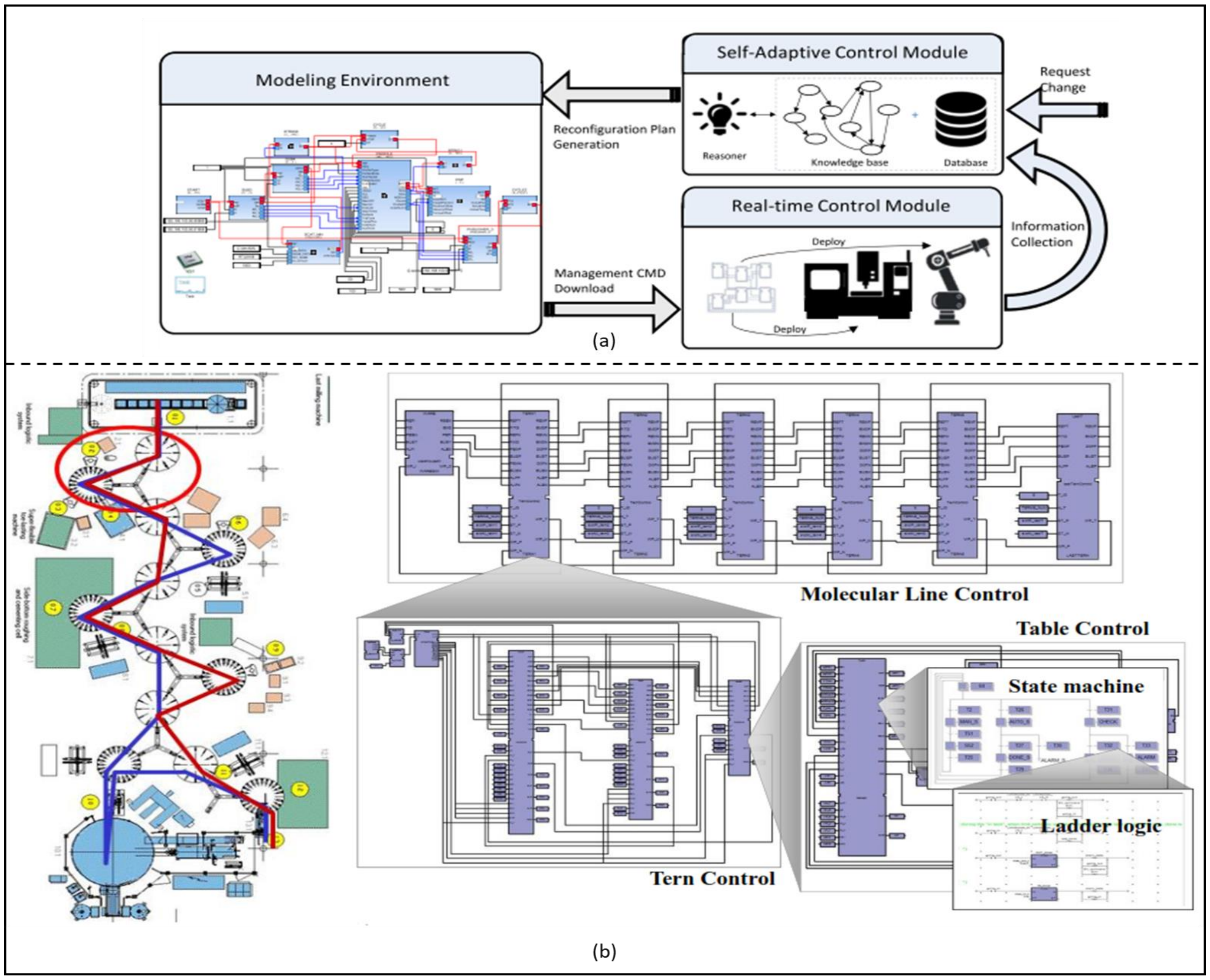


Figure 3.20: (a) [ZLL+17] Adaptive control via IEC 61499 – (b) [Vya11] Molecular line control model.

IEC 61499 technology can be used to model the manufacturing component, therefore it is deployed as a physical control interface between the machines and the controller. [ZLL+17] proposed a real-time self-adaptive controller that uses IEC 61499 to model the manufacturing environment as shown in Figure 3.20a. However, modelling the manufacturing environment via IEC 61499 FBs can be so complex when the system size increases. This can be seen via [Vya11] model of a molecular production line that is shown in Figure 3.20b. The main reason of this complexity is the communication technique that is followed by IEC 61499 technology. The communication among IEC 61499 FBs is based on publish/subscribe pattern [HDW+17]. This negatively influences the control solution in two drawbacks. First, whenever a basic FB needs to send a message, it requires another publisher FB to perform the sending process, plus another subscriber FB to receive the message on at the receiver side. This causes an exponential aggregation of the system, which not only makes it so hard to design the system, but also at some point, it becomes impossible to debug the control solution. The second drawback comes because the publish/subscribe pattern is efficient only when sending basic data types, and cannot conduct highly structured information such as objects. The details of IEC 61499 technology will be the focus of section 4.1.1.1, to study its pros and the cons.

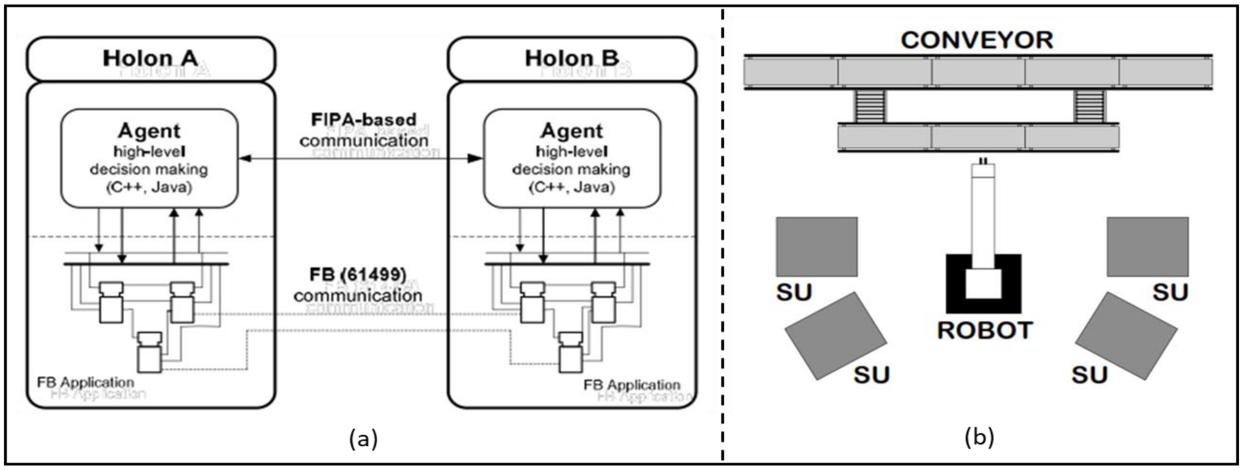


Figure 3.21: (a) [Vya10] a proposed holon model – (b) [KB13] an experimental reconfigurable assembly system case study.

[Vya10] realizes the communication drawback in IEC 61499. Therefore, he proposes a two-layered holon which is composed of IEC 61499 FB over the low control level and a FIPA agent over the high control level, as shown in Figure 3.21a. [Vya10] introduces a theoretical holon model. Thus, it provides a guidance to design both the low and the high control levels of the holon, based on a comprehensive explanation of the responsibilities of each component within the manufacturing system. However, the paper did not apply its theoretical model to a practical case study. This makes it hard to evaluate the model feasibility. The concern in [KB13] is to evaluate the implementation of the IEC 61499 FB vs the autonomous agent technology. The evaluation is based on the implementation of the control system of an experimental reconfigurable assembly system that is shown in Figure 3.21b. Nevertheless, the research does not combine the two technologies together, but it compares between them, then it evaluates them based on the qualitative and quantitative performance of different four hardware configurations. The research results state that agent technology requires less effort and time than IEC 61499 FB to implement three configurations of the system, as the IEC 61499 FB lacks inter-platform communication flexibility.

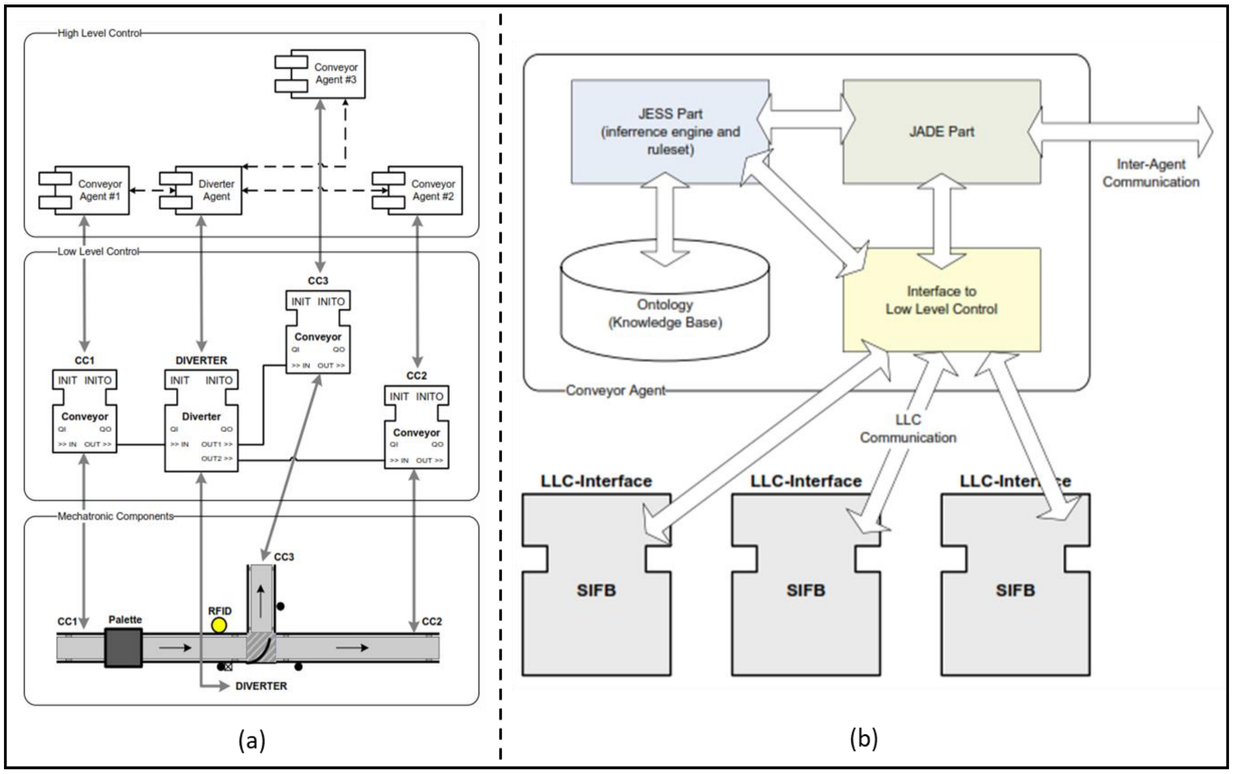


Figure 3.22: [HHZ+08] (a) Case study control architecture – (b) Holon structural model.

[HHZ+08] provides a case study that uses the previously explained two-layered holon architecture to control a conveyor system as shown in Figure 3.22a. The implementation of this case study has been done via the GUI that is provided by the FBDK framework. Furthermore, the research proposes to upgrade the high-level control of the holon structural model, by adding an RE called Jess, as shown in Figure 3.22b. Additionally, the research proposes to represent the system domain via an ontology-based knowledge. There are no details regarding the interaction mechanism or the functions of every component within the proposed model. Thus, this reserach stands as an initial guidance to build the holon implementation model that has been introduced in this dissertation.

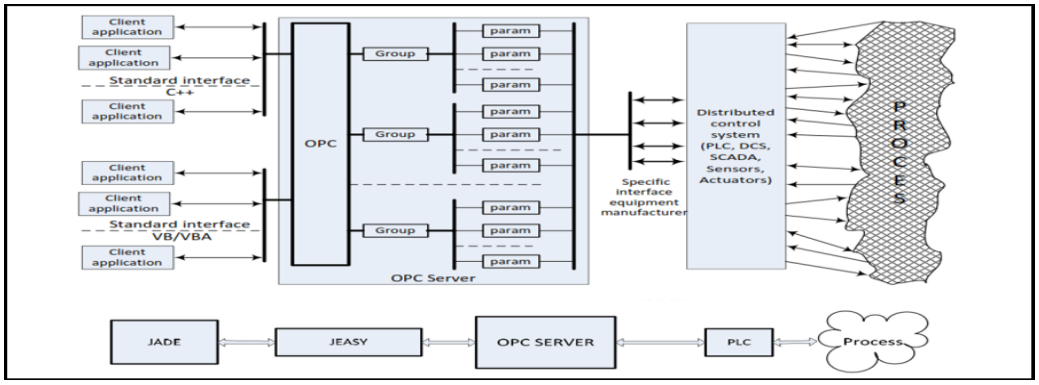

Client application
Standard interface C++
Standard interface VB/VBA
OPC
Group
param
OPC Server
Specific interface equipment manufacturer
Distributed control system (PLC, DCS, SCADA, Sensors, Actuators)
PROCESS
JADE
JEASY
OPC SERVER
PLC
Process

Figure 3.23: [DS12] JADE integration to the convention automation devices via OPC.

Despite the fact that autonomous agent technology is an important focus among the scientific researchers, it is not so commonly used in industry. The main reason behind that is the hardware compatibility. On the one hand, an autonomous agent is a pure software code that is usually written via a high-level programming language. On the other hand, the conventional automation devices such as **P**rogrammable **L**ogic **C**ontrollers (**PLC**s) are rarely to support high-level programming. This in practice causes a huge problem as the present shop floor infrastructure is built upon the conventional automation devices. The research in [DS12] realizes the previously mentioned problem and realizes the advantage of the MAS distributed control as well. Therefore, it proposes an approach that integrates JADE middleware to the conventional automation devices via an **O**pen **P**latform **C**ommunications (**OPC**) as shown in Figure 3.23. A series of experiments have been conducted to show the ability of the approach to fulfil the real-time control. A similar problem will arise during the implementation of this dissertation to integrate JADE to ROS. Thus, a similar integration approach will be discussed in details in section 6.2.3.2.

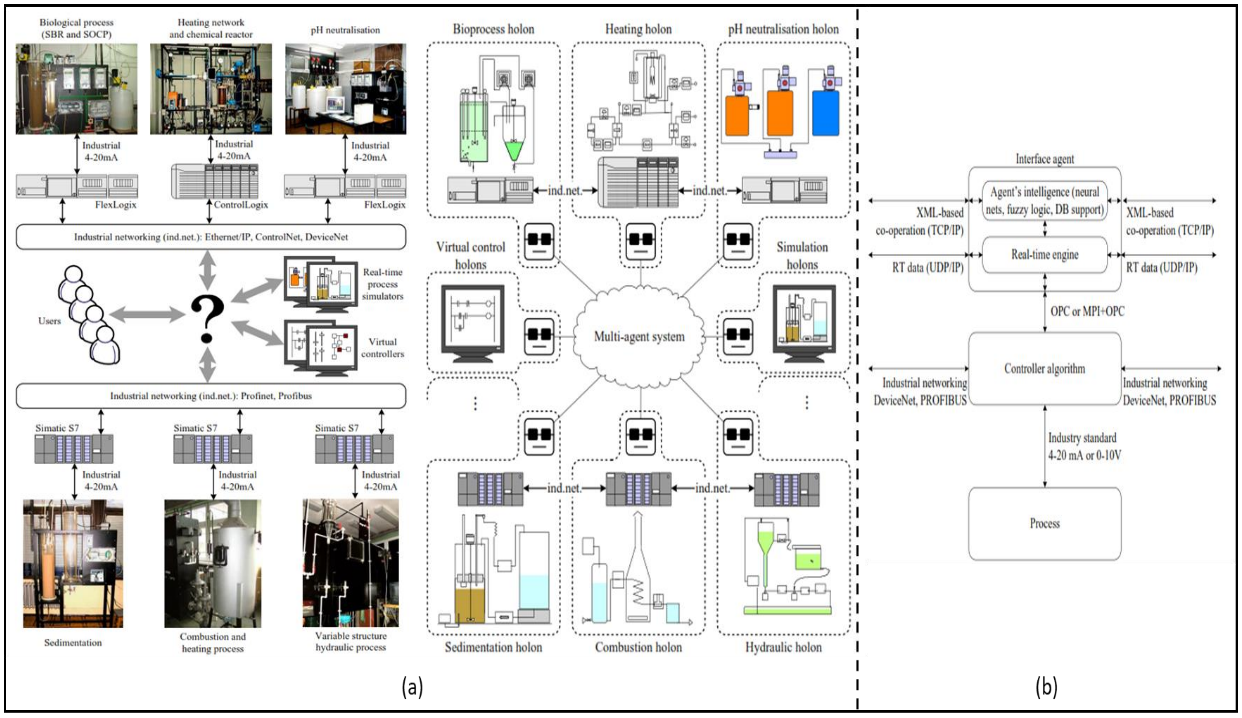
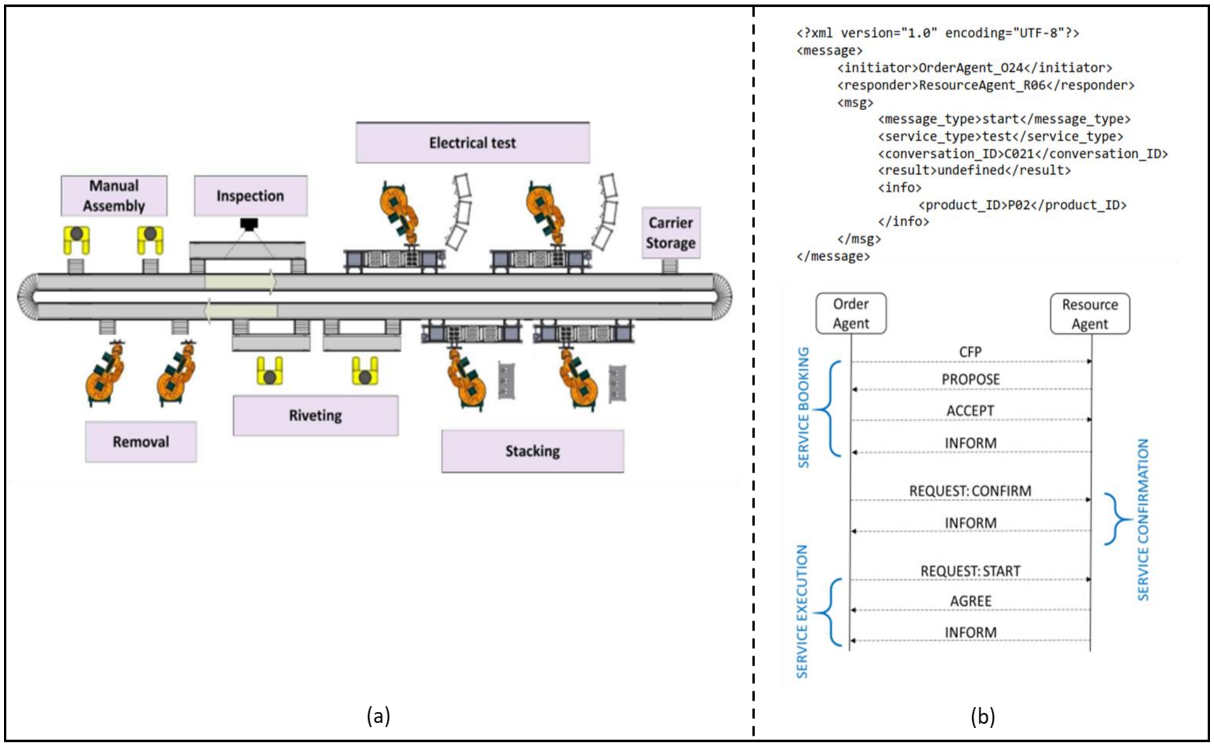

Biological process (SBR and SOCP)
Heating network and chemical reactor
pH neutralisation
Industrial 4-20mA
FlexLogix
ControlLogix
Industrial networking (ind.net.): Ethernet/IP, ControlNet, DeviceNet
Users
Real-time process simulators
Virtual controllers
Industrial networking (ind.net.): Profinet, Profibus
Simatic S7
Sedimentation
Combustion and heating process
Variable structure hydraulic process
Bioprocess holon
Heating holon
pH neutralisation holon
ind.net.
Virtual control holons
Simulation holons
Multi-agent system
Sedimentation holon
Combustion holon
Hydraulic holon
(a)
Interface agent
Agent's intelligence (neural nets, fuzzy logic, DB support)
Real-time engine
XML-based co-operation (TCP/IP)
RT data (UDP/IP)
OPC or MPI+OPC
Controller algorithm
Industrial networking DeviceNet, PROFIBUS
Industry standard 4-20 mA or 0-10V
Process
(b)

Figure 3.24: [MP07] (a) Case study control schematic – (b) Holon structural model.

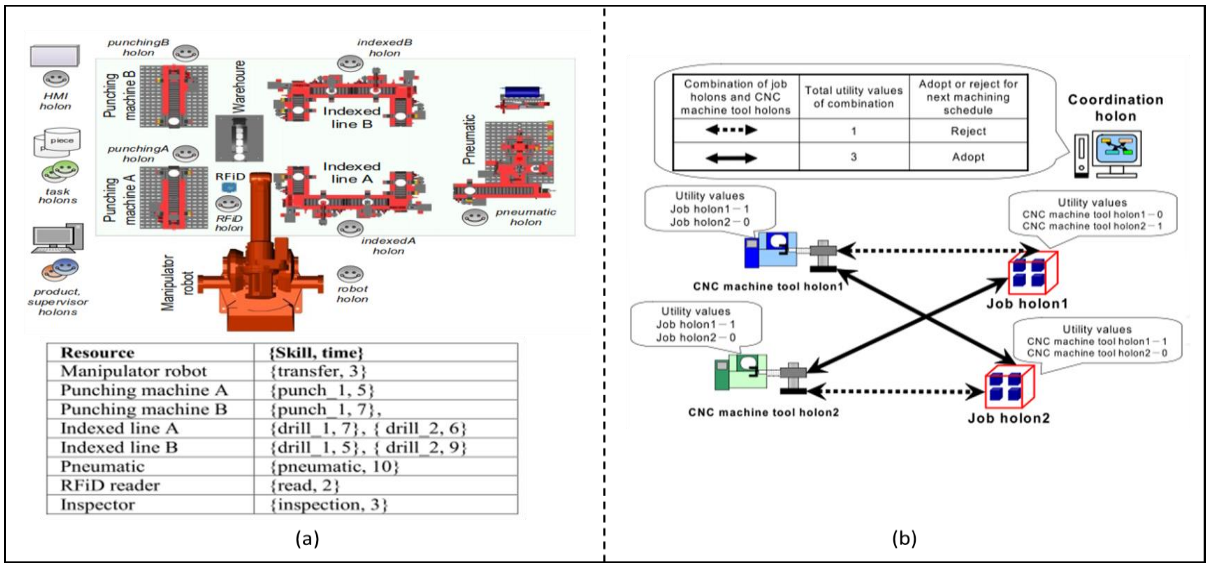
[MP07] implements the idea of the previous research by integrating the PLC controllers of an industrial plant via the MAS as shown in Figure 3.24a. The plant contains a group of commonly used PLCs in industry such as Siemens Sematic Step7 and Rockwell Logix controllers. The solution approach applies the HCA paradigm by gathering all the operational resources that belong to the same group under the same holon. For example, a holon is responsible for the heating system control. Furthermore, the approach benefits from the real-time cross interaction among the holons. This has been achieved via exchanging XML messages over **T**ransmission **C**ontrol **P**rotocol (**TCP**) and **U**ser **D**atagram **P**rotocol (**UDP**), as shown in Figure 3.24b.



```
<?xml version="1.0" encoding="UTF-8"?>
<message>
      <initiator>OrderAgent_O24</initiator>
      <responder>ResourceAgent_R06</responder>
      <msg>
            <message_type>start</message_type>
            <service_type>test</service_type>
            <conversation_ID>C021</conversation_ID>
            <result>undefined</result>
            <info>
                  <product_ID>P02</product_ID>
            </info>
      </msg>
</message>
```

Figure 3.25: [KB18] (a) Case study schematic – (b) Holons XML-based Interaction.

[KB18] applies PROSA reference model to control the operations of an electrical circuit breaker assembly and testing line, which is shown in Figure 3.25a. The research is using an XML message interaction via JADE middleware, to provide service-oriented interaction scenarios. Figure 3.25b shows an example of this interaction scenario, where the XML message contains related information to the manufacturing service and the product type. The services are following the common FIPA Rational Effect pattern, by sending a request message then receiving a confirmation message. This communication pattern is very simple and effective as it ensures the message reception. A similar handshaking communication pattern will be followed during the implementation phase of the dissertation in chapter 6, to ensure the communication reliability.



Resource	{Skill, time}
Manipulator robot	{transfer, 3}
Punching machine A	{punch_1, 5}
Punching machine B	{punch_1, 7},
Indexed line A	{drill_1, 7}, { drill_2, 6}
Indexed line B	{drill_1, 5}, { drill_2, 9}
Pneumatic	{pneumatic, 10}
RFiD reader	{read, 2}
Inspector	{inspection, 3}

Combination of job holons and CNC machine tool holons	Total utility values of combination	Adopt or reject for next machining schedule
◀---▶	1	Reject
◀━━▶	3	Adopt

Figure 3.26: (a) [BDP+16] Scheduling based on the time value – (b) [NNK10] Scheduling based on the utility value.

[BDP+16] applies ADACOR reference model to control the operations of a small size production system, which is shown in Figure 3.26a. The experimental case study composes of an IR (ABB-IRB-1400) that is responsible for parts transportation, a pneumatic station, two punishing machines, two indexed lines, and one warehouse. All the previously mentioned physical resources represent the ORHs during the implementation. Each resource is represented as a function of its skill (i.e., capability) and the required time from the resource to perform this skill (i.e., T_P). The purpose of the article is to obtain the self-organisation based on ADACOR model. This is achieved by sequencing the manufacturing based on matching the resource skill to the required production operation, and scheduling the production plan based on the required time of each operation. Hence, the operation time has been assumed in advance because all the operation resources are representing machines. This assumption is theoretically valid due to the reliability of the machine to perform a task in a specific time. A similar approach has been followed during the dissertation to reschedule the cooperative workcell, which also assumed that the cobot operation time is fixed. However, in the worker case, a more realistic approach has been followed, by recording the WT_P then predicting the next operation time upon that. This approach will be illustrated in details in section 6.4. A more complex scheduling criterion can be also achieved by the HCA. This can be seen via [NNK10], where an objective function is used to calculate a utility value to schedule a group of CNC machines. The utility value is a function in different parameters such as machine efficiency, accuracy, cost, and operation time. Deciding which scheduling criteria to use is certainly depending of the application.

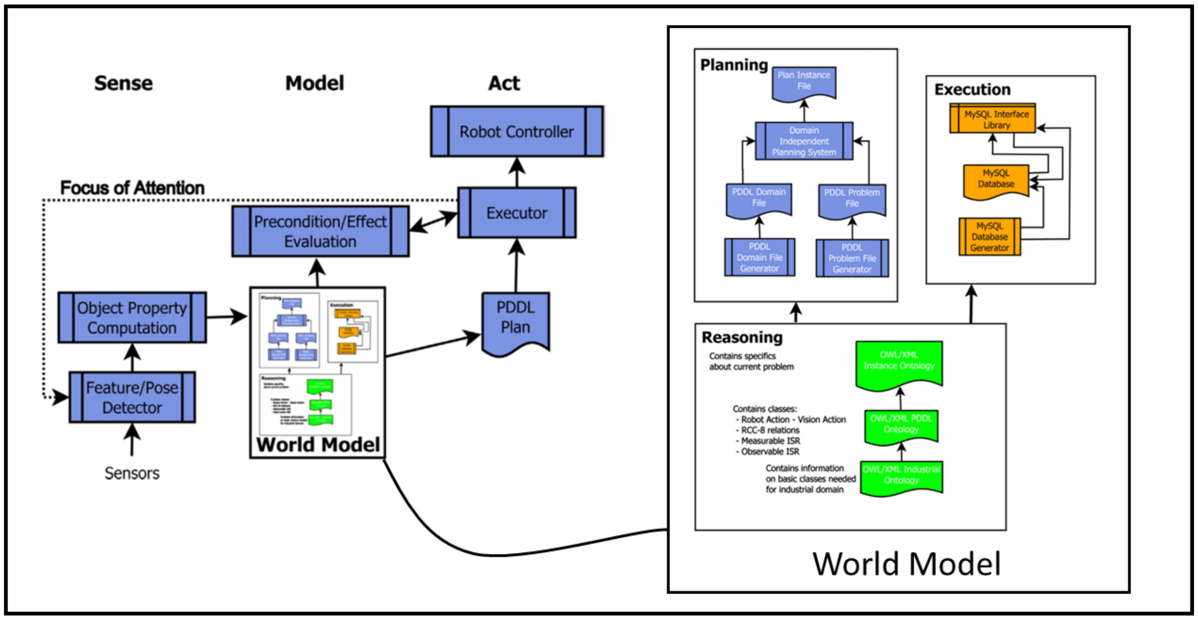


Figure 3.27: [Bal15] Sense-Model-Act paradigm.

The manufacturing environment representation problem has been addressed in [Bal15, FCG+15, LRT12, and RPL13]. Different domain models based on the ontology concept have been developed to represent the manufacturing environment during this literature review. [Bal15] uses an ontology model as a part of its Sense-Model-Act paradigm that is shown in Figure 3.27. The main research problem is the adaptation to the changes in the robot tasks at the manufacturing workcell. The research proposed a knowledge-driven solution by using the ontology to represent the relations among the element of the system architecture as well as the manufacturing knowledge. The world model is composed of three modules that are reasoning, planning, and execution. The reasoning module generates the basic information that is needed to build the product such as the parts, and the required functions that are hard coded over the robot such as pick and place function. The planning module creates static manufacturing plans that contain the set of actions from an initial to a final goal state. The execution module guides the sensor processing system to update specific information from the world model and gets the feedback to make sure that the plans are executed in the right sequence. As a remark of this research, the Sense-Model-Act paradigm that is followed is similar to the autonomous agent internal execution model that will be explained in details in section 4.1.2. More details regarding the structure of the product and the implementation method would make this research more clear. In addition, the reasoning algorithms or rules are not mentioned during the research, which makes it hard to understand the mechanism of inferring new knowledge.

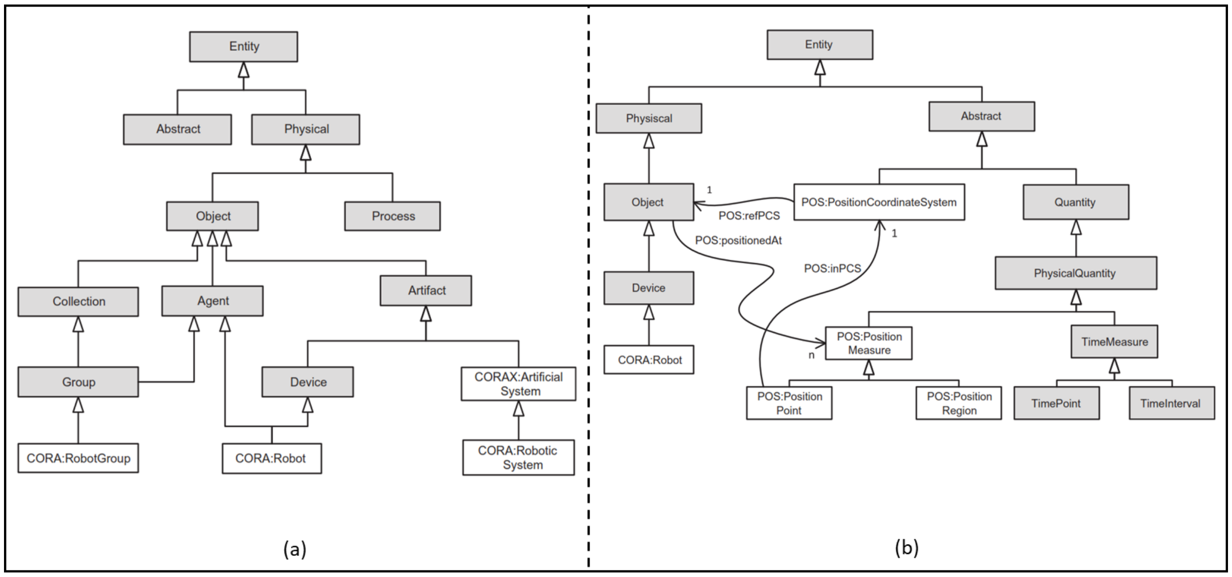


Figure 3.28: [FCG+15] (a) CORA ontology model – (b) CORAX ontology model.

The working group ontologies for robotics and automation along with the IEEE robotics and automation society has proposed a **C**ore **O**ntology for **R**obotics and **A**utomation (**CORA**) which is shown in Figure 3.28a. CORA aims to standardize the robot conceptual model that provides an explicit knowledge during designing the robotics systems. Therefore, CORA defines three broad entities within its model, which are robot, robot group, and robotics system. The definition of any of the previously mentioned entities is a function of its relation to different terms. For example, a robot is an agent and a device at the same time. The definition of every term is an essential part of the model. Another goal of CORA model is to be extended to fit different contexts. Moreover, the research subject in [FCG+15] is to upgrade the CORA ontology and clarify various ambiguous terms within the model.

The extended CORA model (CORAX) defines new attributes such as position, and autonomy in the form of modes of operations, as shown in Figure 3.28b. Position attribute is meant to describe the surround space of the robot in quantitative (i.e., a point in a specific coordination) and qualitative (i.e., a region with respect to another object) terms. The autonomy attribute is meant to describe the degree of intervention of the human during the HRC. The modes of autonomy as described by CORAX can be fully autonomous, semi-autonomous, tele-operated, remotely controlled, or fully automated. CORAX model extends the goals of CORA model as well, as it aims to provide a guidance for designing the control software component. The goal of this software component is to enhance the information communication within the robotics system, by providing a natural language that ensures the semantic interoperability among the members of a cooperative team.

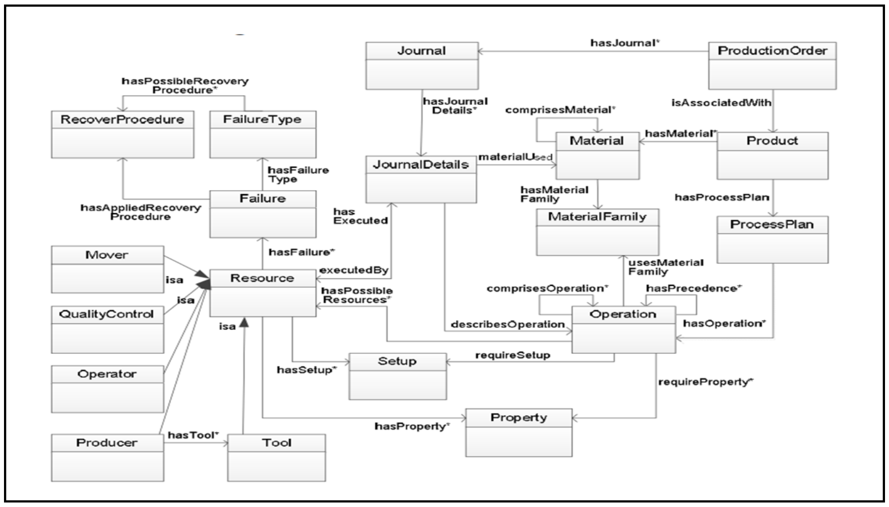


Figure 3.29: [LRT12] GRACE ontology model.

Another ontology model is called GRACE (int**G**gration of p**R**ocess and qu**A**lity **C**ontrol using multi-ag**E**nt technology) has been developed in [LRT12] as shown in Figure 3.29. GRACE model is meant to be a tool for sharing the common knowledge in a distributed manufacturing control system. The manufacturing control system is implemented via JADE technology, which aims to increase the efficiency of the quality control within the production line. The model formalizes the structure of the knowledge related to the operational resources, the product component, and the production operations and history. This formalization is done via dividing the model into three abstract entities, which are concepts, predicts, and attributes. The meaning and the difference among those entities will be discussed in details as a part of the agent ontology in section 4.2.3.

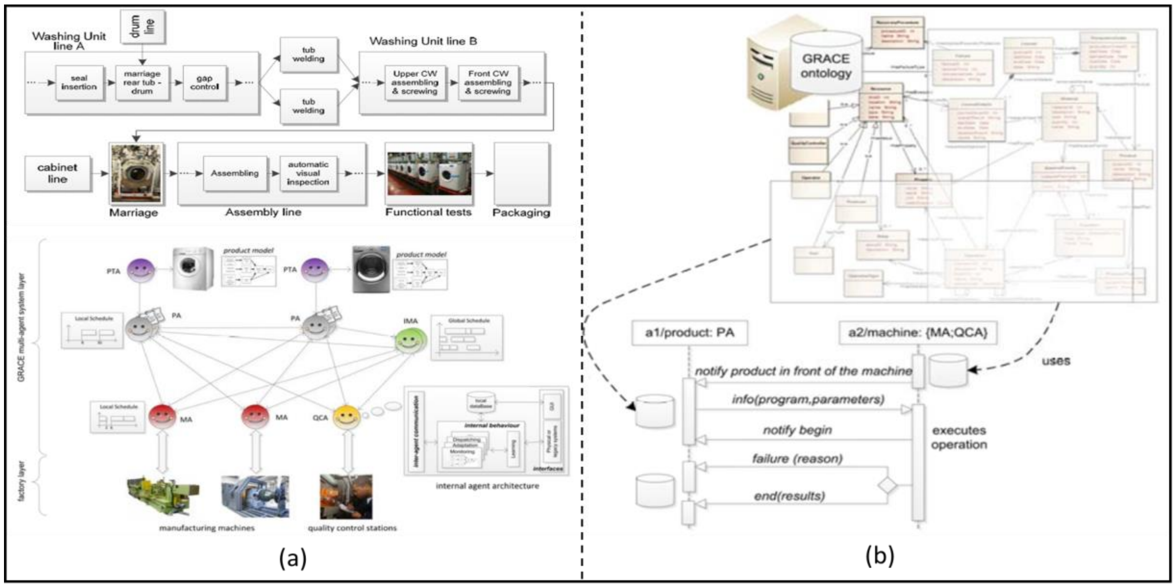


Figure 3.30: [RPL13] (a) Case study model – (b) Implementation model.

[RPL13] uses GRACE ontology model over a washing machine production line that is shown in Figure 3.30a. The washing machine production line has a fixed structure. Therefore, the target of the manufacturing control system is to reduce the production time by eliminating any unnecessary tests during the quality control operation. The manufacturing control system is divided into two layers, the low layer is deploying conventional PLC controllers via IEC 61131-3 framework, while the higher level is deploying an MAS via JADE framework. GRACE ontology terms have been used to build the communication concept among JADE agents, as shown in Figure 3.30b. A scheduler agent has been deployed to plan the flow of the manufacturing operations. This resulted in reducing the production time as one minute over a default of six minutes, which increased the productivity with 20%. From the technical point of view, this research is very useful to show the visibility of deploying an ontology-based MAS over conventional PLC controllers. Although, the integration between the two technologies as well as the details of the PLCs are not mentioned during this article.

3.4 *Summary*

During this literature review, many different subjects that are related to the cooperative manufacturing and the HCA have been reviewed. The first section of the literature review showed the progress in the safety technology in the HRC field in general. Although the research is still going on in the field of HRC safety, the field is advanced enough to be standardized by the industrial robotics standards such as ISO 10218 and ISO/TS 15066, which enabled the idea of cooperative manufacturing.

The second section of the literature review showed that the cooperative manufacturing draws the picture of the factory of the future. Even that the implementation of the cooperative manufacturing ideas is still in progress, the analysis showed that it holds a great potential and benefits for nowadays industry. The benefits of the cooperative manufacturing can be obtained by using the cobot by the worker as a smart tool. That will not only increase the overall productivity, but also will unlock many ergonomic benefits to the worker, such as avoiding physically demanding, repetitive, or hazard tasks.

The third section of the literature review showed the challenges that restrain the cooperative manufacturing. Those challenges can all be answered by the manufacturing control solution. This is because the manufacturing control system is the link between the worker and the cobot. Thus, without that link, the cobot losses its value as a smart tool. The cooperative tasks planning and scheduling plays the main role in increasing the production efficiency. Moreover, the cooperative manufacturing approach fits in a specific production context that stands between the fully manual and the fully automatic. Thus, the manufacturing control system is the only solution to switch from one manufacturing approach to another based on the current need. Many researches proposed different technologies and methods to implement the cooperative workcell control system. However, the gap between the conceptual and the implementation approaches is clear. Although the cooperative workcell is a direct application of the distributed manufacturing system, the current research did not focus on reviewing the previous conceptual solutions of similar problems such as the reconfigurable and flexible manufacturing system. Instead, the current research focuses on implementing a variety of technology such as ROS. Nevertheless, this implementation will be obsolete if it is not supported by a valid conceptual frame.

Therefore, the last section of this literature review has spotted the previous work that addresses the distributed control solution for similar problems to the cooperative manufacturing. The review showed that the HCA is the most known and common conceptual solution. Thus, the review focused on the technologies that have been used to implement the holon component. Different technologies such as IEC 61499 FB and autonomous software agents have been used during the implementation of the holon. Some researchers proposed to combine both the technologies in one software component to obtain both of their advantages. Other researches proposed to separate the holon logic from its communication. The review also showed the feasibility to integrate the mentioned technologies into the present conventional automation infrastructure such as PLCs, DCS, and SCADA systems. Finally, the review presented the idea to implement an ontology-based model via the HCA, to provide complex interaction scenarios that include all the production details. In sum, there is no obligation to use a specific standard technology to implement the HCA. However, the implementation model and technologies must fit the requirements of the applications.

3.5 Conclusion

The conclusion of the literature review can be summarized as follows:

- Although that the worker's physical and psychological safety is an ongoing subject of research, the available safety methods and techniques are mature enough to provide a complete safety for the worker. Therefore, the next question after ensuring the worker's safety is to integrate him with the safe robot in one cooperative workcell. Otherwise, the full potential of the safe robot can never be obtained.
- In practice, the worker trusts in the safety of the cobot, but he cannot see it as a smart or adaptive tool yet. Due to the lack of the proper manufacturing control system, which can provide a common mean of understanding between the worker and the cobot.
- The misconception of the cobot function in manufacturing influences its acceptance on the shop floor. This originated from the idea that the cobot is a replacement of the worker.

However, in reality the cobot is designed to perform light tasks that are physically demanding and harmful for the worker, such as maintaining certain postures or positions for long time. Delegating these hazard tasks to the cobot, will maximize the performance and the quality of the worker's tasks.

- The cooperative manufacturing fills a specific gap in the production paradigms, which is based on the required production volume and customizability.
- Most of the cooperative manufacturing control system research is focused on the system implementation, without a backbone concept. The existing research in cooperative manufacturing field escaped the similarity between the cooperative workcell and the flexible or reconfigurable workcell problem. Therefore, the existing research did not realize that there is a developed conceptual solution, which is the HCA. Instead, most of the present research focuses on the technology. Regardless the fact that the technology is changing with the time. Therefore, any solution that is built upon the current technology without a conceptual background would be obsolete in the future, when the technology is changed or replaced.
- The interaction method between the worker and the manufacturing control system must consider the nature of cooperative manufacturing. Many interaction methods such as body gestures can theoretically obtain this interaction, but in practice, it is an inadequate because it is effort and time consuming and do not fit the industrial environment.
- The cooperative workcell representation from the information point of view is the key to achieve a successful cooperation.
- The holon implementation model is not clear within the holonic control research field. The implementation of the holon model differs from one research to another. IEC 61499 FB and autonomous agent technologies are commonly used among the researcher to implement the holon. Some researches propose to combine between them, to obtain the advantages of both of them. Other technologies can implement the holon as well, such as the industrial web services and ROS, but not commonly applied in this field of research. Accordingly, the selection of a specific technology to implement the holon model must be based on the nature of the problem.
- The HCA is successfully solving the adaptability and the self-organising problems, which are common challenges in flexible and reconfigurable workcells. Therefore, it can solve the same challenges in cooperative workcell.
- Conceptual domain models based on ontologies such as CORA, CORAX, and GRACE have been implemented as a part of the holonic solution. The main reason of using an ontology model is to represent a sophisticated manufacturing environment, which gives more capabilities to adapt with the production variations.

Chapter 4 - Preliminaries: Holonic Software Component

"The Shortest Distance between Two Points is the Straight Line."

Archimedes

4.1 *Holon Implementation Model*

The literature review showed different technologies that are able to implement the holon as a software component. Mainly two technologies have been commonly used to implement the holon, which are the IEC 61499 FB and the autonomous agent technologies. This dissertation extends the implementation model in the literature review to fit the nature of the cooperative manufacturing. This extension follows the modified holon structural model that has been developed in section 2.2.2, as shown in Figure 4.1.

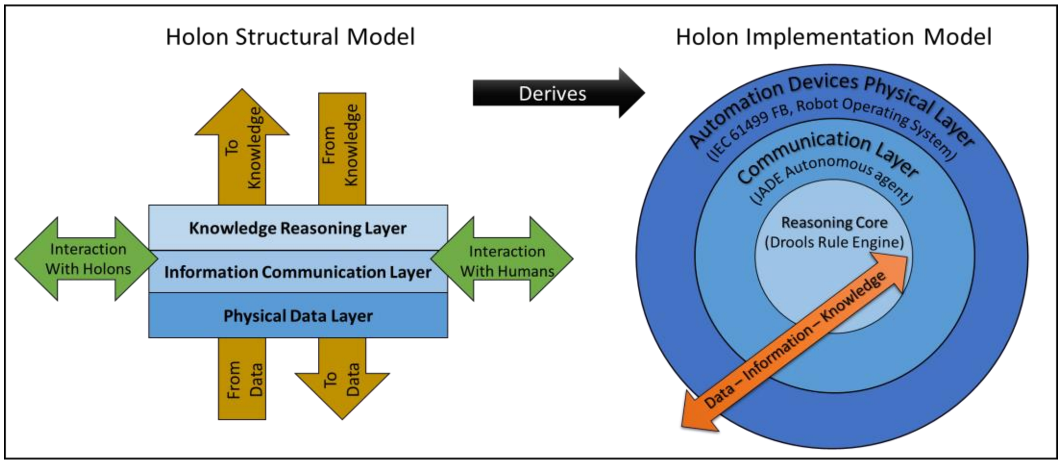


Figure 4.1: Holon structural model derives its implementation model.

The implementation model introduces the holon as a software component with three layers. The first layer is physical and must guarantee the hardware compatibility. Therefore, the HCA is visible to be deployed over a real hardware. IEC 61499 FB supports the compatibility with the distributed industrial controllers such as **P**rogrammable **A**utomation **C**ontrollers (**PAC**s) [IEC16, Nat16]. Therefore, the ORHs that represent either the workers or the machines can use IEC 61499 FB as a physical layer. Furthermore, most of the cobots are operating via a middleware that is called **R**obot **O**perating **S**ystem (i.e., **ROS**) [Ros16]. ROS is a mature standard framework that solves the compatibility problem of the robot hardware. Therefore, ROS will be used to represent the physical layer of the robot. The communication layer at the holon implementation model will be deployed by JADE agent technology. JADE does not only apply the **F**oundation for **I**ntelligent **P**hysical **A**gent (**FIPA**) standard that ensures the software interoperability and extensibility [JAD15, Fip15], but also it provides a reliable communication means via the **FIPA** - **A**gent **C**ommunication **L**anguage (**FIPA-ACL**). FIPA-ACL is more flexible and more expressive than the publish/subscribe communication pattern that is often used by IEC 61499 FB and ROS. Furthermore, JADE agent can interact via an ontology-based model that enables the knowledge construction. The high communication capabilities of JADE are results from the fact that JADE is built upon a reactive oriented agent architecture. However, when it comes to reasoning, a deliberative oriented agent architecture provides more fast, reliable, and robust approach to solve a problem [RN16, BHW07]. This is the main reason to choose Drools RE to represent the third layer of the holon implementation model [Dro16]. Thus, Drools RE separates the holon logic and algorithms from its communication and physical interface.

4.1.1 Physical Data Layer

Holon's physical layer is the direct interface that links between the holon and the human or the machine. From the literature review, it has been found that IEC 61499 FB and ROS are the most proper technologies to implement this layer in the context of cooperative manufacturing. In order to understand the reasons for selecting those two technologies, they will be studied in details in the following sections.

4.1.1.1 IEC 61499 Function Blocks

IEC 61499 standard is an implementation model for the distributed control systems on embedded devices. It extends the capabilities of IEC 1131-3 that is the main standard to manufacture the PLCs. IEC 61499 defines a reference model to develop, reuse and deploy an FB over distributed embedded industrial controllers. The decision process in IEC 61499 standard is associated with the application. Therefore, the decision process does not run under a single processor, as it is divided among several processors. However, in order to control an application, these processors must exchange data with each other, as illustrated in the example in Figure 4.2a. In this example, the processing of an application-x is distributed over three processors, which are A, B, and C. Publishing/subscribing communication pattern is followed by IEC 61499 to achieve its distributability. Publishing/subscribing is a communication pattern where a publisher FB sends a non-programmed message to a subscriber FB. Publishing/subscribing is a primitive and unreliable communication approach that does not guarantee the message delivery. Nevertheless, it is very useful in streaming the sensors data. Inasmuch as, it is not a big problem if some of those data are lost during the communication.

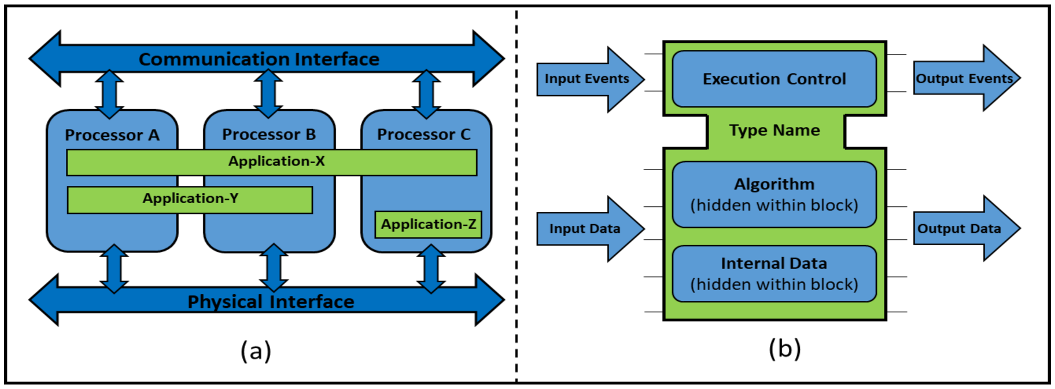


Figure 4.2: (a) IEC 61499 distributability - an example– (b) IEC 61499 FB model.

An FB is composed of three fields as shown in Figure 4.2b. These fields are events I/O, data I/O, and an internal algorithm. No scan cycle is involved to trigger the algorithm execution. Instead, the algorithm execution is initiated by the arrival of events. Thus, the algorithm uses the current data values that are available when the event occurs. The change in the data value is considered an input event as well. There are three FB types: basic FB, composite FB, and service interface FB. A basic FB executes an elemental control function, such as reading a sensor or setting the state of an actuator. A basic FB can be customized to execute a specific logic. Furthermore, a composite FB can be developed by combining a group of basic FBs. A composite FB can be reused to perform a high-level control function. Finally, a service interface FB provides the communication services among devices such as publishing or subscribing. Adding an interface FB to provide a communication service to another FB can make the solution rather complicated. Thus, the communication is a weakness point in IEC 61499 FB. Accordingly, IEC 61499 FB will be used only to customize the ORHs physical interface that ensures the hardware compatibility and distributablity.

Many software development frameworks are based on IEC 61499 standard. Examples of those frameworks are FBDK [Hol16], nxtStudio [nxt16], and 4DIAC [4di16]. Those frameworks differ in minor variations such as their programming style. This dissertation is using FBDK as part of the implementation and testing phase. FBDK is not only an IEC 61499 development framework, but also it is used to simulate the manufacturing system in the real-time before the deployment phase.

4.1.1.2 Robot Operating System

ROS is a flexible middleware that is widely used to develop robotics software. It is a collection of open source tools and libraries, which simplify and standardize the development of robust and complex robotic control solutions [Kan16]. ROS provides many different services in the same manner that a computer **O**perating **S**ystem (**OS**) such as Windows or Linux provide to the user. Including the hardware standardization, low-level physical control, common functions abstraction, and the communication patterns. It also provides tools to obtain, build, write, and run a code across different OSs. ROS does not replace, but instead works on top of a computer OS.

In 2007, ROS has been introduced as a part of a Stanford AI Robot project. Subsequently, many well-known robotics institutes and firms joined the project. The purpose of the project was to standardize and facilitate the development of a robot software. Before ROS, every robotics software developer would start from scratch to create the robot software, as well as the hardware itself. This is a very demanding task, as it requires an excellent knowledge of many different fields such as mechanical, electrical, electronic, plus computer science. The main idea of ROS is to avoid wasting all the time and effort, and to offer standardised functionalities performing hardware abstraction, just like a conventional computer OS. This could be obtained by sharing the expertise from different robotics software developers, whom are all using ROS to solve a variety of common problems in robotics [MF13].

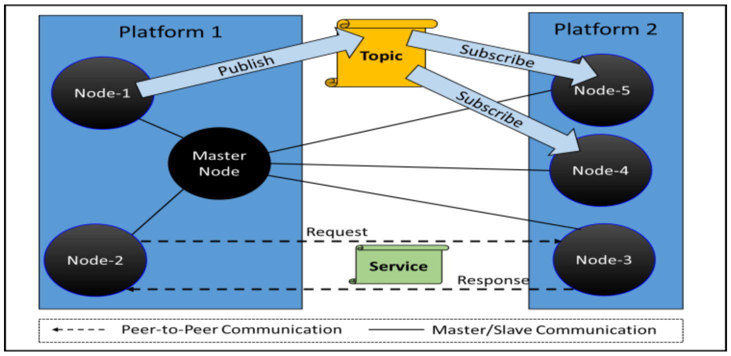


Figure 4.3: ROS centralized topology - an example.

ROS is based on a centralized topology where processing takes place in nodes that may receive or send data, as shown in Figure 4.3. These data could be related to sensors or actuators. The nodes communication is based on the TCPROS that is similar to TCP/IP protocol. The communication is administrated by a master node that handles all the connections and addressing details. The message exchange among the nodes usually occurs asynchronously. This means that there is no expected timeframe from the receiver to respond. This is a very fast pattern of communication as the receiver can store the messages then read them later. Therefore, the processes run by the receiver are not slowed down. Synchronous communication also can be obtained in ROS if needed by creating application services. Since ROS has been purely designed for robotics purposes, the message structure is very simple, not to overload the nodes with overhead data that slow the robots. ROS has standardized a group of terms to simplify creating nodes and establish a successful communication among them. Those terms are as following:

- **Node**: a computational code that executes some tasks and communicates via the protocols and mechanisms that are available in ROS. A node must register itself to the master node with a unique ID and a list of topics or services. The topics and services are used to send or receive messages and some additional connection parameters. ROS nodes can be written in C++ or Python programming languages.
- **Master**: the master is a special node that is launched automatically after deploying ROS. The master node is responsible for the registration, subscriptions, deregistration of the other nodes, and storing all of the configuration parameters. Only one master node is allowed among all the running ROS instances.
- **Message**: a simple data type that might be String, Boolean, Integer, Timestamp, etc.
- **Topic:** a publish/subscribe pattern that is used to exchange commutation messages. A node uses a topic to send data via a publishing process or receiving data via a subscribing process. A topic handles unlimited amount of publishers/subscribers. Connecting the publisher to the subscriber is the responsibility of the master node and it is obtained via the nodes IDs. Then, a P2P connection is obtained by the nodes.
- **Service**: alternative to a topic and replaces the publish/subscribe message exchange pattern by a request/response pattern. A node that uses a service only receives data in response to a query it made. The response may vary based on the information given in the request. A service is provided by only one node and only one request can be sent at the same time. The service contains a description of both the request and the response messages type.
- **Bags**: data logs to store and play back the messages, which are very important for collecting data measured by sensors and subsequently play it back as many times as desired. In addition, it is very useful for the system debugging.

From the communication perspective, ROS has two crucial points of weakness in comparison with JADE. First, ROS is built upon a central topology that makes it very fragile if the master node is crashed. In another words, the whole system fails with the master node. However, in JADE a similar concept of a master node exists but within every platform. This means that if one platform fails, the rest of the system can tolerate this fault. Second, ROS is similar to IEC 61499 standard when it comes to the communication complexity. Both cannot provide highly sophisticated communication scenarios, as they use a simple message structure that can hold basic data type. This makes them very fast and acceptable over the physical layer but inadequate over the communication layer which deals with information and knowledge. Yet, a sophisticated ontology-based communication patterns can be achieved via JADE, as will be seen in details in the next section and in section 4.2.

4.1.2 Information Communication Layer

The information communication layer is the core of the holon. Through this layer, the holons are able to exchange the information among themselves and the human being. The autonomous software agent provides high communication capabilities, thus it is selected to implement this layer. A software agent is a computer system situated in a specific environment that is capable of performing autonomous actions in its environment in order to meet its design objective [JW98]. An agent is autonomous by nature, this means that an agent operates without a direct intervention of the humans and has a high degree of controlling its actions and internal states. In order to achieve this autonomy, an agent must be able to fulfil the following characteristics [SHY+06]:

- **Autonomous**: operates without a direct intervention of the humans. Inasmuch as it has a high degree of controlling its external actions and internal states.
- **Social**: can interact with other autonomous agents or humans within its environment in order to solve a problem.
- **Responsive**: capable of perceiving its environment and respond in a timely fashion to the changes occurring in it.
- **Pro-active**: able to exhibit opportunistic, goal directed behaviour and take initiative.

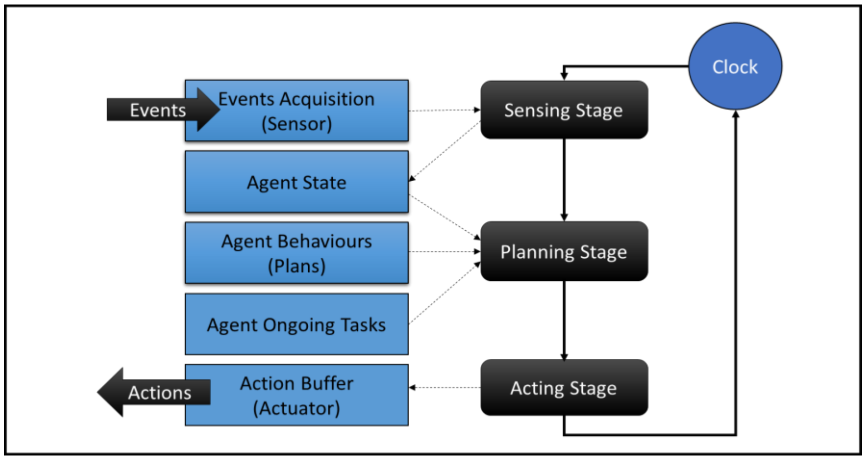


Figure 4.4: Autonomous agent internal execution model [RS11].

Conceptually, an agent is a computing machine that is given a specific problem to solve [Tea10]. Therefore, it chooses a certain set of actions and formulates the proper plans to accomplish an assigned task. The set of actions that are available to be performed by the agent are called an agent behaviour. The agent behaviours are mainly created by its programmer. An agent can execute one or more behaviours to reach its target. The selection of an execution behaviour among others is based on certain criteria that are defined by the agent programmer. Building an execution plan is highly dependent on the information that the agent infers from its environment and the other agents. An MAS is a collective system that contains a group of autonomous agents, teaming together in a flexible distributed topology, to solve a problem beyond the capabilities of a single agent [WJ95]. Figure 4.4 illustrates a generic conceptual model of an autonomous agent. The conceptual model combines the event driven and the task driven control architectures. The model defines the typical agent execution cycle. An agent execution cycle is composed of three stages [RS11]:

- **Sensing**: in this stage, the internal state of the agent is updated with the events that are collected via the sensors. These events are originated from the process or the environment that the agent is involved in.
- **Planning**: in this stage, the execution actions are selected based on the value of the input event, the current state of the agent, and the agent task or sub-task.
- **Acting**: in this stage, the selected actions are executed, and the agent's internal state is updated.

A software agent can be based on one of the following five models [CGA+14]:

1- **Pure deliberative (logic-based or symbolic-based):** the earliest model to develop an agent. The agent presents the world and its behaviours based on a symbolic knowledge base. The reasoning of the deliberative agent is based on applying the common symbolic computational logic.

2- **Pure reactive:** reactive agent does not have a specific symbolic model of the world. Instead, it directly maps an action to a situation. Reactive agent is mainly using a black box approach, to respond to the changes that happen to the world regardless of the world model. Thus, it does not utilize the symbolic logic approach.

3- **Belief-Desire-Intention (BDI):** probably the most known agent model, which can be seen as a philosophical theory rather than a logical model. In a BDI model, an agent consists of three logical terms. Beliefs are a set of facts that reflect the perception of the world. Desires are a set of goals and tasks that can be carried out by the agent. Intentions are a set of plans and behaviours that associate the beliefs with the desires.

4- **Layered:** a hybrid between the deliberative and the reactive agent models, it combines their advantages and at the same time alleviates their disadvantages. This model is very important, as it will be used via this dissertation.

5- **Cognitive:** the most recent agent model that tries to mimic the basic human behaviours by learning them via monitoring the human actions. Cognitive agent model is still a developing field as it depends on the advanced AI methods such as image and sound recognitions.

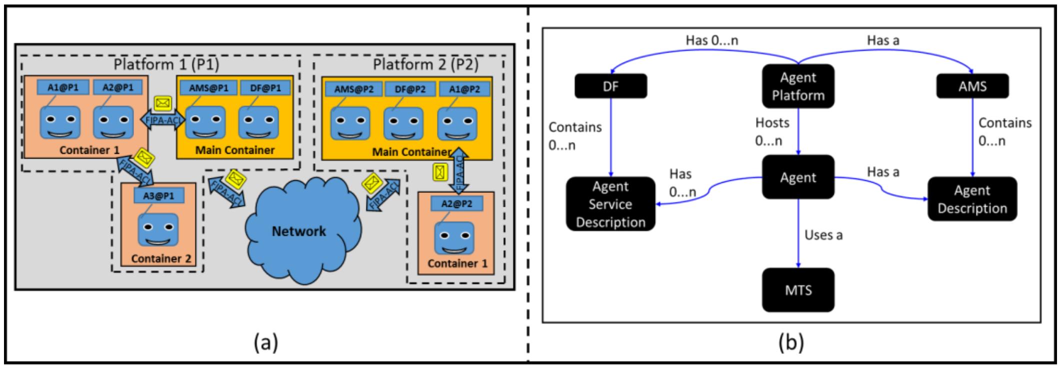


Figure 4.5: (a) JADE distributability - an example – (b) JADE management model.

JADE is a distributed MAS middleware [BCG07]. JADE cannot be described as a pure reactive agent as it provides a method to model the environment via an ontology-based representation. Nevertheless, it does not normally use RE to process the facts. The reason to select JADE as the holon interaction component is briefly the standardization of its communication language. Inasmuch as it supports the agent communication via using FIPA specifications, and provides a graphical interface to deploy and debug the MAS. FIPA is an IEEE standard organization that promotes the agent technology and its interoperability with other technologies. JADE agents use FIPA-ACL to exchange messages either inside its own platform or with another platform in a distributed MAS as shown in Figure 4.5a.

JADE management model can be seen in Figure 4.5b. Every agent system runs under an independent platform. An **A**gent **P**latform (**AP**) provides a physical infrastructure where the agents are deployed. The AP consists of the machine hardware, the OS, FIPA agent management components, and the agents themselves. Every AP must contain an **A**gent **M**anagement **S**ystem (**AMS**) agent. AMS is responsible for managing operations such as creating or deleting an agent. Each agent must register itself with an AMS in order to obtain an **A**gent **ID**entifier (**AID**). An agent AID is addressed by the AMS in a directory of all agents present in the AP. The agent must deregister from the AP to terminate its life cycle. After deregistration, the agent AID is vanished from the AMS directory and can be owned by another agent. Only a single AMS is allowed within an AP even if the AP contains multiple machines. The **D**irectory **F**acilitator (**DF**) is an optional component of an AP that provides yellow pages services to the other agents. Every agent that wishes to publish its services to other agents must contact the DF and request the registration of its agent description. Any agent can deregister or request to modify its description from the DF at any time and for any reason. In addition, an agent may ask the DF to issue a search for another agent with specific criteria within the AP.

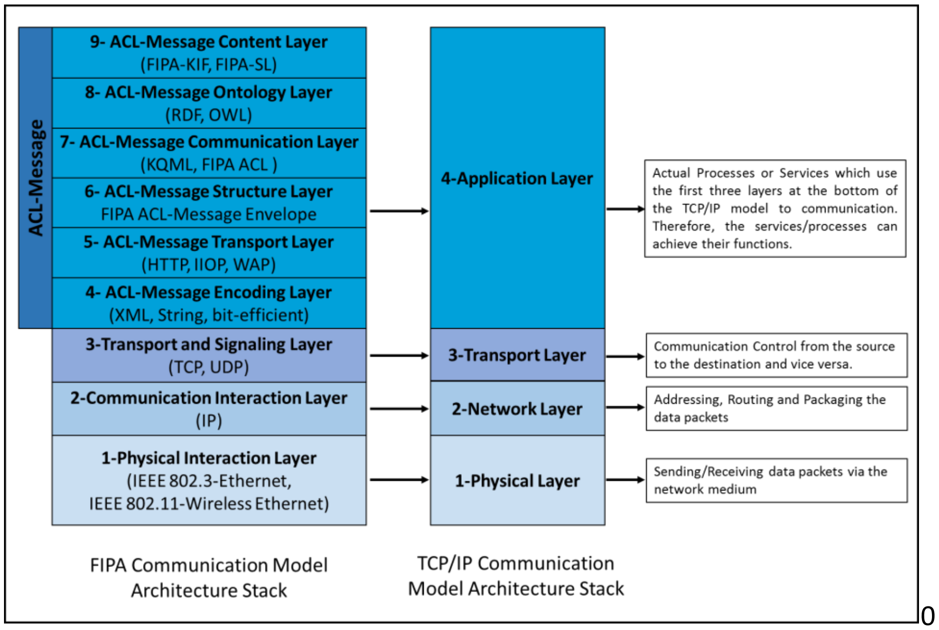

ACL-Message
9- ACL-Message Content Layer (FIPA-KIF, FIPA-SL)
8- ACL-Message Ontology Layer (RDF, OWL)
7- ACL-Message Communication Layer (KQML, FIPA ACL)
6- ACL-Message Structure Layer FIPA ACL-Message Envelope
5- ACL-Message Transport Layer (HTTP, IIOP, WAP)
4- ACL-Message Encoding Layer (XML, String, bit-efficient)
3-Transport and Signaling Layer (TCP, UDP)
2-Communication Interaction Layer (IP)
1-Physical Interaction Layer (IEEE 802.3-Ethernet, IEEE 802.11-Wireless Ethernet)
4-Application Layer
3-Transport Layer
2-Network Layer
1-Physical Layer
Actual Processes or Services which use the first three layers at the bottom of the TCP/IP model to communication. Therefore, the services/processes can achieve their functions.
Communication Control from the source to the destination and vice versa.
Addressing, Routing and Packaging the data packets
Sending/Receiving data packets via the network medium
FIPA Communication Model Architecture Stack
TCP/IP Communication Model Architecture Stack
0

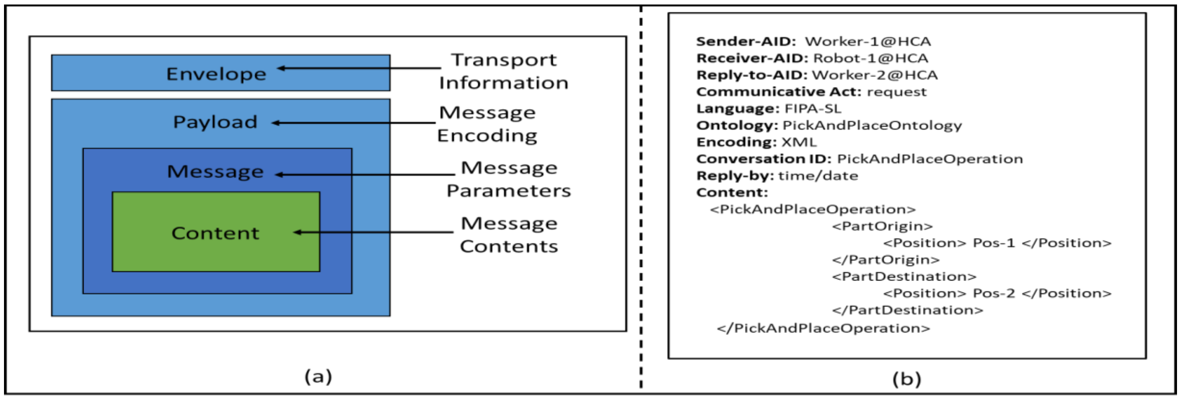
Figure 4.6: An analogy between FIPA and TCP/IP communication models.

In order to achieve the communication process among JADE agents, JADE agent must follow FIPA communication model that is very similar to the TCP/IP model [NSR11]. Generally, a communication architecture model is a stack of layers; each layer specifies a set of data flow, information management, and communication control standard protocols. The purpose of a communication architecture model is to exchange the services and the information among the computing machines within a wired or a wireless network [HC01]. FIPA communication model has nine layers in total. The first three lower layers are exactly the same as in the TCP/IP model, while the upper six layers equal together to the application layer in the TCP/IP model. A brief description of FIPA communication model layers can be seen as follows:

1- **Physical interaction layer:** it is linked directly to the actual network medium that is a part of the machine hardware. Standard IEEE 802.3-Ethernet and IEEE 802.11-Wireless Ethernet protocols are responsible for sending/receiving the network data packets.

2- **Communication interaction layer:** it uses the **I**nternet **P**rotocol (**IP**) to pack/unpack the data into/from the IP datagrams. In addition, it provides the right network rout and IP-address for the datagrams.

3- **Transport layer:** it makes sure that the IP datagrams are delivered from the source to the destination and vice-versa. TCP and UPD are used in this layer as communication control protocol. TCP deploys a fixed length datagram via P2P connection with a reliable acknowledgment of packets delivery. While, UDP deploys a variable length datagram via P2P or broadcasting connection with no need of packets delivery acknowledgment.



```
Sender-AID: Worker-1@HCA
Receiver-AID: Robot-1@HCA
Reply-to-AID: Worker-2@HCA
Communicative Act: request
Language: FIPA-SL
Ontology: PickAndPlaceOntology
Encoding: XML
Conversation ID: PickAndPlaceOperation
Reply-by: time/date
Content:
  <PickAndPlaceOperation>
                  <PartOrigin>
                         <Position> Pos-1 </Position>
                  </PartOrigin>
                  <PartDestination>
                         <Position> Pos-2 </Position>
                  </PartDestination>
   </PickAndPlaceOperation>
```

Figure 4.7: (a) an ACL-Message structure [Fip15] – (b) an ACL-Message example.

4- **FIPA ACL-Message encoding layer:** it specifies the message presentation standard format. An XML format can be used to represent the message content. XML has been invited to store data in a structured text style to describe electronic documents. XML is extensible as it is mainly built from customized elements, which can be flexibly renamed or edited. The simplicity in building an XML document gave it the property of being an OS independent language. Other FIPA standard format such as String and bit-efficient can be also used to represent and ACL-Message.

5- **FIPA ACL-Message transport layer: M**essage **T**ransport **L**ayer (**MTP**) is a standard application protocol that deals with web applications such as **H**ypertext **T**ransfer **P**rotocol (**HTTP**), **I**nternet **I**nter-**O**rb **P**rotocol (**IIOP**), and **W**ireless **A**pplication **P**rotocol (**WAP**). These protocols facilitate the distributed collaborative information exchange via request/response interaction in client/server computer model.

6- **FIPA ACL-Message structure layer:** an ACL-Message has a standard structure that is necessary to encode the message. An ACL-Message structure depends on the upper six layers of FIPA communication model. A schematic of an ACL-Message structure can be seen in Figure 4.7a, and an example of the message can be seen in Figure 4.7b. The fields of an ACL-Message are coded via the agent programmer and they vary due to the application and implementation approach.

Table 4.1: FIPA ACL communicative-acts adapted from [GS07b].

Communication Act	Purpose
propose, accept-proposal, reject-proposal, cfp	negotiation
request, request-when, query-if, query-ref	requesting information
confirm, disconfirm, inform, inform-if, inform-ref	passing information
agree, refuse, cancel, subscribe	performing actions
not-understood, failure	error handling
propagate, proxy	message referencing

7- **FIPA ACL-Message communication layer:** it is very important as it defines a set of terms that are mandatory to build an ACL-Message. FIPA-ACL is the official FIPA standard communication language that extends the features of **K**nowledge **Q**uery and **M**anipulation **L**anguage (**KQML**). For example, the communicative-act is a mandatory field that has standard values that are predefined by the FIPA. The value of a commutation act can be seen as a message type that expresses the purpose of the message, as shown in Table 4.1. The conversation-ID field is mandatory field as well. However, its value is up to the agent coder. The conversation-ID is very important concept to track a thread of messages in a specific conversation during the agents interaction.

8- **FIPA ACL-Message ontology layer:** an ACL-Message is flexible to afford different levels of conversation complexity. A String based massage is the simplest form of conversation among FIPA agents. However, in complex conversation scenarios, an ontology-based content would be the proper conversation method. In an ontology-based ACL-Message, some specific ontology languages are used to structure the message. FIPA supports **R**esource **D**escription **F**ramework (**RDF**), and **W**eb **O**ntology **L**anguage (**OWL**) that are the most common languages to code an ontology. These languages along with the ontology concept will extensively discussed as a part of section 4.2.

9- **FIPA ACL-Message content layer:** FIPA goes beyond the conceptual modelling via an ontology. An ontology model is often used by an agent to represent its own environment. However, agents are meant to exchange and process information in order to achieve the MAS cooperation. Accordingly, reasoning an ontology-based message is an essential task of an autonomous agent. Therefore, FIPA supports logic-based ontology model via two languages, which are FIPA **K**nowledge **I**nterchange **F**ormat (**FIPA-KIF**), and **FIPA S**emantic **L**anguage (**FIPA-SL**). FIPA-KIF and FIPA-SL languages allow modelling the system knowledge based on the first order logic that commonly used in AI. Therefore, using a logic based ontology model enables an RE to process and analysis the knowledge of this model. The knowledge reasoning will be explained in details in section 4.1.3.

From the software perspective, an agent is an extension of an object. Therefore, an agent is able to implement the main concepts of **O**bject-**O**riented **P**rogramming (**OOP**), which are abstraction, encapsulation, inheritance, and polymorphism. Thus, it enriches the OOP by adding new concepts such as the agent distributability, communication, behaviours, and ontology-based interaction. These new enrichments offer a new level of intelligence, which promotes the autonomous agent to operate autonomously and may even learn from its environment. Therefore, an agent can take decisions not only based on the environment input, but also the history of the I/O and the changes in the environment model.

4.1.3 Knowledge Reasoning Layer

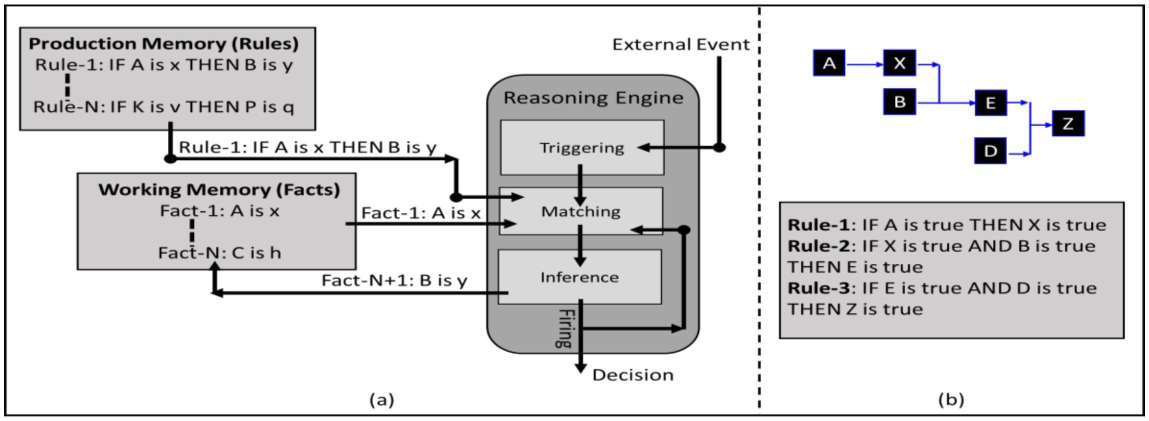

Production Memory (Rules)
Rule-1: IF A is x THEN B is y
Rule-N: IF K is v THEN P is q
Rule-1: IF A is x THEN B is y
Working Memory (Facts)
Fact-1: A is x
Fact-N: C is h
Fact-1: A is x
Fact-N+1: B is y
External Event
Reasoning Engine
Triggering
Matching
Inference
Firing
Decision
(a)
A
X
B
E
D
Z
Rule-1: IF A is true THEN X is true
Rule-2: IF X is true AND B is true THEN E is true
Rule-3: IF E is true AND D is true THEN Z is true
(b)

Figure 4.8: Rule-based system (a) Components and operating mechanism – (b) An example.

The holon knowledge reasoning layer is needed to reason the exchanged information among the holons and the new information from the holon's environment. For this reasoning, a reasoning system is required to implement this layer. Reasoning system is a software framework that generates conclusions based on the available knowledge. RE is one of the most common methods to implement a reasoning system. An RE is often a part of a rule-based system and is programmed via a set of rules that present the reasoning algorithm. A rule-based system is a form of an expert software that uses an ontology-based representation to codify the facts into a knowledge base which can be used for reasoning [OEO+16]. In a rule-based system, an RE interacts with two kinds of memory, as shown in Figure 4.8a. The working memory holds the facts that represent the domain knowledge, while the production memory holds a set of rules. A rule is a declarative method to represent the knowledge in a concise, non-ambiguous conditional statement. The RE is the kernel of the rule system that solves a given problem by matching the present facts with the existing rules. Initially, an external event must trigger a reasoning session. When one or more facts match the IF part in one or more rule statements, the THEN part is inferred and new facts are generated. The new facts will be updated into the working memory. New reasoning session will be triggered automatically, every time a match between a fact and a rule occurs. A chain of reasoning sessions will take place until no match is fulfilled. An example of a reasoning chain can be seen in Figure 4.8b.

The RE organizes and controls the flow of the solution route based on one of the following reasoning methods [Ajl15]:

- **Forward reasoning**: in forward chaining, the RE starts with the initial facts and derives new facts whenever a rule matches a fact. The RE goes through consequence of rules firing to find its final goal, which ultimately leads to finding the solution route from the initial facts to the final goal in a forward route.

- **Backward reasoning**: in backward chaining, the RE starts from the final goal, then it finds which rules must be fired to lead to this goal, therefore the RE can determine the facts that are needed to reach the final goal. During the chaining, a consequence of sub-goals appear which ultimately leads to finding the solution route from the final goal to the initial facts in a backward route.

Drools is an expert system that is created by jboss organization, to extend the features of a regular rule-based system [Doc17]. It provides a tool for a rule-based solution creation, managing, deploying, and analysing. Drools RE can apply a hybrid chaining reasoning. A hybrid chaining reasoning is a mix between the forward and the backward chaining and can be more efficient in some cases than both. Drools RE may apply two techniques in a reasoning session. The first is called a stateless reasoning session. In this technique. Drools RE does not take into consideration the change in the values of the facts during executing the reasoning session. Therefore, the rules that are based on these facts will not be fired. The second is called a stateful reasoning session. In this technique, Drools RE takes into consideration the change in the values of the facts during executing the reasoning session. Therefore, the rules that are associated with those facts are fired. The stateless reasoning session is used to derive new results based on new facts, while stateful reasoning session is useful in more complicated scenarios such as diagnosis or monitoring. Moreover, Drools RE extends the Rete algorithm for pattern matching that is called ReteOO. ReteOO denotes the enhancement of Rete algorithm by combining it with the other OOP concepts, as shown in Figure 4.9. In Figure 4.9, the attributes of an instance from object-A matches the conditional part of the rule. Thus, the rule creates a new instance from object-B, which holds attributes that are assigned by the rule conclusion part.

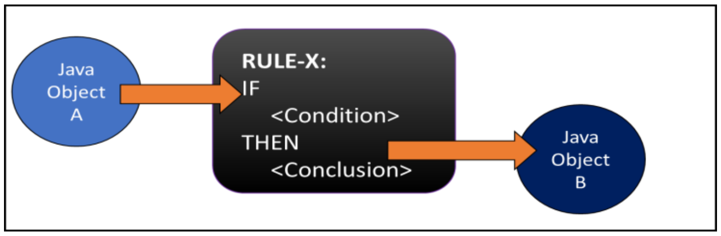


Figure 4.9: Drools ReteOO algorithm for pattern matching.

Drools RE implements a deliberative oriented agent architecture that is written as a high-level language than a symbolic language to model the agent world and its reasoning scheme. Moreover, Drools RE affords many other privileges in the following sense [tut16]:

- **Declarative programming**: rules programming is far easier to express complicated solutions of difficult problems, and to verify these solutions as well. Thus, Drools RE provides a decision-making table option in a form of MS-excel, which makes it possible for someone with no coding background to read and modify the rules.

- **Logic and data separation**: as a direct result of the ReteOO algorithm, the facts locate in the domain of the object, while the logic locates in the domain of the rules. This is very useful to provide different levels of authorities and accessibilities for the users.
- **Centralization of the knowledge**: by creating a single facts knowledge base, the knowledge can be accessible and shared by all the agents. Thus, a problem can be solved faster and more efficient.
- **Speed and reliability**: using Drools RE options such as ReteOO, Hybrid reasoning, or stateful reasoning session make Drools RE very fast and reliable tool.

4.2 Holon Ontology-based Interaction

4.2.1 Ontology Concept

The concept of ontology originates from philosophical roots. In philosophy, an ontology aims at studying the nature of the existence and reality and forming tight relations among the beings, whom are part of that reality. The word ontology itself comes from the combining the two Greek words "Onto" which means a being and "Logia" which means a divine origin. Thus, ontology can be translated literally to the origin of the being. However, the concept of ontology has been frequently used in computer science in association with the AI to define the relationships among the software objects. Although, there is no specific definition of the word ontology within the computer science field, the following definitions will draw a complete picture of the ontology meaning [Rod12]:

- An ontology defines the basic terms and relations that comprise the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions for this vocabulary [NFF+91].
- An ontology is a formal, explicit specification of a shared conceptualization [Gru95].
- An ontology is a logical theory that clarifies the intended meaning of a formal vocabulary. "i.e., a commitment to a particular domain conceptualization" [WGP+08].
- An ontology provides the meta-information to describe the data semantics, represents the knowledge, and communicates with various types of entities (e.g., software agents and humans) [Fen02].
- An ontology is a mean that enables the communication and knowledge sharing, by capturing a shared understanding of the terms that are used both by humans and machine software" [Lai07].

The previous definitions of ontology can derive a precise definition of the ontology within the scope of cooperative manufacturing. Thus, an ontology is a conceptual tool to represent and create a common understanding for the entities within the manufacturing workcell (i.e., holons). Furthermore, this common understanding enables to exchange, reuse and extend the manufacturing knowledge.

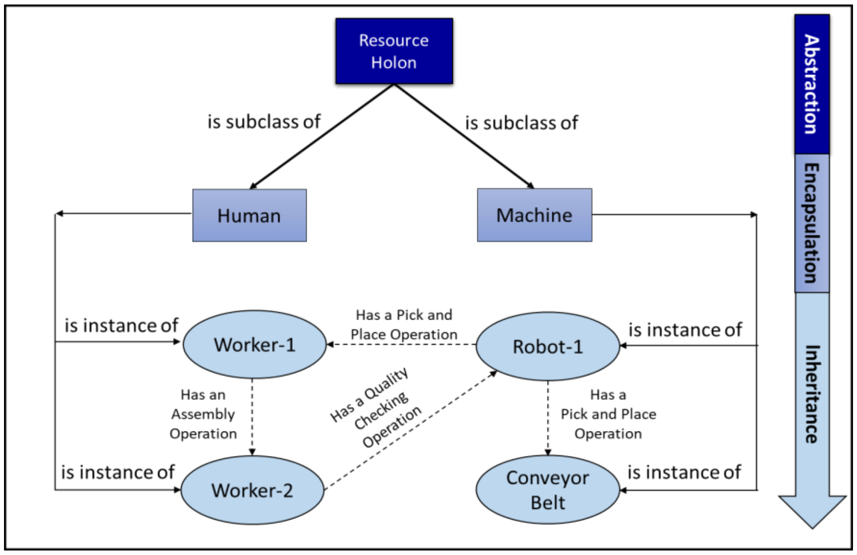


Figure 4.10: Ontology model – an example (ORHs scenario).

An ontology-based model extends the OOP concepts by defining relations among the objects. Moreover, any ontology-based model must achieve the following OOP concepts:

- **Abstraction:** a generalization process that differs due to the context and the purpose of the code. In another words, abstraction is the process of exposing an object's parameters that are general and can be used by all the other objects within the software domain.
- **Encapsulation:** can be seen as the opposite of abstraction, as it means to hide the object's parameters that can only be used by the object, to fit the context of the software domain.
- **Inheritance:** the idea of having different instances of the same object with different values of their parameters.

Generally, an ontology uses five elements that are shown in Figure 4.10 and discussed below to describe a specific domain:

- **Class/Sub-class:** an element that represents an abstraction of an object. The classes are the first elements to construct when representing a specific domain as all the four other ontology elements are essentially depending on the class elements. In the resource holons ontology model that is shown in Figure 4.10, there is only one main class that represents the resource holons. Other subclasses such as human or machine can extend this class to add more specific or private attributes from the subclass.

- **Attribute:** a basic data type that describes a characteristic of a class. For example, in the resource holons domain, a String data type may describe an ID attribute.
- **Instance**: an element that specifies a certain instance of a class with specific attributes values. For example, worker-1 and worker-2 in the shown example must have two different ID values.
- **Relation**: an element that expresses a relation between two instances. For example, robot-1 has a pick and place operation which can be invoked either to worker-1 or to the conveyor belt.
- **Axiom**: a logical rule in the domain. For example, the cooperative workcell can handle two workers and one robot as maximum capacity.

As shown in the previous example, there is no standard method to build an ontology. The shown model could be built based on the type of operational resource (i.e. worker, robot, etc.). The level of ontology abstraction and classification is simply a matter of design ratio that serves the designer's objective. However, an ontology design guide has been proposed by Noy and McGuninness in [NM01]. In this approach, the following seven-step process is followed to build an ontology model:

- Step 1: determine the domain scope, coverage, and usability.
- Step 2: considering the possibilities of reusing/modifying previous ontologies from a similar domain.
- Step 3: clustering all the important terms into groups.
- Step 4: mapping the relations among the classes and constructing a hierarchal structure.
- Step 5: assigning attributes to the classes.
- Step 6: defining the axioms and the restrictions within the hierarchal structure.
- Step 7: creation of individual instances of the classes.

The idea of using an ontology-based model is to achieve a set of privileges that are essential, particularly in the context of cooperative manufacturing. These privileges are briefly mentioned as follows:

- **Common Understanding**: sharing common understanding of the information structure among humans and/or software agents is counted as the most important privilege of an ontology model. This characteristic has been illustrated in details in the example in Figure 4.10. This is achieved by providing a common language, which can be understood by the human and simultaneously by the software agent. Accordingly, different forms of interaction and cooperation are fulfilled.
- **Reusability**: model reusability is one of the goals of the ontology developer. Within the same ontology model, the same relation can be reused among the different objects. For example, the relation "has a pick and place operation" is used twice by robot-1 to serve worker-1 and the conveyor belt respectively. Furthermore, over the model scale, the

same model can be reused in a similar domain, or it can be modified to fit another domain or application. For instance, the same model in the example can be applied over another workcell with the exact resource holon, while considering changing the instances attribute values. Additionally, the same model can be modified to fit a workcell with more resource holons.

- **Interoperability:** the interoperability definition has been briefly introduced from the point of view of the manufacturing control during the focus of this dissertation in section 1.3. A more generic definition of interoperability is the ability of two or more systems or components to exchange information and to use this exchanged information [Gha10]. The software interoperability exists in many different levels such as: syntactic, structural, semantic, and pragmatic. The meaning and the difference among these levels of interoperability will be shown in details as a part of the ontology spectrum during the following part of the dissertation. The different levels of interoperability are guaranteed via using the appropriate method and implementation model.

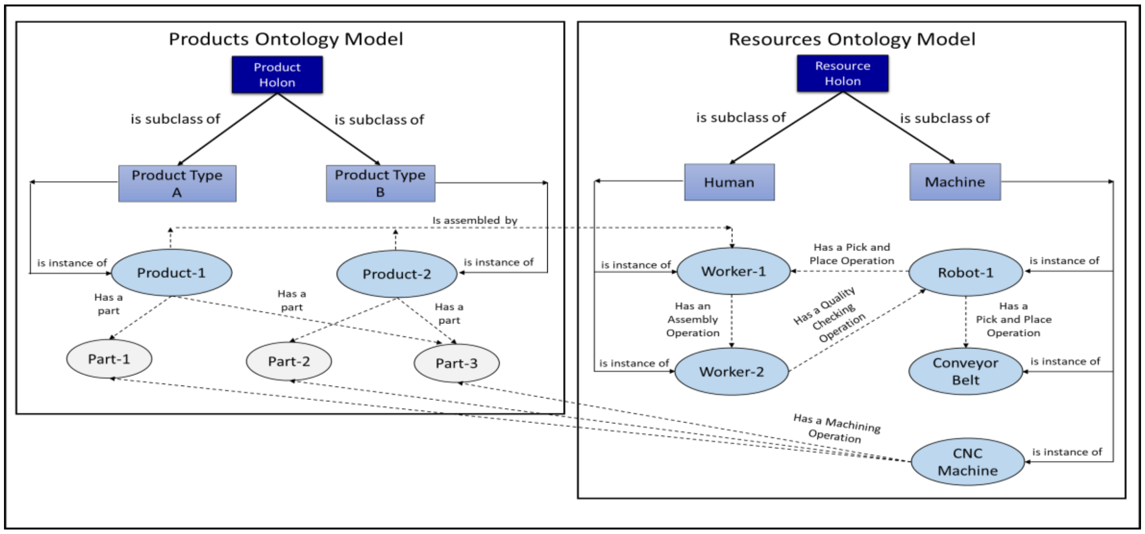


Figure 4.11: Intra/Inter extensibility in ontology – an example.

- **Intra-Extensibility and Inter-Extensibility**: extensibility in ontology is far superior concept than in OOP. Intra-Extensibility of an ontology denotes the idea of extending the same class by new subclasses. For instance, in the example shown in Figure 4.11, a new resource like a CNC machine can be added under the machine class of the model. Inter-Extensibility refers to the idea of merging two or more ontology models that serve or complete each other. For example, two different ontology models have been combined together in Figure 4.11, the first model addresses the structure and functionality of the resource holons, while the second ontology model describes the products' details that can be manufactured by the different resources at the cooperative workcell.

- **Separating the domain knowledge from the operational knowledge**: is a direct result of the extensibility. As mentioned before, two ontologies models are separated but they can be merged together. One model can hold the domain knowledge such as operational resource structure and attributes, while a second ontology model may contain the details of every operation in the workcell, and a third one model may obtain the products features. This privilege is very important in association with the HCA that divides the manufacturing process into resource, production, and order holons.
- **Providing a dynamic knowledge base for the reasoning**: the ontology approach does not only give the privilege of structuring the information into knowledge, but also it enables the access of this knowledge by an RE to analyse and update that knowledge. Which makes the ontology concept very different and superior than the database. As in a database, it is needed to retrieve the data and then form the information and knowledge. In another words, an RE cannot directly use a database without converting the knowledge into data and vice-versa.

4.2.2 Ontology Spectrum and Implementation

Similar to the OOP, an ontology is a conceptual theory that can be implemented via different languages and techniques. There is no standard approach to implement an ontology model, therefore the implementation method and language must be based on the application's purpose and complexity. For this reason, the ontology spectrum that is based on the research in [Obr13, and CLP03] is shown in Figure 4.12. The first type of ontology is based on taxonomy. Taxonomy is a derivative of the Greek word taxis that means arrangement or division, and nomos that means law. Thus, the overall meaning of a taxonomy must be the science of classification according to a predetermined categorization, which provides a conceptual framework for discussion, analysis, or information retrieval. In theory, a good taxonomy classifies all the possible elements of a group (taxon) into subgroups (taxa) that are mutually exclusive and unambiguous.

In practice, a good taxonomy must be simple, easy to remember, and easy to use. Markup languages are commonly used to implement a taxonomy. Examples of these languages are: **XML**, **S**tandard **G**eneralized **M**arkup **L**anguage (**SGML**), **H**yper**T**ext **M**arkup **L**anguage (**HTML**), and e**X**tensible **H**yper**T**ext **M**arkup **L**anguage (**XHTML**). Furthermore, these languages are frequently used as meta-languages to build more complex languages and applications. The syntactic interoperability meaning can be derived from the definition of syntax. A syntax of a computer language is the set of rules, symbols, and principles (i.e., language grammar) that are necessary to govern the language structure. Therefore, syntactic interoperability is obtained if both ends are communicating via the same syntax to interpret the message grammar. Furthermore, the syntactic interoperability is a prerequisite to obtain the structural and semantic interoperability.

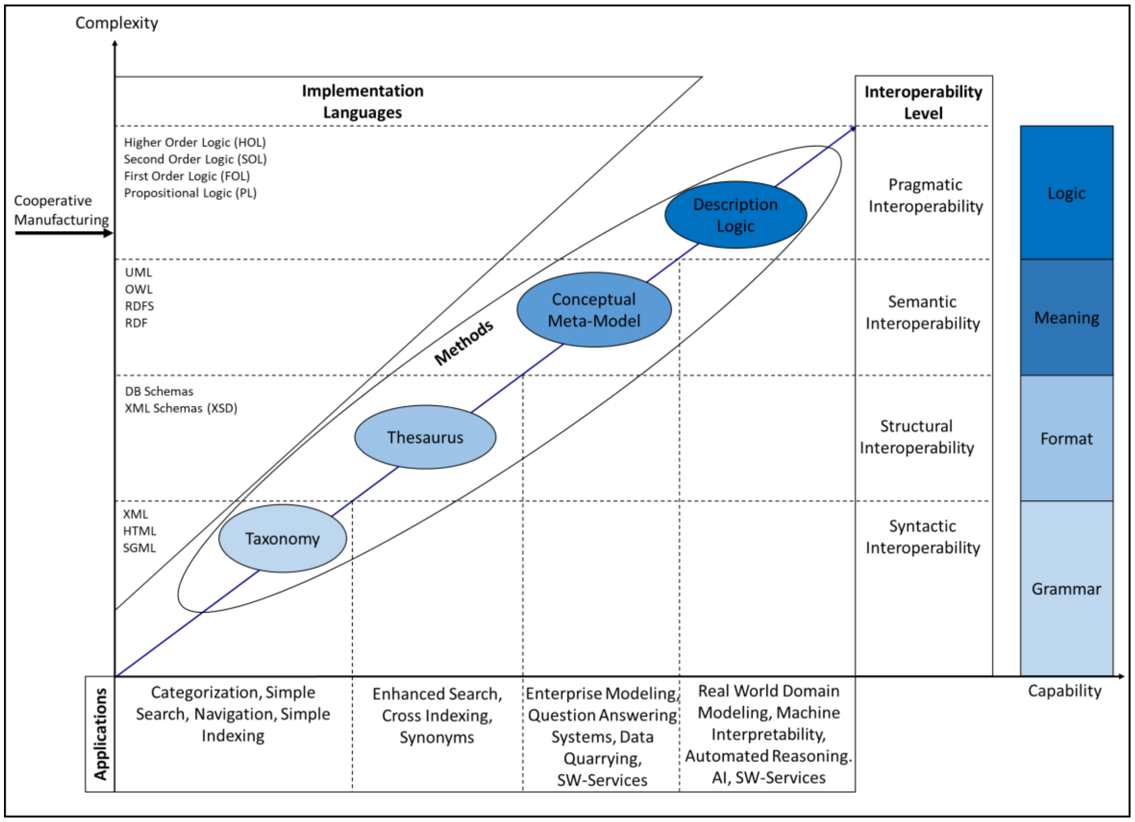

Complexity
Implementation Languages
Interoperability Level
Higher Order Logic (HOL)
Second Order Logic (SOL)
First Order Logic (FOL)
Propositional Logic (PL)
Cooperative Manufacturing
UML
OWL
RDFS
RDF
DB Schemas
XML Schemas (XSD)
XML
HTML
SGML
Description Logic
Conceptual Meta-Model
Methods
Thesaurus
Taxonomy
Pragmatic Interoperability
Semantic Interoperability
Structural Interoperability
Syntactic Interoperability
Logic
Meaning
Format
Grammar
Capability
Applications
Categorization, Simple Search, Navigation, Simple Indexing
Enhanced Search, Cross Indexing, Synonyms
Enterprise Modeling, Question Answering Systems, Data Quarrying, SW-Services
Real World Domain Modeling, Machine Interpretability, Automated Reasoning. AI, SW-Services

Figure 4.12: Ontology spectrum and implementation methods and languages.

A thesaurus concept is similar to a dictionary that stores a networked collection of controlled vocabulary terms, which helps to retrieve these vocabulary terms when they are needed. A thesaurus focuses on the terms and their immediate relationships with each other more than their hierarchy. A thesaurus extends the simple parent-child relation that exist in a taxonomy to one of the following relations: hierarchical (i.e., broader-narrower term), associative with (i.e., see also), and equivalent to (i.e., use/used from or see/seen from). The schema method is usually used to implement a thesaurus. A schema is a representation of a structured data-types model which includes the relations among these data-types. Examples of thesaurus schemas are database schema or **X**ML **S**chema **D**efinition (**XSD**). A schema is often written by one of the previously mentioned markup languages to guarantee the structural interoperability. Structural interoperability is an intermediate level between the syntactic and semantic interoperability, which defines the standard structure and format of exchanged data (i.e., the message standard format).

A conceptual ontology model is an explicit model that constructs the rules, which are needed to express the human understanding of the knowledge of a domain of interest [Inf17]. The importance of using conceptual ontology model has been considered after Tim Berners-Lee, the creator of the **W**orld **W**ide **W**eb (**WWW**), has coined his idea of the semantic web, where webpages are annotated via an ontology-based meta-data [Mot02].

The idea of Tim Berners-Lee is that the webpage must not only link a group of documents to each other, but it must also recognise and utilise the contents of these documents [Küc04]. In other words, the webpage contents must be comprehended by both the human and the machine, which promotes an ontology-based webpage as a solution to extract and retrieve the information. Thus, the meaning of the contents can be processed by external software agents that carry out sophisticated tasks on behalf of the reader and promotes a greater degree of cooperation among the humans and the computing machines. Therefore, an ontology meta-model guarantees the semantic interoperability. The language semantics is the meaning of this language vocabulary due to a predefined syntactic grammar.

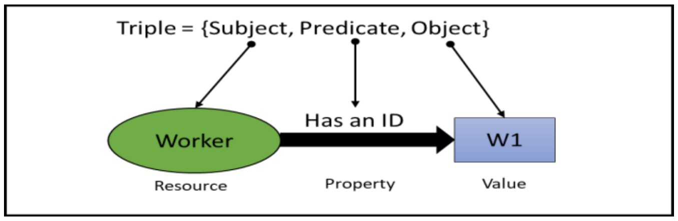


Figure 4.13: RDF triple statement.

Many ontology languages have been developed to solve a particular aspect of the conceptual models. The most famous and simple semantic web language is the RDF. RDF is a data model of resources with their relationships declared by an XML syntax. RDF offers very basic semantics via a triple statement that is called a triple, as shown in Figure 4.13. An RDF triple contains three elements, which are: a subject with a unique name that describes the represented element, a predicate is a relation between the subject and the object, and an object that represents a value of the subject due to a predicate relation. The combination of the RDF document and its schema is commonly known as **R**esource **D**escription **F**ramework **S**chema (**RDFS**). RDFS extends the vocabulary and the relations of RDF to allow taxonomies among the classes and their properties. The OWL also extends RDFS to overcome some of its cons and describes more complex models. OWL introduces a more expressive manner to represent the knowledge by offering a rich set of modelling constructors, such as disjoints, cardinality in types and symmetry in attributes. OWL also offers predefined model templates. Another knowledge representation approach is the UML. OWL is dedicated to develop an ontology-based model, but UML is a general-purpose software language for developing, modelling, and visualizing the software solutions. UML is commonly used in developing database and software modules interaction models. UML models the software systems as classes with attributes and behaviours in an object-oriented style. Thus, UML can construct higher abstraction levels with comparison to OWL [Păt15].

RDF and OWL are used to construct ontology-based conceptual models, but they lack the complex reasoning capabilities. The reasoning capability is stretching the semantic definition beyond the meaning of the vocabulary. This stretch is the pragmatic interoperability, pragmatics is the study of contextual meaning. The meaning of a sentence such as “give me a hand” can be understood literally by a machine to give a hand, while the real meaning is to get a help. The logical theory is the proper mean to enhance the abilities of the web semantics to obtain a better understanding of the contextual meaning and to achieve the pragmatic interoperability [Wu13]. **D**escription **L**ogic (**DL**) is a family of formal logics that are often used in knowledge representation. DL introduces the concept of reasoning in ontologies, which is a very crucial tool to derive extra facts that are not expressed clearly via a conceptual model language. DL is important to ensure the quality of a domain model, as it can be used to verify and test the ontology model during the design phase [BHS05]. Moreover, it provides a mathematical mean to represent complex and ambiguous entities. Different types of logic can be used on top of the conceptual model to construct a DL. The DL language must be proportional with the application complexity. The logic theory offers different DL languages starting from a simple **P**roportional **L**ogic (**PL**) and ending with **H**igher **O**rder **L**ogic (**HOL**).

DL is an appropriate method to model and verify a manufacturing scenario, due to the high level of ambiguity and possibilities in manufacturing. Specifically in cooperative manufacturing, the same concern of semantic web does exist. This concern is a common understanding between the machine which is the cobot and the human who is the worker. Considering the fact that the worker and the cobot understand and express via different types of languages. Moreover, after finding a proper and natural common ground of understanding between the cobot and the worker, they must interact via a meaningful information and knowledge which can be inferred during the cooperation. Therefore, DL ontology is a unique approach to represent and implement a cooperative manufacturing scenario. As an extension of the advantages of the autonomous agent technology that has been discussed earlier, an agent interaction based on DL ontology model has been implemented during this dissertation. The implementation is using FIPA-SL that uses a **F**irst **O**rder **L**ogic (**FOL**) to construct an autonomous agent ontology. JADE ontology-based communication and interaction model will be explained in details in the next section.

4.2.3 Agent Ontology-Based Interaction

Section 4.1.2 introduced the autonomous agent technology as the core of the holon model, where JADE and FIPA communication models along with the ACL-Message structure have been explained in details. An agent uses an ACL-Message as the main tool to interact with the other agents and to achieve the cooperation goals. The power of an autonomous agent that distinguishes it from a software object locates in its ability to interact rather than applying static algorithms. An agent interaction is a form internal conversation where a group of agents negotiate to reach a mutual acceptable agreement on a certain matter

[Kum12]. An agent interaction can be based on cooperative or competitive negotiation. Competitive negotiation takes place when the agents have conflicted local goals, which every agent races to achieve with a minimum cost function. Cooperative negotiation takes place when the agents have common global goals, which they try to achieve together with a minimum cost function. Competitive and cooperative negotiation can exist along together in the same MAS [BHW07]. For example, in the previously proposed HCA, the holons are cooperating to obtain a list of global goals, while the instances of the same holon category may compete to reduce a cost function such as the time. The agent behaviours define its negotiation patterns.

From the philosophical point of view, an agent is similar to a human being who has a group of behaviours and uses one or a group of them concurrently to respond to the environment situations. From the software point of view, a behaviour is an event handler routine that the agent uses to modify its parameters and negotiate with other agents. JADE offers different behaviours that are used to build an agent, four of those behaviours have been used during the implementation phase, these behaviours are the following:

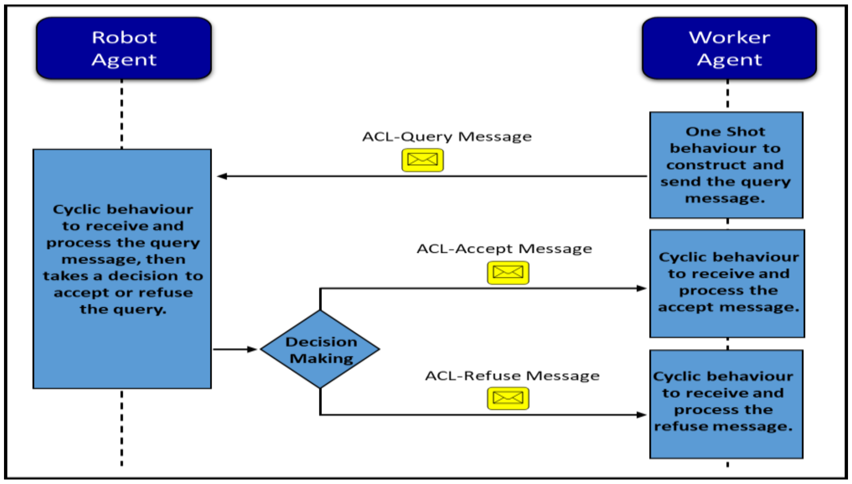

Robot Agent
Worker Agent
ACL-Query Message
One Shot behaviour to construct and send the query message.
Cyclic behaviour to receive and process the query message, then takes a decision to accept or refuse the query.
ACL-Accept Message
Cyclic behaviour to receive and process the accept message.
Decision Making
ACL-Refuse Message
Cyclic behaviour to receive and process the refuse message.

Figure 4.14: Autonomous agent interaction - a pipeline example.

1- **One-shot behaviour:** a simple behaviour that is executed once when it is called by the agent, then it ends. Thus, it is useful to trigger an event and to send an ACL-Message.

2- **Cyclic behaviour:** a simple behaviour that stays active as long as the agent is alive, and it is so useful to receive a message with specific conversation-ID or communicative-act.

3- **Sequential behaviour:** a composite behaviour that controls the sequence of execution of more than one-shot behaviour. The sequence control is not always based on a fixed order. As it can be based on the sum of the input events from the agent environment.

4- **Parallel behaviour:** is a composite behaviour that concurrently controls the execution and the termination of more than parallel one-shot behaviour.

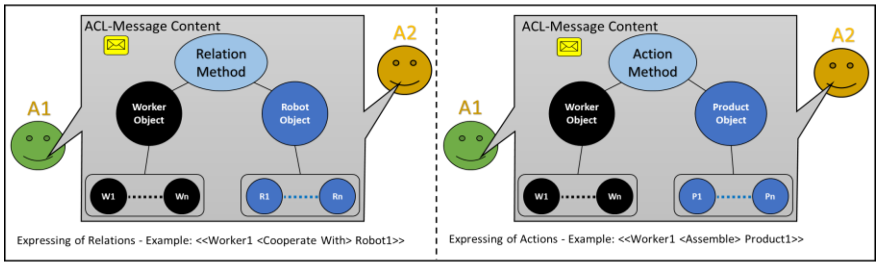
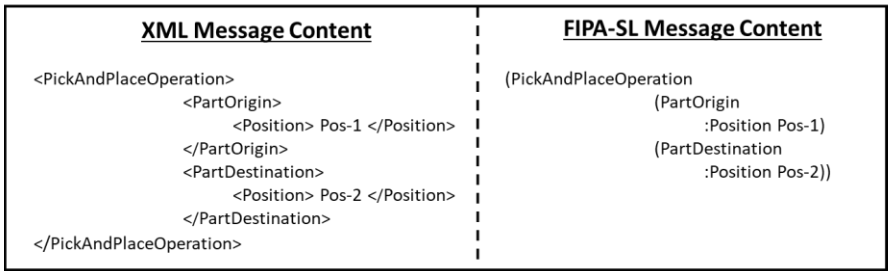


Figure 4.15: Agents communication via ontology.

As it could be seen from the previous behaviour, an agent architecture goes many steps farther than the software object by providing new concepts and tools such as the behaviours and the ACL-Message. One of the main roles of a behaviour is to control the flow of the ACL-Messages. The summation of an agent behaviour and an ACL-Message equals to an agent interaction. The researches in [WAG+05, and GW06] state that, "*agent interaction is more powerful concept than ordinary algorithm*". The main reason behind the previous statement is that an interaction is providing the capability of learning from the I/O history tracking, and then it changes the environment model based on this obtained knowledge. While, an algorithm is static and meant to produce an output due to a specific input. Furthermore, using the concept of interaction via the agent behaviours and ACL-Messages, gives the agent the sense of intelligence by taking its own decisions based on the interaction context, as shown in Figure 4.14.



XML Message Content

```
<PickAndPlaceOperation>
        <PartOrigin>
                <Position> Pos-1 </Position>
        </PartOrigin>
        <PartDestination>
                <Position> Pos-2 </Position>
        </PartDestination>
</PickAndPlaceOperation>
```

FIPA-SL Message Content

```
(PickAndPlaceOperation
        (PartOrigin
                :Position Pos-1)
        (PartDestination
                :Position Pos-2))
```

Figure 4.16: XML syntax versus FIPA-SL syntax.

The representation of the content field of an ACL-Message is a String data type in case of simple agent conversation. However, in a complex agent conversation, it is not only a must to represent the software objects, but also the relations among these objects and the actions they perform, as shown in the example in Figure 4.15. Therefore, an ontology-based content is the proper method of conversation. XML or FIPA-SL languages are commonly used to represent an ACL-Message content. XML and FIPA-SL are slightly different in their syntax, as shown in Figure 4.16. However, representing an ACL-Message in FIPA-SL is preferable in case of further reasoning of the message based on the logic theory is needed. The representation of an ACL-Message can be also as a sequence of Bytes that is a light-weighted method from the programming point of view. Although, it is not preferable, as it is non-human readable language, thus it cannot be debugged.

One of the advantages of using an ontology model is the mutual understanding among the software agent. In order to bring this mutual understanding among the agents, they must communicate via the same syntax and semantics. JADE agents are written as a JAVA code that obligates them to process JAVA objects. Therefore, during sending an ontology-based message, the sender must convert its internal representation of the message from JAVA syntax to the appropriate syntax of the corresponding content language (i.e. XML, FIPA-SL, Bytes). However, before the syntax conversion, semantic checking must be fulfilled. Thus, an agent must check and validate an ontology-based ACL-Message against the domain semantics before converting it to the proper syntax. Ontology schemas are used by an agent to express its domain semantics. JADE agents define three types of schemas which are summarized as follows:

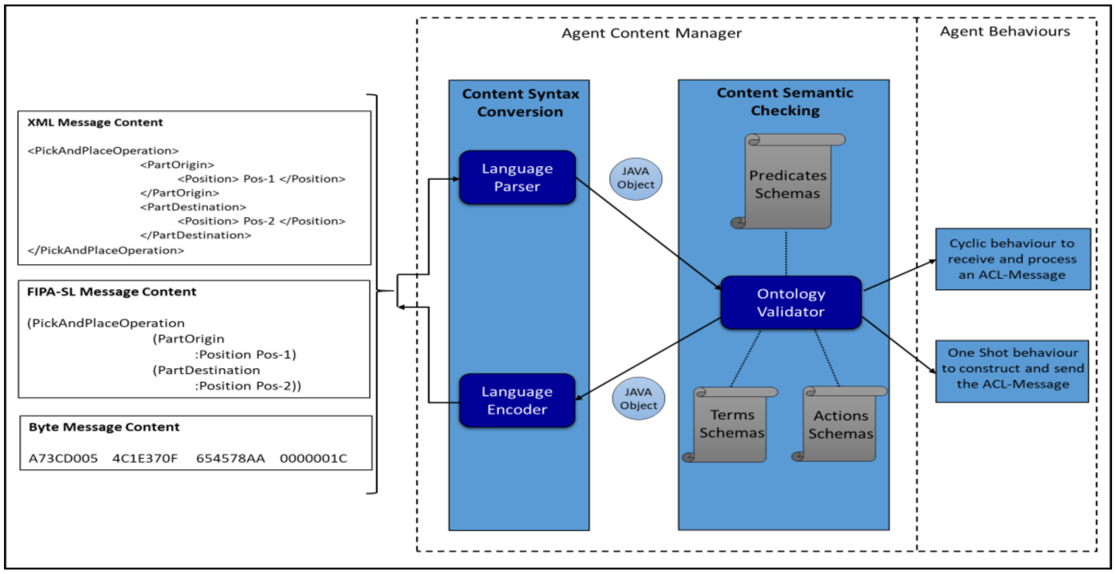


Figure 4.17: Sending/Receiving mechanism of an ontology-based ACL-Message.

- **Term schema**: contains expressions that indicate entities that exist in the agent domain, hence the agents may reason about. Terms are either primitives that are atomic data types such as Strings or Integers, or concepts that are complex structure such as objects.
- **Predicate schema**: contains expressions that describe the status of the agent's domain and the relationships between the concepts.
- **Action schema**: contains expressions that describe the routines and the operations that can be executed by an agent.

During an ontology-based ACL-Message reception, the opposite mechanism of the sending operation must be fulfilled. Therefore, the same schemas of the sender must be accessible by the receiver in order to interpret the message content. An illustration of the sending/receiving operations of an ontology-based ACL-Message is shown in details in Figure 4.17.

Chapter 5 - Solution Concept: Holonic Control Architecture

"Simplicity is the Ultimate Sophistication."

Leonardo da Vinci

5.1 *Definition of Terminology*

The definition of terminologies that are involved in cooperative manufacturing is an essential part of this dissertation. Due to the novelty of the subject, new terminologies will be introduced to provide a mutual understanding of the solution concept. All the definitions are adapted from ISA-88 [ISA88] and ISA-95 [ISA95] standards. Both ISA-88 and ISA-95 are created by the **I**nstrumentation, **S**ystem, and **A**utomation (**ISA**) society. ISA-88 addresses batch process control and supports both the automatic and the manual manufacturing. While, ISA-95 targets the industrial enterprise automation, by developing an automated interface between the enterprise and the control systems. ISA-95 is a generic manufacturing standard that is implemented in many industries. Figure 5.1 assumes an industrial enterprise that applies the cellular manufacturing concept over its shop floor. The manufacturing scenario starts from the customer, whom sends a customized order to the enterprise MOM. The MOM defines the production order, which contains all the details to manufacture this order. Therefore, the production order is broken into processes. Then, the processes are distributed over the manufacturing workcells based on matching every workcell capability with the required process. If the production style is sequential, the product flows from one workcell to another until reaching a final product. Accordingly, a coordination among the manufacturing workcells is required.

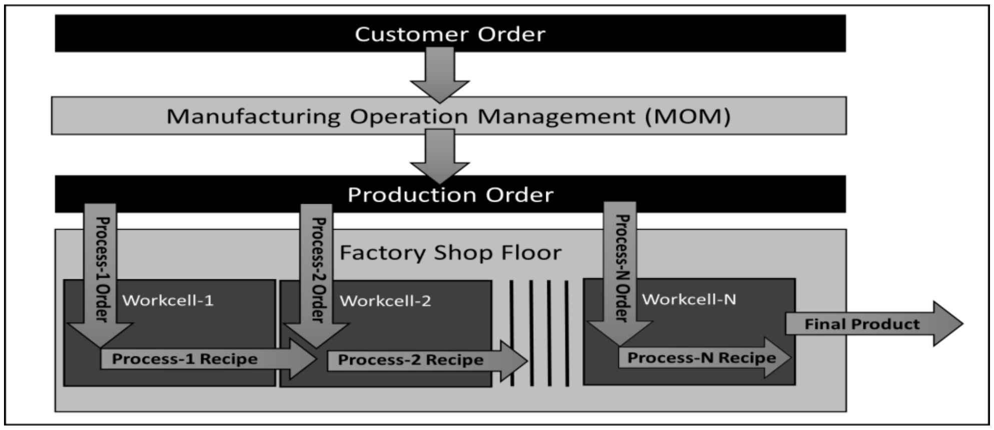


Figure 5.1: Cellular manufacturing concept.

Figure 5.2 is an example of a product order. In this example, every process has a specific recipe. A process recipe contains all the knowledge, which is needed to complete the process. Every process contains a set of operations in a specific order. Every operation can be seen as a schema, which contains the related information to this specific operation (i.e., operation plan). The formulization of the terms that have been used to construct the manufacturing scenario in Figure 5.1 and Figure 5.2, are stated as follows:

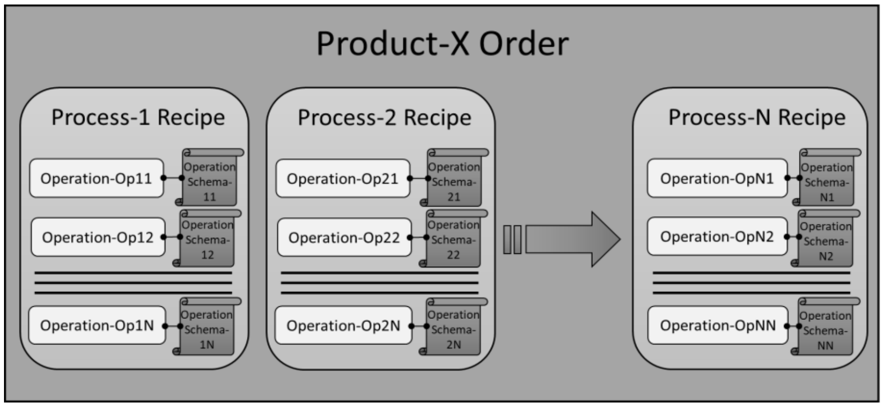


Figure 5.2: Product order structure.

- **Customer order**: the minimum set of information, which is required from the customer and is needed to build the product order.
- **Product order**: the sequence of processes, which are needed to manufacture a product. A product can extend from a single process to a set of processes.
- **Process order**: the sequence of operations, which are needed to form a process. A process may extend from a single operation to a set of operations.
- **Process recipe**: based on ISA-88 standard, process recipe is the necessary knowledge that uniquely defines the production requirements of a specific product or process. This knowledge differs from one process to another based on the nature of the process.
- **Operation plan/schema:** an operation plan or schema is a structured representation of all the required information, which is needed to perform an operation.
- **Operation**: an operation can be a specific manufacturing operation such as (assembly, drilling, machining, welding, gluing, painting, etc.), or a general production operation such as (material or parts handling, packaging, quality checking, etc.).
- **Workcell capability**: a process or the set of processes, which are provided by a workcell.
- **Resource capability**: a production operation or a set of operations, which are associated with a specific resource. In other words, what can every operational resource do.
- **Identification (ID)**: an identifier that uniquely addresses a workcell, a product, a process, an operation, a resource, or a task.

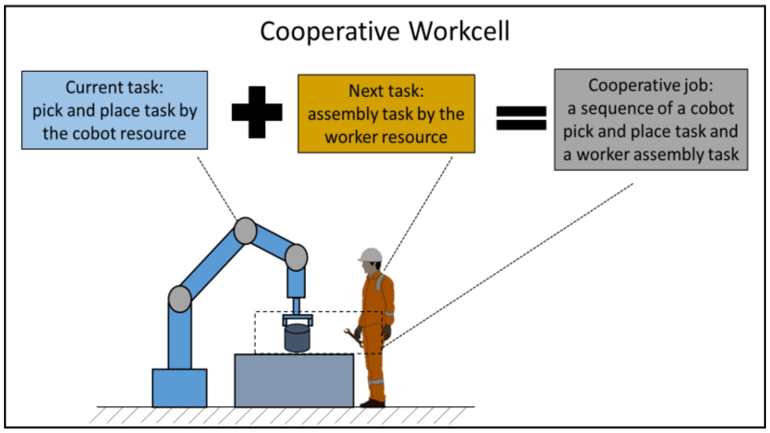

Cooperative Workcell
Current task: pick and place task by the cobot resource
Next task: assembly task by the worker resource
Cooperative job: a sequence of a cobot pick and place task and a worker assembly task

Figure 5.3: Cooperative job - an example**.**

The previously described manufacturing scenario is very generic to provide the whole picture of the enterprise. However, as the enterprise size is beyond the implementation limits of this dissertation, some assumptions will be made during the implementation phase in chapter 6. One of those assumptions is that the product order is composed of only one process. In other words, the product order is equal to one process order. For example, an assembly process is considered as the whole product order. An assembly process contains two operations, which are pick and place and assembly. Furthermore, new terminologies such as a cooperative job is introduced by this dissertation. An illustration of the meaning of a cooperative job is shown in Figure 5.3. The related terms that give the full understanding of the cooperative job meaning are explained as follows:

- **Production task**: a specific operation plan with a specific ID and status.
- **Assigned task**: a production task, which is assigned to a specific operational resource.
- **Task processing time**: the time needed by the resource to finish an assigned task.
- **Cooperative job**: within the context of cooperative manufacturing, a cooperative job is a combination of two consequent tasks that complement each other. The first task is assigned to the worker and the second is assigned to the robot, or vice-versa.
- **Cooperative resources pairing**: resources pairing is the selection of two resources, who perform together a cooperative job. Then assign a task to the first resource and reserve the second resource who must wait the first resource to finish. The resources paring is required only in some cases, when many resources can perform the same operation.
- **Workstation unit:** an equipped work location that is dedicated for the performance of a specific task via a worker or a machine. A workcell is often composed of a group of workstations.
- **Buffer unit:** a storage unit that stores the output of a previous workstation until the next workstation is ready to process this output.

- **Task history**: production history contains detailed information regarding the previous production tasks. This information includes when did the task start and finish and for whom it has been assigned. Task history is useful in some cases during the task assignment decision-making. In addition, it is essential to calculate important factors such as the workcell productivity and performance.
- **Process history:** a similar concept to the task history but applied all over the process.
- **Production constrains**: alarms and warnings, which caused by some unexpected events during the manufacturing and restrain the production. Examples of production constrains can be specific parts or tools that are not available, a robot fail down, or a conveyor is blocked.
- **Production orders scheduling**: rearranging the product orders in a specific sequence that obtains an optimum production efficiency.
- **Product customization:** the product features that are required by the customer.
- **Production volume:** the number of product units that are required by the customer.

5.2 *Concept Construction: Holon Object-Oriented Model*

Chapter 4 has studied the holon implementation model from the software perspective. This model is based on the definition that a holon is a distributed software control component which stores, exchanges, processes, and validates the information within the manufacturing system. The nature of the information that the holon deals with differs due to the function of the holon. Therefore, during this section of the dissertation, a holon object-oriented model is introduced based on its function within the cooperative manufacturing system. The holon object-oriented model is defined in terms of the holon knowledge and behaviours. The holon knowledge can be seen as the way it stores the information in a structured format. While, the holon behaviours represent the way it exchanges and processes this knowledge. Thus, the holon knowledge and behaviours are studied in details with respect to the holon function in cooperative manufacturing at the rest of this section. The proposed holon object-oriented models are generic to be implemented in any OOP language, as it will be seen in chapter 6.

5.2.1 Operational Resource Holon

An ORH object-oriented model in a cooperative manufacturing scenario that describes either a **W**orker **H**olon (**WH**) or a **R**obot **H**olon (**RH**). In addition, it fits other operational resources such as a machine. Accordingly, the design of the ORH object-oriented model are generic enough to cover the previously mentioned operational resources. The responsibilities of the ORH are the main source of its behaviours and knowledge as shown in Figure 5.4, thus these responsibilities are discussed in details as follows:

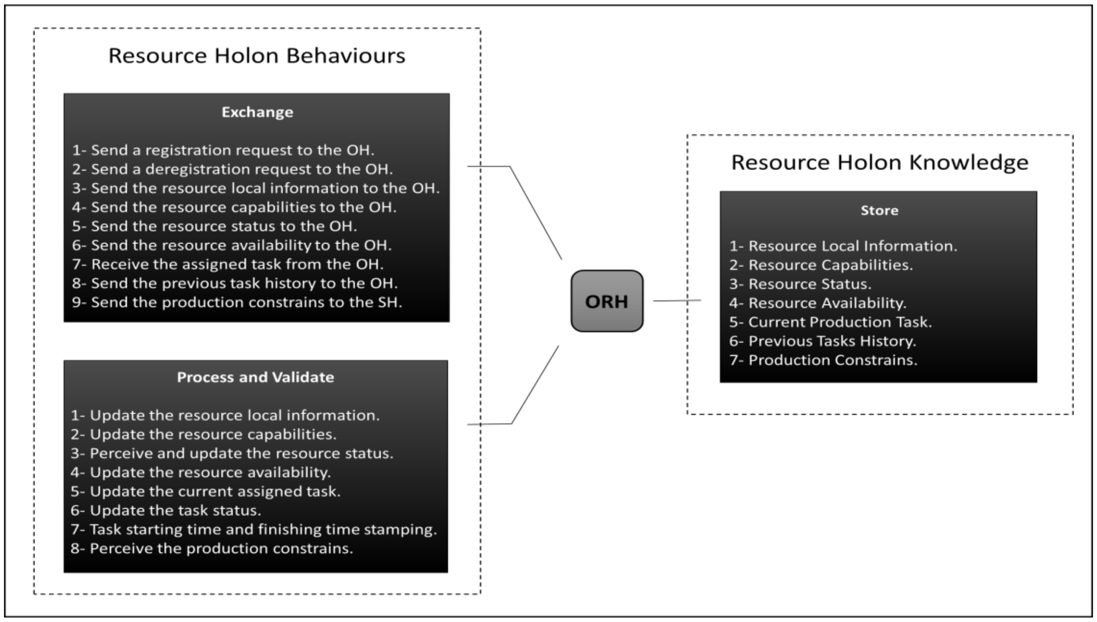
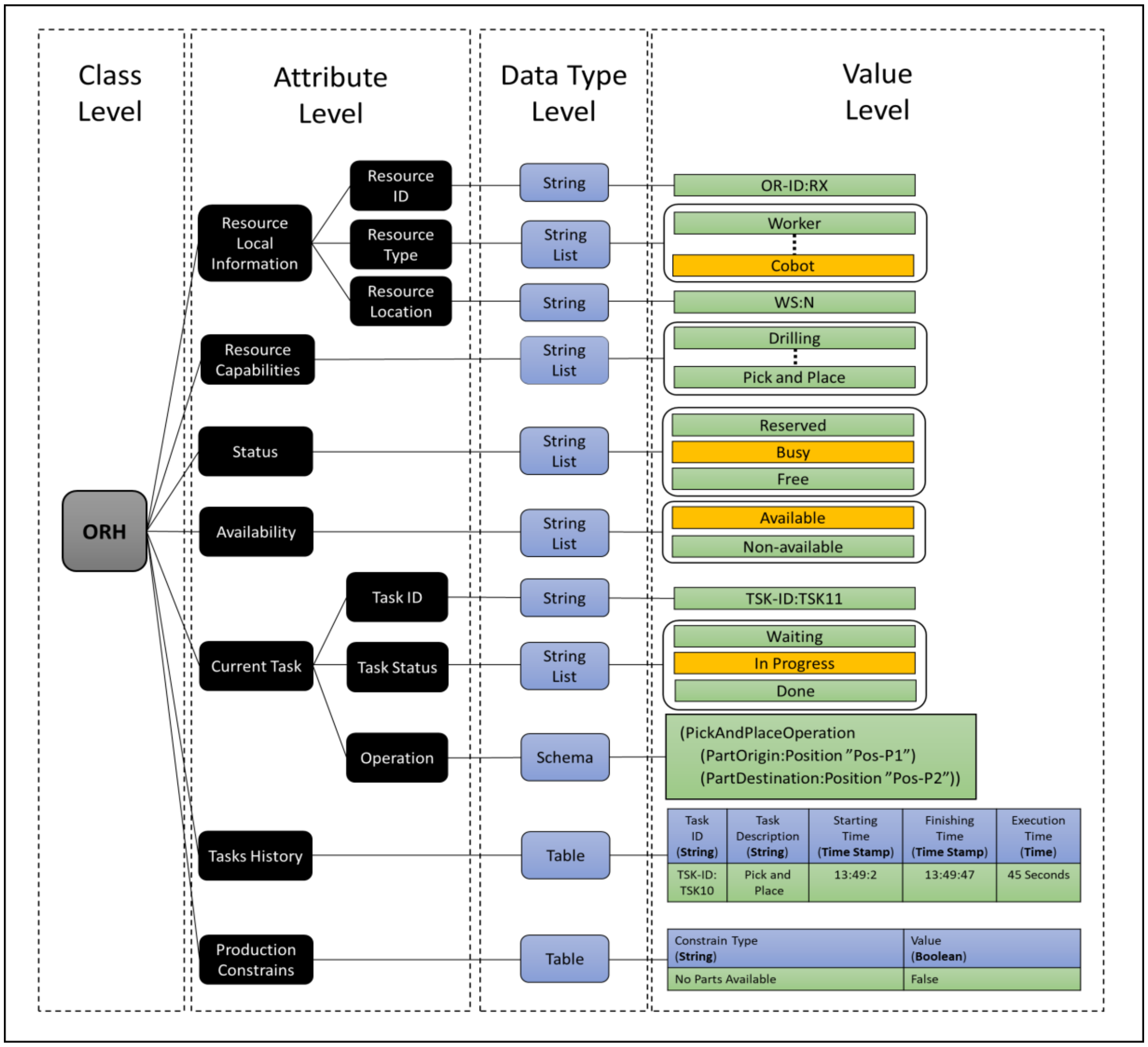

Resource Holon Behaviours
Exchange
1- Send a registration request to the OH.
2- Send a deregistration request to the OH.
3- Send the resource local information to the OH.
4- Send the resource capabilities to the OH.
5- Send the resource status to the OH.
6- Send the resource availability to the OH.
7- Receive the assigned task from the OH.
8- Send the previous task history to the OH.
9- Send the production constrains to the SH.
Process and Validate
1- Update the resource local information.
2- Update the resource capabilities.
3- Perceive and update the resource status.
4- Update the resource availability.
5- Update the current assigned task.
6- Update the task status.
7- Task starting time and finishing time stamping.
8- Perceive the production constrains.
ORH
Resource Holon Knowledge
Store
1- Resource Local Information.
2- Resource Capabilities.
3- Resource Status.
4- Resource Availability.
5- Current Production Task.
6- Previous Tasks History.
7- Production Constrains.

Figure 5.4: ORH object-oriented model in cooperative manufacturing.

- **Resource registration/deregistration**: as mentioned before, the cooperative workcell must cope with the variations in the number of operational resources within the workcell. Therefore, any resource must register its presence at the very beginning of a manufacturing shift and deregister itself after ending the manufacturing shift.
- **Resource local information**: after a new resource registers to the workcell, a set of related local information is associated with this resource. The resource local information contains different attributes that can be seen in details in Figure 5.5. Attributes such as the resource-ID and type help to distinguish this resource while taking a related decision, during the task assignment. Other attributes such as the resource location helps to physically reach the resource, during transferring the product. In the workcell, the resource is often located in a predefined location that called a workstation.
- **Resource capabilities**: a resource may have different capabilities and thereby perform different operations. For example, a worker may perform an assembly and a packaging at the same time. The resource capabilities must be known by the OH, thus it can assign the right operation to this specific resource.
- **Resource status**: the resource holon as a software component must reflect the current physical status of the resource. The resource status can be busy, reserved, or free. If the resource status is free, the OH is allowed to consider this resource to assign a task. Otherwise, the OH looks for another free resource with the same capability, or waits until the resource becomes free. The reserved status is needed to reserve a resource until the cooperative resource finishes.



Task ID (**String**)	Task Description (**String**)	Starting Time (**Time Stamp**)	Finishing Time (**Time Stamp**)	Execution Time (**Time**)
TSK-ID: TSK10	Pick and Place	13:49:2	13:49:47	45 Seconds

Constrain Type (**String**)	Value (**Boolean**)
No Parts Available	False

Figure 5.5: ORH knowledge structure model - an example.

- **Resource availability**: the resource holon must reflect the availability of the physical resource as well. A resource can be registered to the workcell, but it is not available. For instance, a worker is unavailable during taking a break or a rest, and a robot is unavailable during the reprogramming or the maintenance.
- **Current production task**: current production task describes the operation that is assigned to the resource. Every resource can have one task assignment at a time regardless how many capabilities are owned by this resource. The task representation informs the resource with the know-how of performing the task. This representation can be a string based in case of simple manufacturing operations, or a schema based in case of complex manufacturing operations. The production task is identified by a task-ID attribute. Finally, the task status attribute can have different values that helps to track the task progress and control the other tasks execution sequence.

- **Previous tasks history**: the ORH is responsible for the time stamping of every assigned task at the start and at the end of the task. Accordingly, the task's T_P can be calculated and stored as a task history. Then, the resource sends the task history to the OH for further processing.
- **Production constrains:** are problems within the shop floor, which can be directly detected by the sensors on the physical layer, or reported by the operational resources (e.g., workers). The ORHs collect all the constrains and send them to the SH in a large-scale enterprise or to the PH in a small-scale enterprise for further processing.

5.2.2 Order Holon

An OH links the ORHs to the PH. Therefore, it coordinates with the PH to obtain the current production task, and then it negotiates with the ORHs to assign the task to one of them. The required behaviours and knowledge by the OH to achieve its responsibilities are shown in Figure 5.6, and discussed in details as follows:

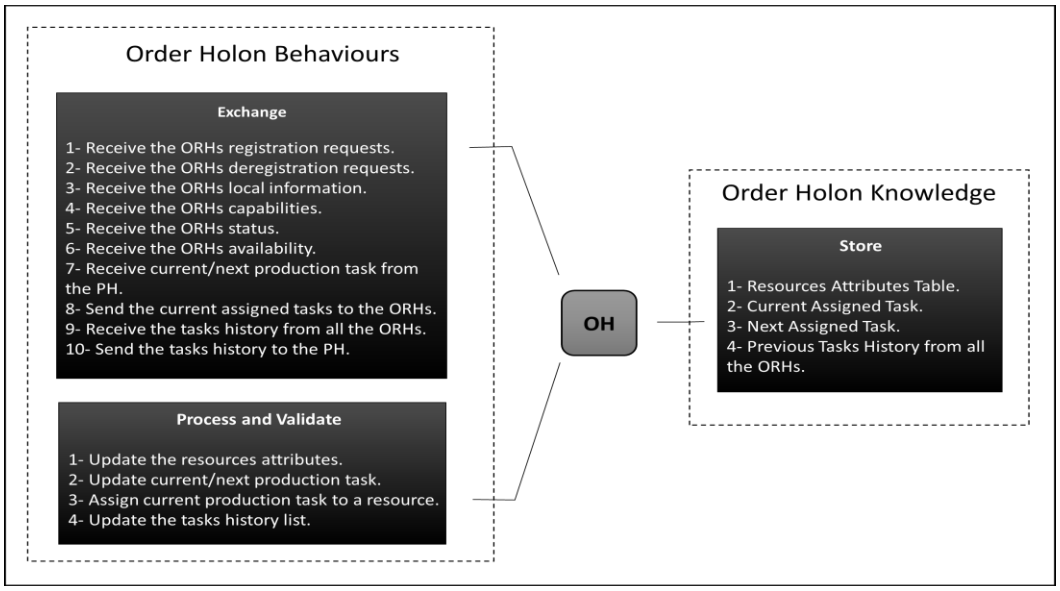
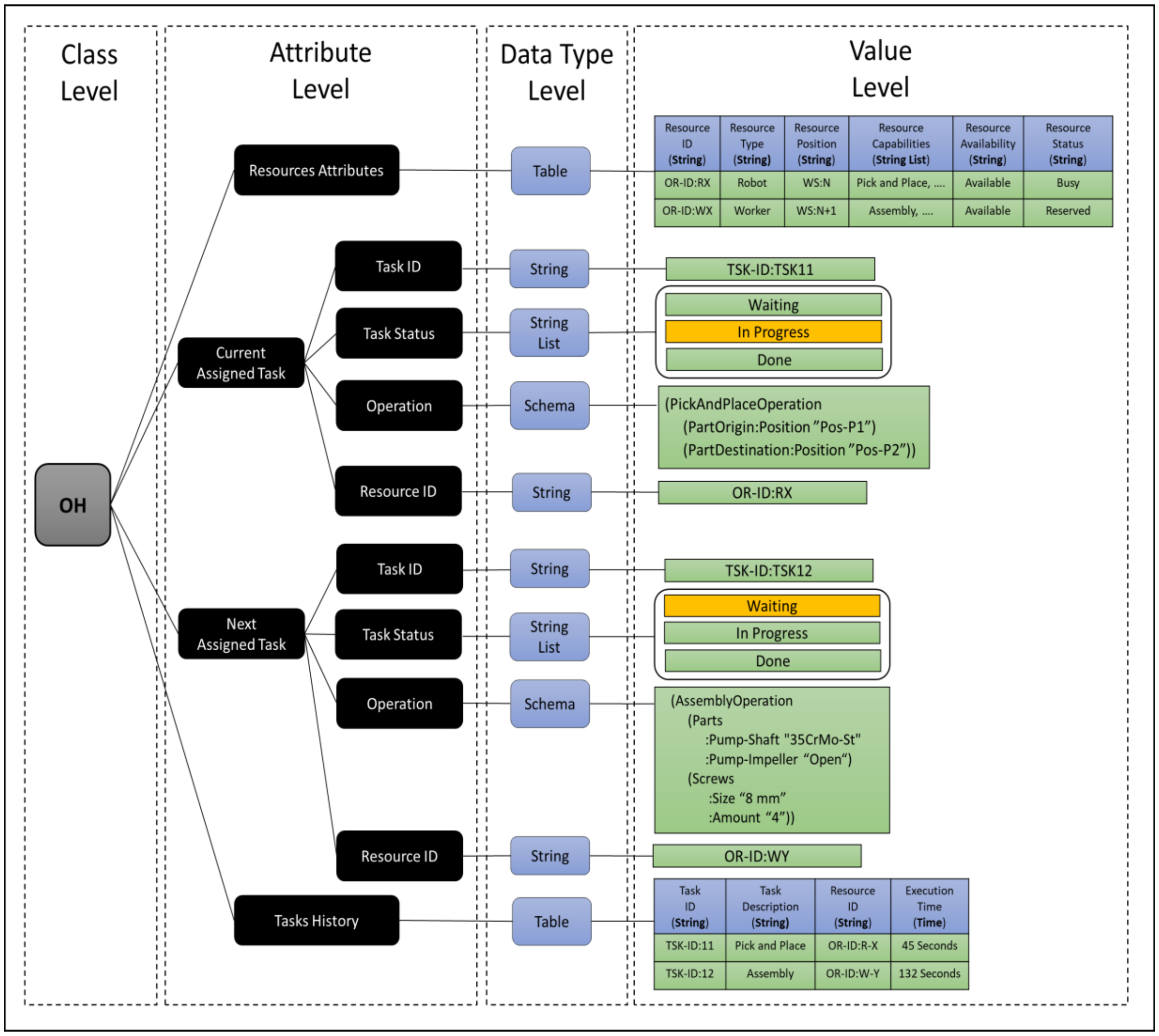

Order Holon Behaviours
Exchange
1- Receive the ORHs registration requests.
2- Receive the ORHs deregistration requests.
3- Receive the ORHs local information.
4- Receive the ORHs capabilities.
5- Receive the ORHs status.
6- Receive the ORHs availability.
7- Receive current/next production task from the PH.
8- Send the current assigned tasks to the ORHs.
9- Receive the tasks history from all the ORHs.
10- Send the tasks history to the PH.
Process and Validate
1- Update the resources attributes.
2- Update current/next production task.
3- Assign current production task to a resource.
4- Update the tasks history list.
OH
Order Holon Knowledge
Store
1- Resources Attributes Table.
2- Current Assigned Task.
3- Next Assigned Task.
4- Previous Tasks History from all the ORHs.

Figure 5.6: OH object-oriented model in cooperative manufacturing.

- **Resource registration/deregistration**: when the ORHs register to or deregister from the cooperative workcell, they interact with the OH to accomplish this process. Before a physical resource registers itself to the workcell, the ORH instance of this physical resource does not exist. However, when the physical resource triggers a registration event from an automation device, the OH receives this event and creates an ORH instance that is dedicated to this physical resource. The opposite scenario happens when a physical operational resource deregisters from the workcell, as the OH deletes this dedicated ORH from the HCA.

- **Resources attributes**: the OH must monitor all the operational resources in order to assign the production tasks to them. Therefore, an OH resource continuously collects static and dynamic values of the resources attributes in a table form, which can be seen via Figure 5.7. The static attributes of an ORH are those ones with a fixed value such as the resource's local information and capabilities. While, the dynamic attributes are those ones that vary such as the resource status and availability.



Resource ID **(String)**	Resource Type **(String)**	Resource Position **(String)**	Resource Capabilities **(String List)**	Resource Availability **(String)**	Resource Status **(String)**
OR-ID:RX	Robot	WS:N	Pick and Place, ...	Available	Busy
OR-ID:WX	Worker	WS:N+1	Assembly, ...	Available	Reserved

Task ID **(String)**	Task Description **(String)**	Resource ID **(String)**	Execution Time **(Time)**
TSK-ID:11	Pick and Place	OR-ID:R-X	45 Seconds
TSK-ID:12	Assembly	OR-ID:W-Y	132 Seconds

Figure 5.7: OH knowledge structure model - an example.

- **Current assigned task**: the main purpose of the OH is to find a suitable ORH to assign a task to it. Therefore, the OH receives the current task from the PH and matches it with the capabilities of the available free resources. When a resource match is found, the OH sends the current task assignment to this resource as explained earlier in the previous section.

- **Next assigned task**: the nature of a cooperative job implies that in some cases, two cooperative resources may operate together at the same time, but they do not finish at the same exact time. For example, if a robot handles a specific quantity of product parts to a worker who performs an assembly operation. The worker can start the assembly operation after the required parts of only one product are handled, while the robot is still handling the rest of the parts to the worker. In such cases, the OH must know the next production task and assigns it to the resource whom finishes first, assuming that there are more than one worker in cooperation with more than one robot.
- **Previous tasks history**: the previous tasks history is very influential factor to consider while assigning a task in some ambiguous situations. For instance, when two or more operational resources are in free status and all of them are capable of performing the same task. Accordingly, the OH collects the tasks history from all the ORHs in a table form that can be seen in Figure 5.7, and then the OH uses the tasks history when it is needed in the decision-making algorithm. Moreover, the tasks history is sent by the OH to the PH for further processing.

5.2.3 Product Holon

A PH stores and processes consistent up-to-date information that is related to the production processes. In a small-scale enterprise, the PH directly generates the product recipe from the customer order. While, in a large-scale enterprise, the PH obtains the process orders from the SH. The required behaviours and knowledge by the PH to achieve its responsibilities are shown in Figure 5.8, and discussed in details as follows:

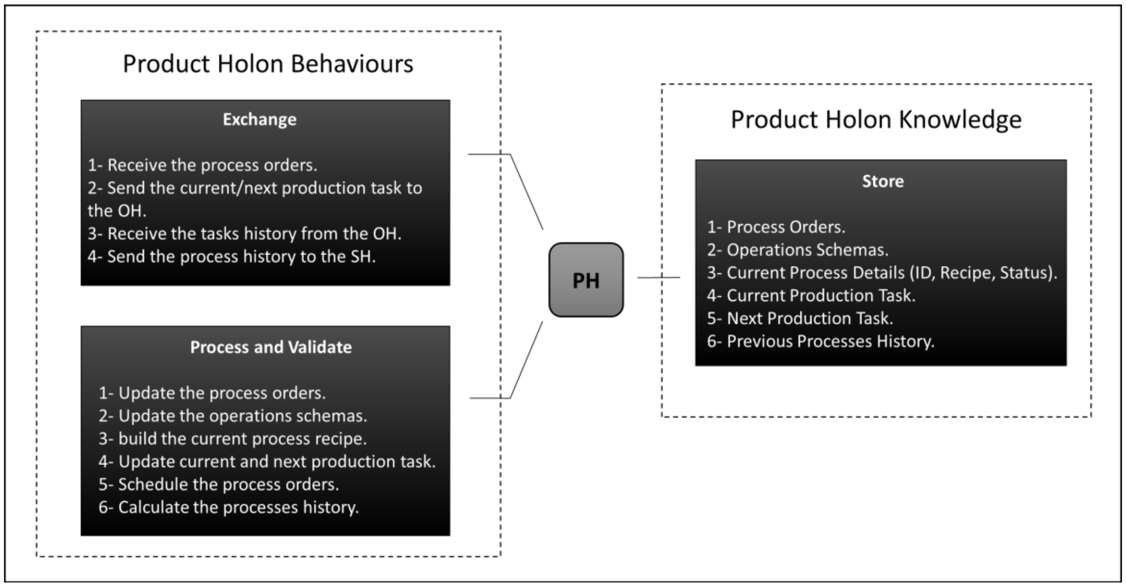
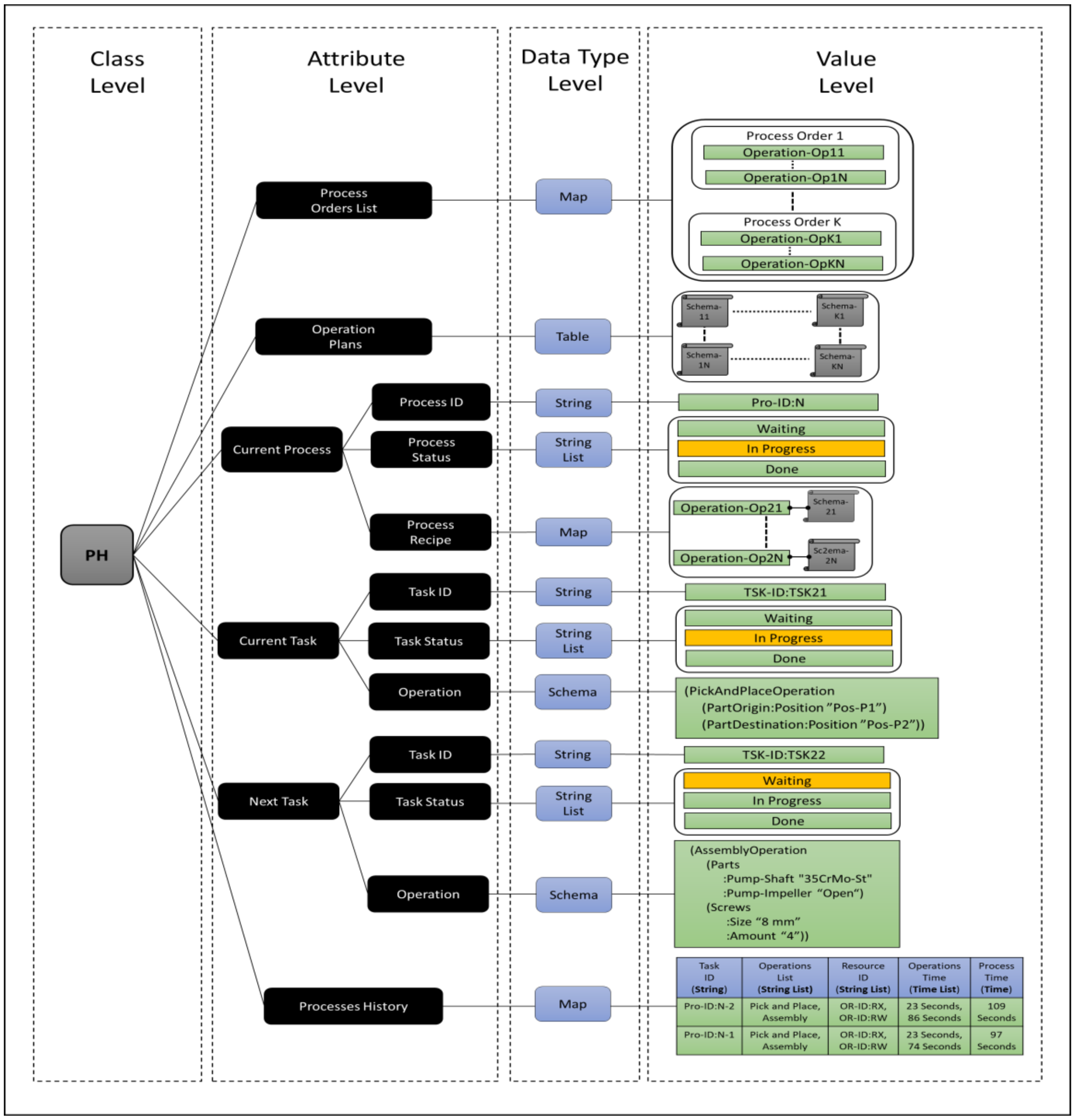

Product Holon Behaviours
Exchange
1- Receive the process orders.
2- Send the current/next production task to the OH.
3- Receive the tasks history from the OH.
4- Send the process history to the SH.
Process and Validate
1- Update the process orders.
2- Update the operations schemas.
3- build the current process recipe.
4- Update current and next production task.
5- Schedule the process orders.
6- Calculate the processes history.
PH
Product Holon Knowledge
Store
1- Process Orders.
2- Operations Schemas.
3- Current Process Details (ID, Recipe, Status).
4- Current Production Task.
5- Next Production Task.
6- Previous Processes History.

Figure 5.8: PH object-oriented model in cooperative manufacturing.

- **Process orders list**: the PH receives the process orders from the SH, then stacks them in a list and ensures the correct execution of this list. An example of a process orders list can be seen in Figure 5.9. It is possible that a workcell provides more than one process, if the operation resources can perform more than one operation per each, or if the sequence of the operations within the process is alternated.
- **Operation plans**: operation plans can be also denoted as the operation schemas. An operation schema represents the know-how of an operation. A schema is one method to encapsulate the manufacturing knowledge. However, other methods such as software objects, collections, or arrays can be also used to achieve the same purpose. The selection of the schema method is up to the designer and the complexity of the operations.



```
(PickAndPlaceOperation
   (PartOrigin:Position "Pos-P1")
   (PartDestination:Position "Pos-P2"))
```

```
(AssemblyOperation
   (Parts
      :Pump-Shaft "35CrMo-St"
      :Pump-Impeller “Open“)
   (Screws
      :Size “8 mm”
      :Amount “4”))
```

Task ID **(String)**	Operations List **(String List)**	Resource ID **(String List)**	Operations Time **(Time List)**	Process Time **(Time)**
Pro-ID:N-2	Pick and Place, Assembly	OR-ID:RX, OR-ID:RW	23 Seconds, 86 Seconds	109 Seconds
Pro-ID:N-1	Pick and Place, Assembly	OR-ID:RX, OR-ID:RW	23 Seconds, 74 Seconds	97 Seconds

Figure 5.9: PH knowledge structure model - an example.

- **Current process**: a process order is a sequence of operations that are needed to form a process. The PH associates these operations with their plans and links them together to form a process recipe. Then, it addresses this recipe by an ID. Moreover, the PH links the process recipe to a status attribute and value that help to track the progress of the process. An example of a production process can be seen via Figure 5.9. The selection of the current process is based on the **F**irst **C**ome **F**irst **S**erve (**FCFS**) bases if the processes are depending on each other (i.e., sequential production method). However, in other production methods where there are no dependencies among the processes, FCFS is not the most efficient method to select the current process. Scheduling the production processes is the responsibility of the PH, which will be covered in details in section 5.3.3.
- **Current production task**: after selecting the current production process, the PH breaks it into operations to create the current production task. Similar to the current production process, the PH links the current production operation to an ID and status attributes to address and track it.
- **Next production task:** the same mechanism to create the current production task is followed to create the next production task. Both the current and the next production tasks are sent by the PH to the OH, whom is responsible for assigning them to the ORHs.
- **Previous processes history:** as briefly discussed before, the PH is responsible for scheduling the production processes. In order to do that, the PH calculates the previous processes T_P. Thus, it can apply the proper scheduling algorithm. Moreover, the processes history is sent by the PH to the SH for further processing.

5.2.4 Supervisor Holon

An SH is required in case of a large-scale enterprise where there are more than one cooperative workcell. In the vertical direction of an enterprise, the SH is linked to the MOM, and the ERP. In the horizontal direction of an enterprise, the SH is linked to the other SHs from the other workcells. The SH will be excluded from the implementation phase as it locates outside the boundaries of the cooperative workcell. However, it will be included in the design phase to extend the current implementation in the future work. The required behaviours and knowledge by the SH to achieve its responsibilities are shown in Figure 5.10, and discussed in details as follows:

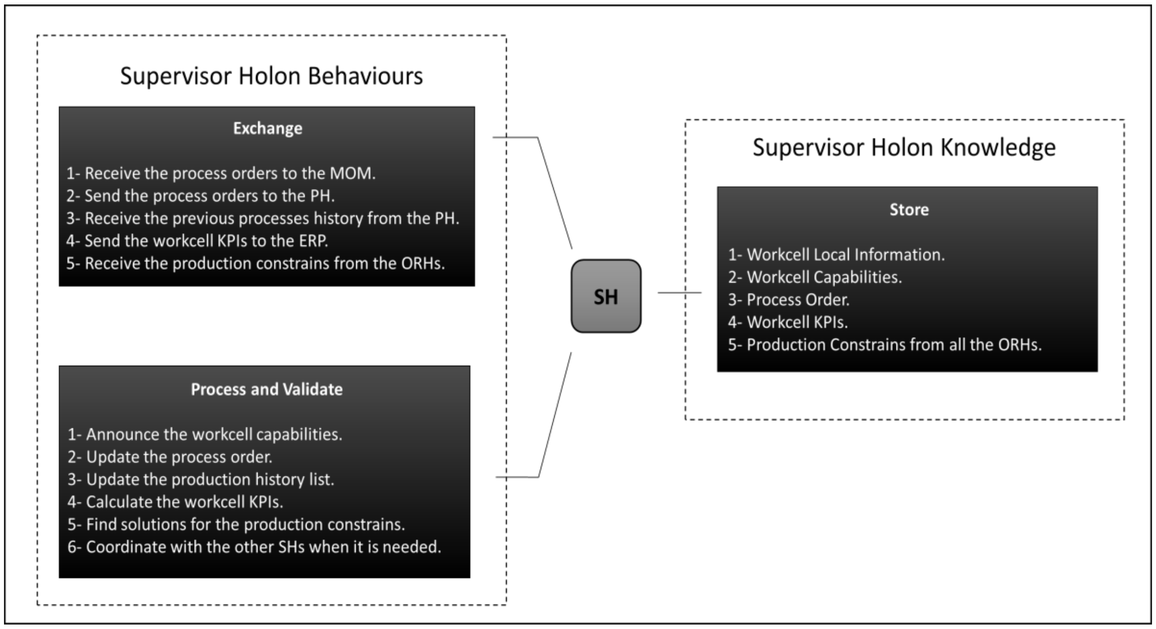
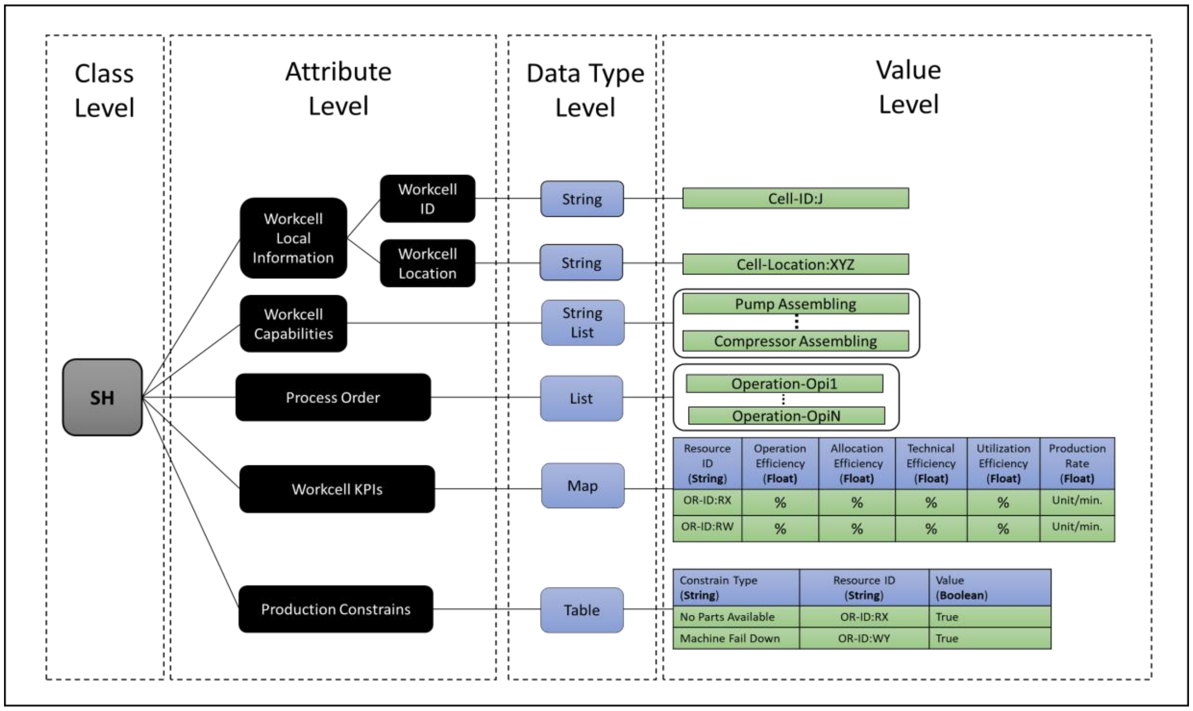

Supervisor Holon Behaviours
Exchange
1- Receive the process orders to the MOM.
2- Send the process orders to the PH.
3- Receive the previous processes history from the PH.
4- Send the workcell KPIs to the ERP.
5- Receive the production constrains from the ORHs.
Process and Validate
1- Announce the workcell capabilities.
2- Update the process order.
3- Update the production history list.
4- Calculate the workcell KPIs.
5- Find solutions for the production constrains.
6- Coordinate with the other SHs when it is needed.
SH
Supervisor Holon Knowledge
Store
1- Workcell Local Information.
2- Workcell Capabilities.
3- Process Order.
4- Workcell KPIs.
5- Production Constrains from all the ORHs.

Figure 5.10: SH object-oriented model in cooperative manufacturing.

- **Workcell local information**: the SH holds the workcell local information such as the workcell-ID and location which help to address and reach this specific workcell. An example of the workcell local information attribute can be seen in Figure 5.11.
- **Workcell capabilities**: a workcell can perform more than one specific process. For example, one workcell may assemble different products, thus every product assembly is considered as a different process. Therefore, the SH must know these processes (i.e., the workcell capabilities) while negotiating with the MOM to extract the appropriate processes from the production order.
- **Process order:** the SH works as a broker who announces the workcell capabilities to the MOM. Thus, the MOM sends the appropriate process order to be performed by this workcell. Accordingly, the SH forwards the process order to the PH which stacks it in a process order list until its turn comes.
- **Production KPIs:** the SH receives the processes history from the PH. Then, it uses this history to calculate the cooperative workcell KPIs, to send it after that to the ERP. The ERP collects the workcell KPIs from all the SHs to evaluate the enterprise's overall performance. The cooperative workcell KPIs and the formulas that are used to calculate their value are studied in details Table 5.3 in section 5.4.3.
- **Production constrains:** one of the SH responsibilities is to ensure the continuous flow of the production in the workcell. Accordingly, the SH has to deal with the production constrains that are the main causes to stop the production. Therefore, The SH collects the production constrains from all the ORHs, and then takes the appropriate decisions based on them. For example, if a specific worker runs out of production components,

the SH would negotiate with the inventory to provide the missing components. In case of small-scale enterprise where the SH does not exist, the PH processes the workcell constrains instead, as shown in the case study in section 6.1.



Resource ID (**String**)	Operation Efficiency (**Float**)	Allocation Efficiency (**Float**)	Technical Efficiency (**Float**)	Utilization Efficiency (**Float**)	Production Rate (**Float**)
OR-ID:RX	%	%	%	%	Unit/min.
OR-ID:RW	%	%	%	%	Unit/min.

Constrain Type (**String**)	Resource ID (**String**)	Value (**Boolean**)
No Parts Available	OR-ID:RX	True
Machine Fail Down	OR-ID:WY	True

Figure 5.11: SH knowledge structure model - an example.

5.2.5 Non-Standard Holons

The HCA reference models use the previously described four standard holons to form their architecture. However, other holons can exist in the HCA to provide more services. For example, A **C**ustomer **H**olon (**CH**) has been used during the second and the third case studies of this dissertation, to initialize the customer orders. The CH can be a remote application that provides a GUI to the customer and helps to select and customize a customer order. Then the CH forwards the customer order to the MOM in case of a large-scale enterprise, or to the PH in case of a small-scale enterprise for further processing. Other holons such as an **I**nventory **H**olon (**IH**) or an **E**nergy **H**olon (**EH**) can be also connected to the PH or the SH, to manage the other facilities within the enterprise. Nevertheless, those holons are outside the scope of this dissertation.

5.3 *Concept Instantiation: Cooperative Workcell*

5.3.1 Cooperative Workcell Layout

The layout of the cooperative workcell dramatically influences its flexibility [GS17]. Thus, Figure 5.12 proposes a cooperative workcell layout that applies the proposed HCA in section 5.2. In order to make this layout clear, an electronic devices assembly scenario is considered. This assembly scenario has been adopted from a real-life application that uses a cobot in cooperation with a worker [You17c]. Although, a conventional IR can replace the cobot in this cooperative assembly scenario, by using one of the safety techniques that have been studied in the literature review in section 3.2.1. Thus, the general term robot is used to indicate a cobot or a cooperative IR. The proposed cooperative workcell is composed of six workstations. One workstation is dedicated to the robot. A workstation is dedicated to the assembly parts, and another to the required tools. Three workstations are dedicated to three workers. Nevertheless, the number of workers as well as their assembly tasks can change due to the production demands. The quantity and the customization of the product orders are up to the customers demand. The task of the robot in the proposed cooperative workcell is to handle the required parts to the workers, and if necessary, the tools based on the current production order. In this context, the robot is a very efficient operational resource for the following reasons:

1- **Shared operational resource:** the time needed from the robot to pick and place product parts is without a doubt less than the time needed from a worker to assemble a product. This means that the robot is operating most of the time as well as the workers. In other words, using the robot as a shared operational resource by all the workers, reduces the wasted operational time of all the resources.

2- **Better workcell design:** using a robot as a shared operational resource in the mentioned assembly scenario directly leverages three design values. First is the space, the shop floor area is a very expensive asset of the enterprise. Using a robot to directly handle parts or material does not only save a big space in comparison with a conveyor belt system, but also save the time of the manipulation. This is because, the robot follows a straight line or a curve path to deliver a part, while the conveyor belt is limited by its overall geometry. Second design value is the complexity, it is much easier to control and program a robot in comparison with a conveyor belt system. In order to control a conveyor system autonomously, every conveyor segment represents an ORH by its own (refer to [BV09]), which complicates the control development. The final design value is the cost, as the robots prices are much lower than a whole conveyor system.

3- **Worker focus:** in a cooperative assembly scenario, the worker's focus is only on the assembly task. In comparison with a manual assembly scenario, the worker has to look for the proper parts and tools to assemble a customized product. It is estimated in the research in [TLP+03] that 24.5% of the assembly time in automotive industry is wasted in task preparation. Moreover, memorizing the parts and the tools locations is a hideous mental overhead, which disturbs the worker's attention and drains his capabilities.

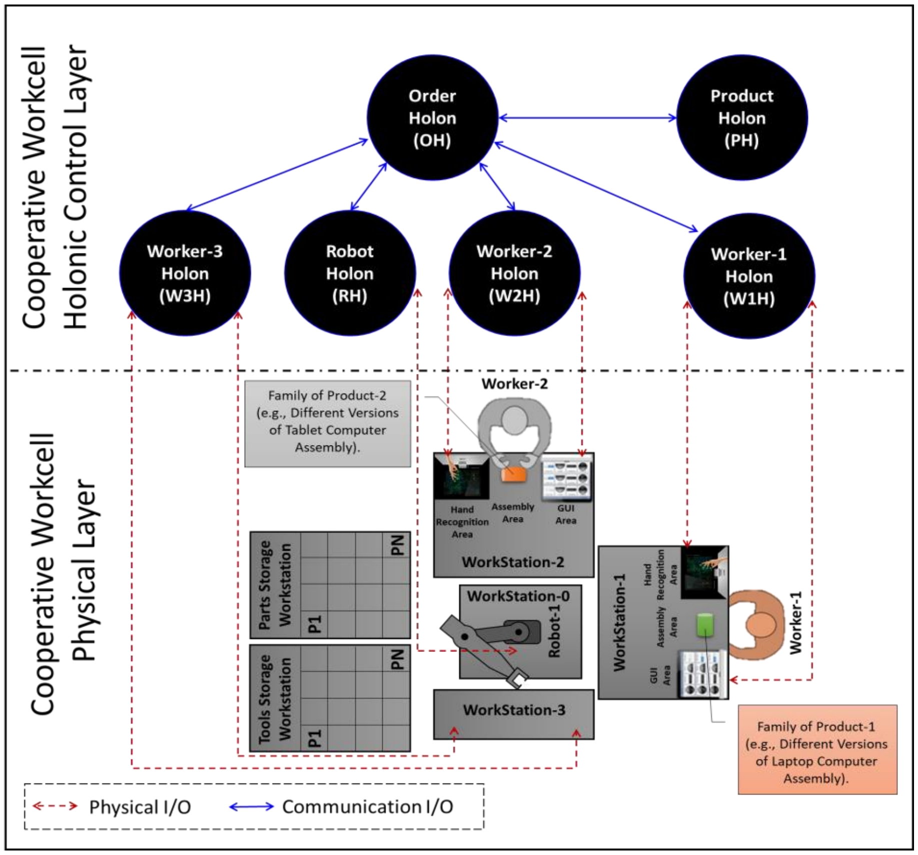


Figure 5.12: Cooperative workcell layout.

4- **Application suitability:** the cooperative robotics technology is still in progress. Accordingly, the available safe robots still have limitations such as the maximum weight lifting. Fanuc CR-35 (35 kg payload) is the highest payload cobot in nowadays market. However, the average payload among all the other cobots is around 1 kg (refer to Table 2.2 in section 2.1.2). This leads that some of the heavy industrial applications are still out of the cooperative manufacturing scope. Consequently, selecting the right application of the robot has a great effect on the cooperative workcell design. During the design of the proposed cooperative workcell, objects handling task was assigned to the robot, which is the most common application for the robot in the cooperative manufacturing. The same proposed workcell layout can be adapted to other cooperative manufacturing scenario, by replacing the robot handling task with another needed task.

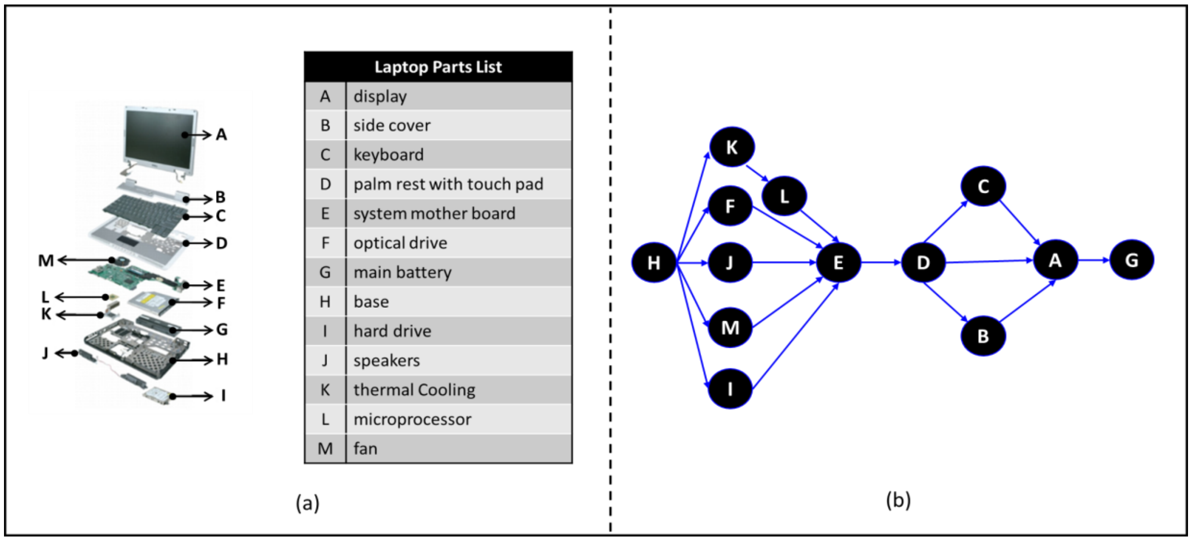
5.3.2 Cooperative Workcell Flexibility

Flexibility in manufacturing is the capacity of the system to change and assume different positions or states in response to the changing requirements with little penalty in time, effort, cost, and performance [TT98, Sal01]. The definition of flexibility in manufacturing is very generic and can lead to different levels of understanding. Therefore, this dissertation approaches the flexibility in cooperative manufacturing in two dimensions. The first dimension is the flexibility of the software component, which has been studied in chapter 4. Characteristics such as modularity, customizability, distributability, and extensibility enables to build a flexible control architecture. A flexible control architecture is the other half of the flexible cooperative workcell, as shown earlier in Figure 5.12. The combination between the cooperative workcell physical layer and control layer derives the second dimension of flexibility. Due to the research in [Elm05, Shi06, and Pes14], different types of flexibility can exist in production systems. Thus, the dissertation applies these types of flexibility to the cooperative workcell. The results are the aspects of flexibility in a cooperative workcell that are going to be discussed follows:

1- **Robot task definition:** two considerations must be taken when designing a robot task. First is the ability to add a new robot task to the cooperative manufacturing scenario at any time with the minimum interruption of the process. Most of the available commercial robots can be programmed by demonstration. This means that a worker without any programming background can move the robot arm in a specific path with a specific speed then the robot repeats the same motion profile. The idea is to store and announce a new robot task as a new operation which is available in the cooperative workcell, therefore it can be used whenever it is needed. The second consideration is using the robot as a multi-task shared operational resources. A robot is a shared operational resource when more than one worker are able to cooperate with it at the same time, as described earlier. Therefore, the robot must be able to perform more than one task due to the manufacturing context. For example, in the mentioned assembly scenario, a robot task might be to handle the product parts to the workers, after that the robot picks and places the complete product to stack it in the storage unit or to transfer it to a conveyor belt. Moreover, the robot can be used during the assembly scenario to pass the necessary assembling tools to the worker whenever it is needed.

2- **Worker's task time:** the cooperative workcell must be flexible to deal with the variations in the time needed from the worker to perform a specific task. In contrast with the robot, a worker always takes different time to perform the very same task. In many cases, the WT_P is needed to be known in advance, thus the workcell must be able to assess it. Therefore, the WT_P can be used via a decision-making algorithm, the production orders scheduling, or the production performance measuring.

3- **Worker interaction:** there are plenty of methods that can be used by the worker to interact with the manufacturing control system. GUI is one of the most common and simple methods to provide an interaction tool. A GUI is an excellent output to provide the worker with a graphical details of the assigned task, however the GUI lacks the characteristics to act as a good input device in the cooperative manufacturing context. This is because the GUI cannot sense the progress in the worker's task, then expresses the worker's status. Therefore, a combination between a GUI and another interaction method would be a better I/O approach. Hand gesture recognition along with the worker GUI have been used during this dissertation as a flexible interaction method. Hand gesture recognition has been selected in the context of the cooperative manufacturing as it is a nature method that the worker can use to interact with the robot in explicit or implicit modes. On the one hand, an implicit interaction mode takes place when a specific worker gesture is detected during the manufacturing. For instance, a pick and place gesture is unintentional implicit worker gesture that takes place during the assembly operation. Detecting the pick action means that the operation started, and detected the place action means that at the operation is in progress. On the other hand, explicit interaction mode occurs when some intentional gestures are detected. The explicit interaction can be used to give direct commands to the robot. For instance, to order the robot to bring a specific tool.



Laptop Parts List	
A	display
B	side cover
C	keyboard
D	palm rest with touch pad
E	system mother board
F	optical drive
G	main battery
H	base
I	hard drive
J	speakers
K	thermal Cooling
L	microprocessor
M	fan

Figure 5.13: (a) Laptop parts list– (b) Laptop assembly precedence graph - source [HKW+11].

4- **Operations sequence:** there is no guarantee that a worker will perform a process in the same steps all the time. Which makes the sequence of performing an operation is always varying. Furthermore, this parameter is getting exponentially sophisticated when dealing with more than one worker who perform similar operations. To illustrate this point, a laptop assembly recipe is explained in Figure 5.13. Figure 5.13a shows the parts of a laptop computer that can be assembled via the precedence graph which is shown in Figure 5.13b. Part-D cannot be assembled as long as part-E is not ready. To complete part-E assembly, many previous assembly steps must be completed. Therefore, the workcell must be flexible enough to tolerate different sequence of operations as long as they achieve the conditions of the assembly recipe (e.g., the precedence diagram).

5- **Operations dependencies:** the precedence diagram can only describe the relations and the sequence among the operations. However, other dependencies such as special assembly tools, screws, or nuts might also be necessary to complete the operation. Those dependencies differ from one operation to another. Therefore, the cooperative workcell must provide these variable dependencies along with the required assembly parts. Additionally, the cooperative workcell must monitor the tools and the other assembly dependencies. Thus, it can provide and manage them when it is needed.

6- **Product customization:** the cooperative workcell must be flexible to comprehend, execute, and manage different product recipes simultaneously. For example, Figure 5.12 proposes a cooperative workcell that is able to assemble two product families at the same time. The customization of the product starts from the customer who provides the minimum required details. Therefore, the cooperative workcell can build the product recipe and order. Then, it controls and directs the current operational resources based on their capabilities to produce that customer order. Conceptually, the cooperative workcell is designed to assemble a wide range of products. This can be achieved by switching the workers over the workstations. For instance, workstation-1 in Figure 5.12 has worker-1 who assembles laptop computers. But in another production shift, workerstation-1 may have worker-4 who assembles digital cameras. Also, the very same worker might have more than one capability. Thus, worker-1 might be able to assemble laptop computers and digital cameras concurrently. Switching among the workers is a flexible method to control the production volume as well, as it will be explained in the next points.

7- **Production volume:** the production volume is an important part of the customer order. For example, in the previously mentioned assembly scenario, the robot would handle to worker-1 (i.e., who is dedicated for laptop assembling) the right amount of parts in the right sequence due to the customer demand. The cooperative workcell must be able to increase or decrease its production capacity by controlling its operational resources, as will be explained in the next point.

8- **Workcell expansion:** normally a cooperative workcell has a maximum capacity because it contains a limited number of workstations. As mentioned before, it is less likely to need more robots than workers in a cooperative workcell. This is due to the fact that the robot is faster than the worker. Therefore, in the most common cases there are a fixed number of robots, who cooperate with a variable number of workers. This arrangement gives the cooperative workcell the capability to control its production volume by changing the number of workers due to the need. For example, workstation-3 is free to be used whenever the customer demand increases, as shown in Figure 5.12. Also, workstation-3 can be used to assemble a third product family (e.g., digital cameras), in order to efficiently use the area of the shop floor.

9- **Workcell layout:** workcell environment might be exposed to two sources of irregularities, which are the physical obstacles and the predefined positions changing. Irregularities in workcell environment are merely influencing the worker but the robot. In contrast to a conveyor belt system, a robot can easily maneuver around the obstacles. This can be done easily by representing the obstacles in an offline programing environment, or by teaching the robot alternative paths, as a robot can take more than one path to reach the same point. However, this must involve one of the physical safety methods that have been explained in the literature review in section 3.1.1. The second irregularity is changing the predefined positions of the objects. If a major change in the workcell layout happened, this would require teaching the new positions to the robot, and re-teach the old ones. Nevertheless, in case of minor changes, the robot can use another sensing method along with the position to look for the objects. For instance, most of the cobots are equipped by a camera which can be used in object detection, and hence compensates the position shifts.

10- **Shop floor expansion:** this aspect of flexibility is needed when two or more than two workcells are coordinating together. In the previous example, another workcell might be needed after the assembly stage to check the product quality. The cooperative shop floor must be able to expand horizontally to add a new process. For example, another workcell after the quality checking could be need for packaging the products. Also, the cooperative shop floor must be able to replicate the workcells which are doing the same process. For example, another assembly workcell might be needed to increase the production volume. In either cases, the workcell must be flexible in both the layout and the control system to tolerate the shop floor expansion. For instance, adding a new robot to the cooperative workcell who is responsible for transferring the finished product from the workcell to another, or adding a new task to an existing robot to transfer the finished product to the next workcell or to a conveyor belt.

The previously discussed flexibility aspects represent the conceptual guidelines to consider when designing a cooperative workcell. Because they dramatically influence both the workcell layout and the control system design. As an example, all the workstations in the assembly scenario in Figure 5.12 must have an automation device that permanently contains the physical layer of the ORH. Therefore, an operational resource uses this physical layer to register to or deregister from the cooperative workcell. Other examples that affect the flexibility of the control system will be illustrated in more details via the chapter 6. It is also worth to mention that the flexibility in a cooperative workcell has been illustrated through an assembly scenario example. As the assembly is the most common and the most time-wasting process in production due to Bosch Rexroth report [Bos05] and the research in [TLP+03]. Moreover, it is the easier to explain an assembly process than other manufacturing processes. However, the mentioned assembly example is objective and can be generalized over other cooperative scenarios that involve other manufacturing operations.

5.3.3 Cooperative Workcell Scheduling

One of the applications of the proposed HCA is the production scheduling. Production scheduling has a major impact on the enterprise productivity, as it lowers the production time and cost. Scheduling of similar cases of the cooperative workcell is commonly known in production engineering as the flow shop scheduling problem. Flow shop scheduling problem appears when a group of jobs share the same processing sequence on two or more machines sequentially [SN99]. Flow shop scheduling tries to optimize the sequence order of this group of jobs over the existing machines. The goal of flow shop scheduling is to obtain the continuity of the flow of the jobs over the machines. This can be obtained by minimizing the delays between two consequent jobs. Therefore, the overall makespan is minimized. A typical two sequential stages flow shop problem is solved by Johnson's scheduling [Grz16]. Johnson's scheduling model can be seen in Figure 5.14. The preconditions of this scheduling problem are that the production jobs are always processed by Machine-A (M-A) first, and then processed by Machine-B (M-B). The production speeds of M-A and B are different. Therefore, a buffer unit has to exist between M-A and B to recover this speed difference. Two kinds of delay can be found in this case, which as follows:

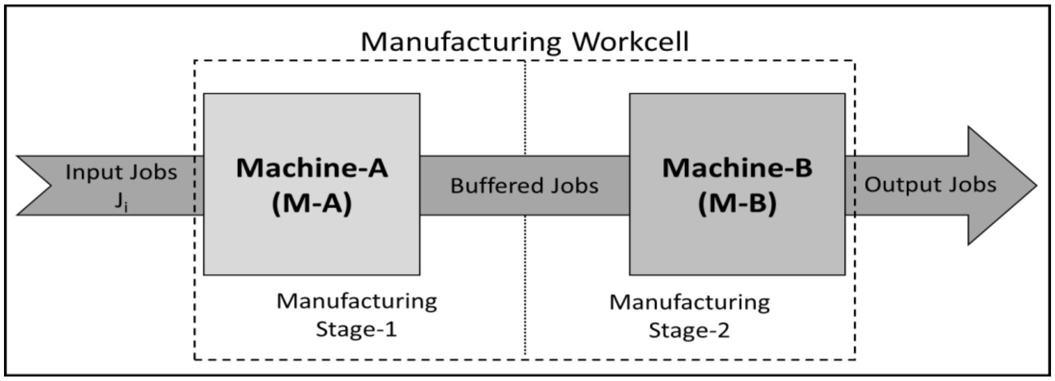


Figure 5.14: Sequential workcell model.

- **Starvation delay:** this means that M-B is free and waiting M-A to finish the current job to be able to start processing the same job. This delay must happen at least once at the first job assignment.
- **Buffer delay:** this delay happens when M-A has finished a job (J_i), while M-B is busy processing a prior job. Therefore, J_i has to wait in the buffer until M-B is free and J_i processing turn comes.

Table 5.1: Johnson's scheduling vs. First Come First Serve scheduling.

Unscheduled Jobs List			First Come First Serve Scheduling			Johnson Scheduling		
J_i	T_{Pi}@M-A	T_{Pi}@M-B	J_i	T_{AOi}@M-A	T_{AOi}@M-B	J_i	T_{AOi}@M-A	T_{AOi}@M-B
J_1	10	5	J_1	0:10	10:15	J_2	0:6	6:18
J_2	6	12	J_2	10:16	16:28	J_4	6:14	18:28
J_3	8	9	J_3	16:24	28:37	J_3	14:22	28:37
J_4	8	10	J_4	24:32	37:47	J_5	22:34	37:44
J_5	12	7	J_5	32:44	47:54	J_1	34:44	44:49

Johnson's scheduling minimizes the previous delays, to ultimately minimize the makespan of M-B that is the longest makespan of that manufacturing workcell [Alh97]. Johnson's scheduling can be summarized by the next three rules:

- **Rule-1:** find the shortest T_P among all the present non-scheduled standing jobs. If two or more jobs have the same T_P. Select one of them arbitrarily.
- **Rule-2:** if the shortest T_P locates on M-A, schedule the corresponding job as the soonest. If the shortest T_P locates on M-B, schedule the corresponding job as the latest.
- **Rule-3:** eliminate the job that has been scheduled from the present non-scheduled standing jobs. Repeat rules 1 and 2 till scheduling all the jobs.

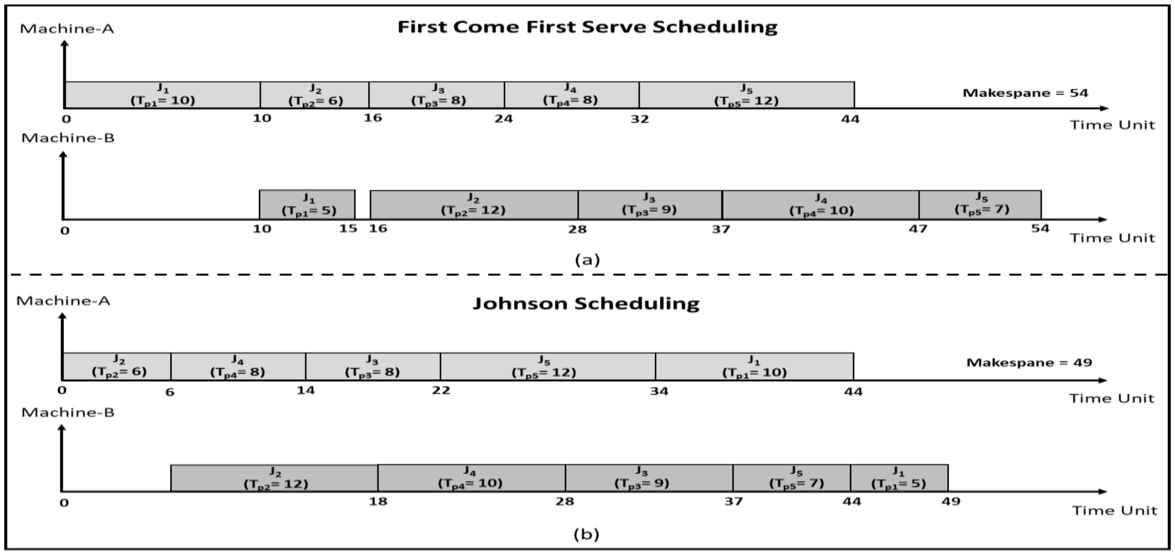


Figure 5.15: Gantt chart (a) First Come First Serve scheduling– (b) Johnson's scheduling.

The left section of Table 5.1 shows a group of unscheduled jobs J_i. In order to schedule those jobs using Johnson's rules, the shortest T_P must be found. In this case, the shortest T_P belongs to J_1 and equals to 5 time units, since this time locates on M-B, therefore it has to be scheduled at the end of the queue. J_2 comes after, with the shortest T_P on M-A, therefore it has to be scheduled at the beginning of the queue. Then, J_5 is the job with the shortest T_P on M-B, and J_5 goes before J_1. Finally, J_3 and J_4 are equal in the shortest T_P on M-A, therefore one of them is selected arbitrarily to come after J_2. In this example, J_4 has been selected to be scheduled first then J_3. In order to prove that Johnson's scheduling is the optimum solution, it is compared with the FCFS scheduling method, which can be seen at the middle section of Table 5.1 and in Figure 5.15a. By calculating the actual operation time (T_{AO}) based on the given T_P. It can be seen that the makespan of the FCFS scheduling is 54 time units. While, by applying Johnson's scheduling, the makespan is reduced to 49 time units, as shown at the right section of Table 5.1 and in Figure 5.15b.

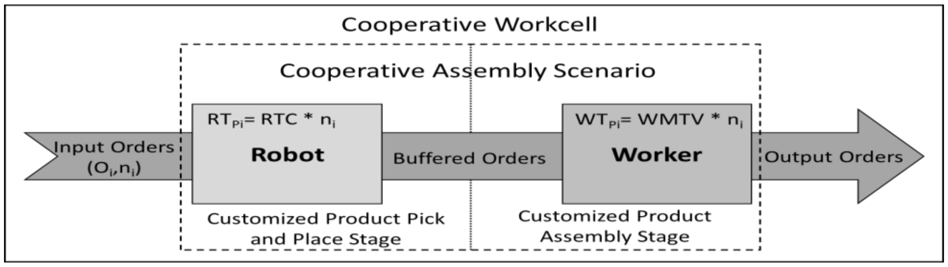


Figure 5.16: Cooperative workcell scheduling based on Johnson's rules.

The same concept of Johnson's scheduling can be applied over the cooperative workcell. Figure 5.16 proposes a cooperative assembly scenario based on the layout shown in Figure 5.14. In this scenario, the robot replaces M-A to pick and place the input order (O_i, n_i). Where, O_i denotes the customized production order, and n_i denotes the required number of units from this order. Then, the orders are stored in the buffer if it is necessary. Ultimately, the worker performs the assembly operation to produce the output orders.

Table 5.2: Johnson's scheduling based on comparing RTC to WMTV.

Unscheduled Jobs List			Johnson Scheduling Case-1: RTC ≤ WMTV			Johnson Scheduling Case-2: RTC ≥ WMTV		
$J_i = (O_i, n_i)$	RT_{Pi}	WT_{Pi}	RT_{Pi}	WT_{Pi}	RT_{Pi}	RT_{Pi}	WT_{Pi}	RT_{Pi}
$J_1 = (O_1, n_1)$	RTC * n_1	WMTV * n_1	J_1	RTC * n_1	WMTV * n_1	J_2	RTC * n_2	WMTV * n_2
$J_2 = (O_2, n_2)$	RTC * n_2	WMTV * n_2	J_2	RTC * n_2	WMTV * n_2	J_1	RTC * n_1	WMTV * n_1

At the first stage of the cooperative workcell, a **R**obot **T**ime **C**onstant (**RTC**) is assumed as the time taken from the robot to pick and place the product parts. Thus, the Robot T_P to pick and place a customized order (RT_{Pi}) can be obtained from the following equation:

$$RT_{Pi} = RTC \times n_i \tag{5.1}$$

At the second stage of the cooperative workcell, a **W**orker **M**ean **T**ime **V**alue (**WMTV**) can be assumed as the average time of the worker to assemble one product. Thus, the T_P required by the worker to assemble a customized order (WT_{Pi}) can be obtained from the following equation:

$$WT_{Pi} = WMTV \times n_i \tag{5.2}$$

Johnson's scheduling is mainly depending on finding the shortest T_P, which makes it easier to apply Johnson's rules based on the previous assumptions. This is because RT_{Pi} and WT_{Pi} are correlated to each other by n_i. Therefore, Johnson's scheduling can be obtained by comparing RTC to WMTV as shown in Table 5.2. In this table, if it is assumed that n_1 is less than n_2 and RTC is less than or equal to WMTV. Then RT_{P1} must be the shortest unscheduled T_P. Therefore, the sequence of the scheduled jobs will be J_1 then J_2. In the same manner, if it is assumed that RTC is greater than or equal to WMTV, this means that WT_{P1} must be the shortest unscheduled T_P. Therefore, the sequence of the scheduled jobs will be J_2 then J_1. From this comparison, Johnson's rules can be modified to schedule the jobs based on the ascending sequence of n_i, if RTC is less than or equal to WMTV. Otherwise, scheduling of the jobs is based on the descending sequence of n_i.

5.4 Concept Scaling: Cooperative Enterprise

5.4.1 Cooperative Enterprise Holarchy

The holon concept in cooperative manufacturing has been studied from two different perspectives so far. The first perspective is the holon structure as a software component, while the second perspective is the holon object-oriented model based on its functions. Both of these perspectives supported the development of the cooperative workcell concept. However, on the cooperative enterprise level, it is important to consider the distribution of the holons over the different layers of the industrial enterprise. Therefore, the coordination among the cooperative workcells is achieved. Moreover, the integration between the proposed HCA and the other enterprise software such as the MOM and ERP is defined. To distribute the proposed HCA over the cooperative enterprise, the industrial enterprise must be modelled into a layered structure. ISA-95 is the most common standard that is concerned with the industrial enterprise modelling. The main purpose of ISA-95 standard is to define a model that facilitates the integration of the automated control solution with the other layers of the industrial enterprise. In addition, ISA-95 model aims to extend and upgrade the existing enterprise software and hardware infrastructure by recommending platform independent technologies such as XML. ISA-95 divides the industrial enterprise into four layers that can be seen in Figure 5.17 and summarized as follows:

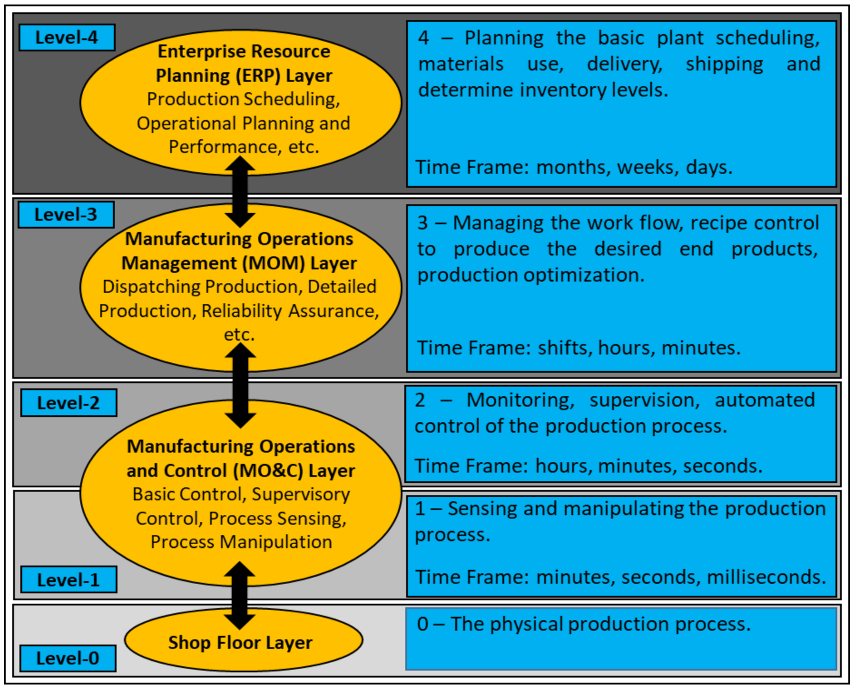

Level-4
Enterprise Resource Planning (ERP) Layer
Production Scheduling, Operational Planning and Performance, etc.
4 – Planning the basic plant scheduling, materials use, delivery, shipping and determine inventory levels.
Time Frame: months, weeks, days.
Level-3
Manufacturing Operations Management (MOM) Layer
Dispatching Production, Detailed Production, Reliability Assurance, etc.
3 – Managing the work flow, recipe control to produce the desired end products, production optimization.
Time Frame: shifts, hours, minutes.
Level-2
Manufacturing Operations and Control (MO&C) Layer
Basic Control, Supervisory Control, Process Sensing, Process Manipulation
2 – Monitoring, supervision, automated control of the production process.
Time Frame: hours, minutes, seconds.
Level-1
1 – Sensing and manipulating the production process.
Time Frame: minutes, seconds, milliseconds.
Level-0
Shop Floor Layer
0 – The physical production process.

Figure 5.17: Enterprise levels based on ANSI/ISA-95 standard reference model [ISO14].

1- **Shop floor layer:** is the ground level (i.e. level-0) of the industrial enterprise, where the actual production takes place. The shop floor layer contains all the physical operational resources and the actual product.

2- **Manufacturing operations and control layer:** this layer is directly supervising and controlling the production processes. ISA-95 standard divides this layer into two levels. Level-1 contains the sensing devices and the automation controllers, which are used to provide the inputs to or obtain the outputs from the physical operational resources. The time frame in the level varies from milliseconds to minutes based on the nature of the controlled process. Level-2 of ISA-95 standard monitors the inputs that are generated by the sensing devices at level-1, and controls the outputs that are commending to the automation controllers at level-1 of ISA-95. The time frame within level-2 of ISA-95 can be as short as seconds and can extend to hours.

3- **Manufacturing operations management layer:** MOM layer has a variety of responsibilities in the industrial enterprise. MOM directly manags the manufacturing processes by interacting with Level-2 of ISA-95 standard. Therefore, it ensures the correct setup and configuration of the sensing and automation devices. The MOM layer controls and schedules the production flow and optimize the manufacturing cost. Also, it collects all the related production history. The time frame in the MOM layer can extend from minutes to days.

4- **Enterprise resource planning layer:** ERP layer is the central core of the enterprise business. ERP layer collects the production information from Level-3 of ISA-95 standard, then parses this information to business related knowledge. Accordingly, ERP is responsible for planning the whole manufacturing execution. As it coordinates the manufacturing execution with all the other related business units such as the inventory, the finance, and the energy. The time frame in the ERP layer can extend from days to months.

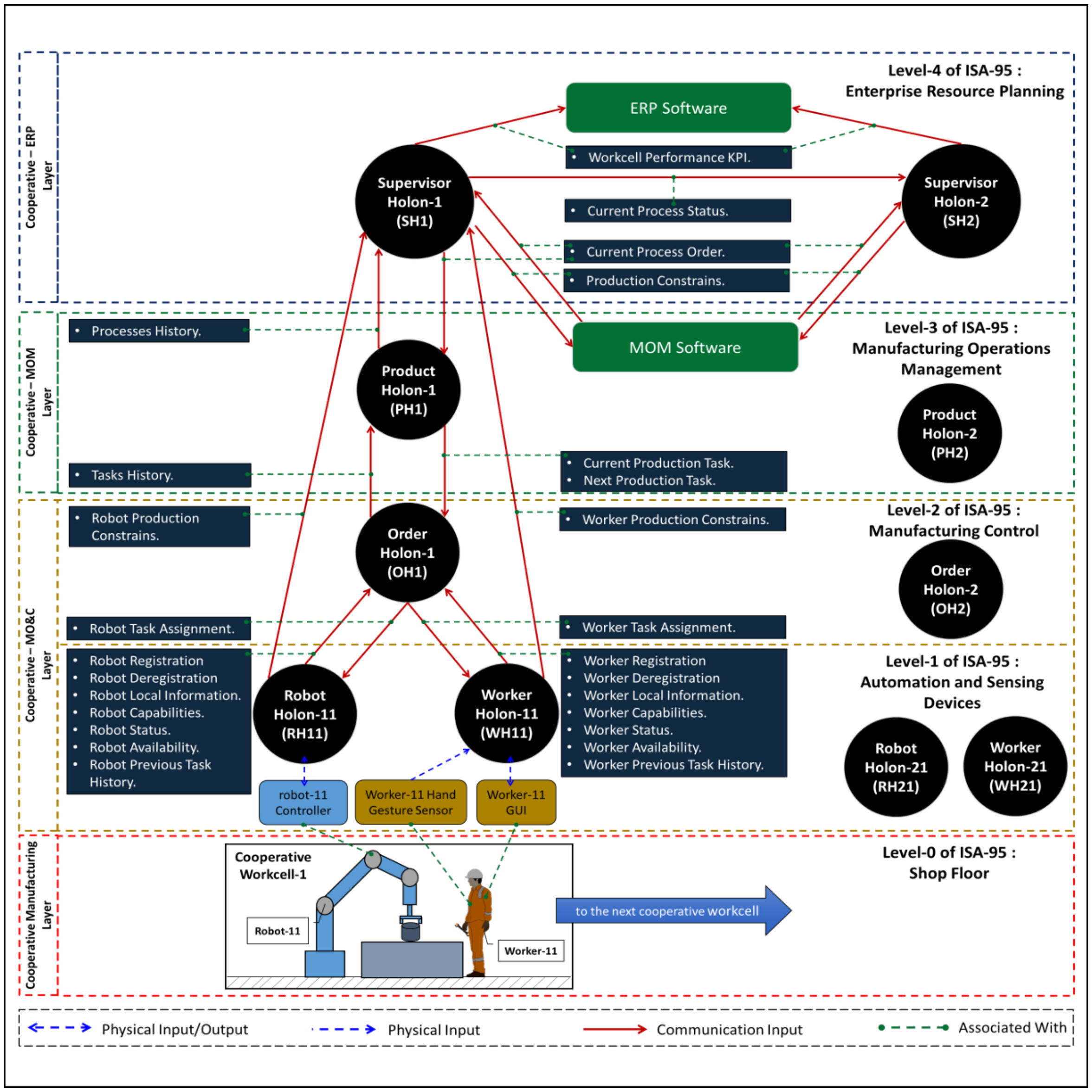


Figure 5.18: Cooperative enterprise holarchy.

ISA-95 standard model has been fused with the previously developed holon model to derive the cooperative enterprise holarchy that is shown in Figure 5.18. Similar to ISA-95 model, the cooperative enterprise holarchy contains four levels. Level-0 of the cooperative enterprise consists of a group of a cooperative workcells. Every cooperative workcell contains at least one worker, one robot, and the other production components. The number of the cooperative robots and workers is varying based on the workcell layout and the production demands. In addition, the number of active cooperative workcells over the shop floor may vary upon the production need.

Level-1 of the cooperative enterprise contains the ORHs. The physical component of an ORH is always connected to the associated automation or sensing devices. For example, in the robot case, the physical layer of the RH is connected with the robot's controller. While in a worker case, the physical layer of the WH is connected with a gesture recognition sensor and a GUI. The communication component of an ORH is only created when an operational resource registers itself to the workcell. Subsequently, the communication component of an ORH is deleted when an operational resource deregisters itself from the workcell. A cooperative workcell contains only one OH, whom connected to all the ORHs. Creating or deleting the ORHs is done by the OH, whom locates at level-2 of the cooperative enterprise. Both the RH and the WH report the manufacturing information to the OH, and receive the task assignment from the OH as explained in details in sections 5.2.1 and 5.2.2.

Level-3 of the cooperative enterprise contains the PHs. Every cooperative workcell contains only one PH. The PH can manage more than one product or process simultaneously at the same cooperative workcell. The PH directly interacts with the SH, whom locates in level-4 of the cooperative enterprise to obtain the current production process. Then, the OH converts the process recipe into tasks that are assigned to the ORHs as explained in details in section 5.2.3. Every workcell contains only one SH that coordinates with the other SHs in the other workcells to organize the production flow. The SH interacts with the MOM to obtain the assigned process, and hence it builds the process recipe. Also, The SH informs the MOM with all the related manufacturing information such as the production constrains and the energy consumption. Moreover, the SH reports to the ERP with all the related business information such as the KPIs.

ISA-95 model is not the only standard that supports the industrial enterprise integration and digitalization. Other standards such as reference architectural model industry 4.0 [Pho16] and the automation pyramids [MPM17] are currently in progress, to fit the new requirements of the smart factory. These standards are dividing the industrial enterprise in very similar layers to ISA-95. The main reason to use ISA-95 in this dissertation is the maturity of this standard as it has been experimented in a variety of industrial fields. Also, ISA-95 provides solid definitions of the terms that are used in the production control. Those terms were adapted to fit the cooperative manufacturing as discussed earlier in section 5.1.

5.4.2 Cooperative Enterprise Re-configurability

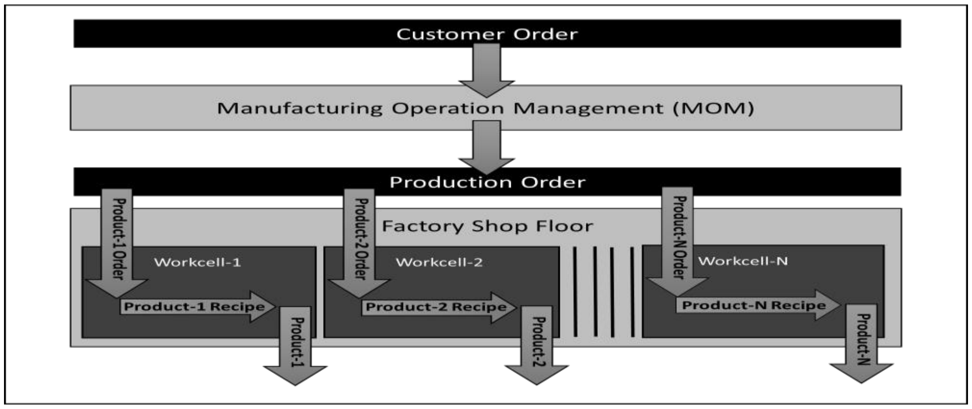


Figure 5.19: Product family production via cellular manufacture concept.

Re-configurability in manufacturing is the ability to support different manufacturing system configurations, i.e., different manufacturing scenarios with small customization tasks [Lei04]. The difference between an FMS and an RMS might be obscure, due to the shared features between the two systems. However, the distinction between them can be understood from their goals. On the one hand, FMS is willing to produce a product from scratch, which requires very high level of customizability of the final product, thus the production volume is limited. On the other hand, an RMS aims to customize some features of a product to produce what is so-called a product family, thus the production volume is moderate. The same concept of cellular manufacture that has been explained at the beginning of this dissertation is applied to manufacture a product family. Assuming a product order with only one process, which creates a reconfigurable shop floor that is capable of producing a product family, as shown in Figure 5.19. In Figure 5.19, every workcell operates independently from the previous and can manufacture one or more product.

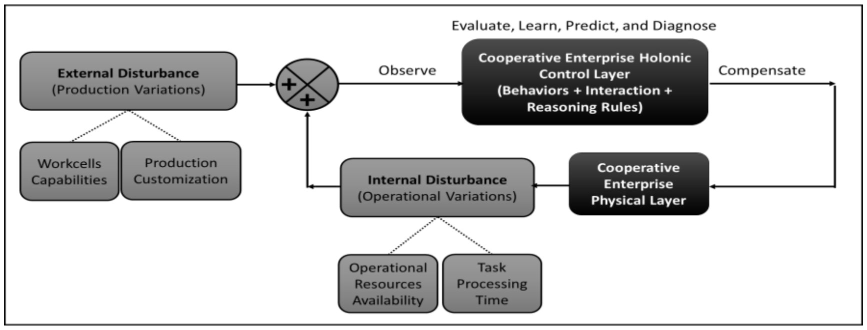

Evaluate, Learn, Predict, and Diagnose
External Disturbance (Production Variations)
Workcells Capabilities
Production Customization
Observe
Cooperative Enterprise Holonic Control Layer (Behaviors + Interaction + Reasoning Rules)
Compensate
Internal Disturbance (Operational Variations)
Cooperative Enterprise Physical Layer
Operational Resources Availability
Task Processing Time

Figure 5.20: Self-organisation in the cooperative enterprise.

Cooperative enterprise self-organisation is the ultimate goal of the proposed HCA to achieve the re-configurability. Self-organization is a spontaneous trend that exists in nature among different organisms. The generic definition of the self-organization concept is the process where some form of an overall order arises in a group due to the local interaction among the members of this group [SIK06]. In control systems engineering, self-organization refers to the interaction mechanisms that are followed by the control solution to compensate the system disturbances. Disturbance are the unexpected variations that take place during the production. Figure 5.20 entitles two types of disturbance in the cooperative enterprise, which are internal and external. Internal disturbance results from the variations in the operation such as the change in the operational resources availability and the task TP. External disturbance results from the variations in the production demands such as the production customization and the workcells capabilities.

The sum of the external and internal disturbances is compensated at the real-time of the production via the holons interaction. Compensation of the disturbance means to take the set of decisions that optimize the overall performance of the cooperative enterprise. This compensation occurs in different forms. First, the HCA evaluates the current situation of the cooperative enterprise by processing the incoming information. Therefore, it learns the different trends that take place by analysing the processed information. Then, it uses this knowledge to predict the future behaviours of the cooperative enterprise, and to diagnose the current status as well. The proposed HCA uses a combination of the holons behaviours and interaction, in addition to the reasoning rules to compensate the cooperative enterprise disturbance. The interaction is achieved in forms of control events, string-based ACL-Messages, or ontology-based ACL-Messages. In most of the cooperative scenarios, patterns of the three mentioned forms of interaction are combined. Compensating the cooperative enterprise disturbances via the interaction will be discussed as follows:

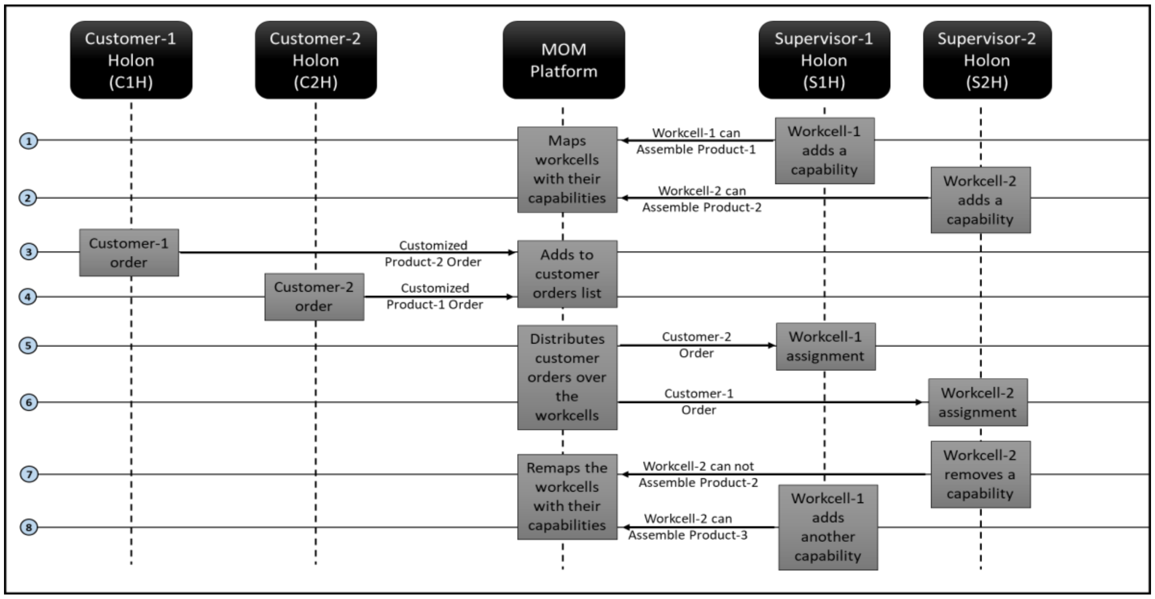

Customer-1 Holon (C1H)
Customer-2 Holon (C2H)
MOM Platform
Supervisor-1 Holon (S1H)
Supervisor-2 Holon (S2H)
1
2
3
4
5
6
7
8
Maps workcells with their capabilities
Workcell-1 can Assemble Product-1
Workcell-1 adds a capability
Workcell-2 can Assemble Product-2
Workcell-2 adds a capability
Customer-1 order
Customized Product-2 Order
Adds to customer orders list
Customer-2 order
Customized Product-1 Order
Distributes customer orders over the workcells
Customer-2 Order
Workcell-1 assignment
Customer-1 Order
Workcell-2 assignment
Remaps the workcells with their capabilities
Workcell-2 can not Assemble Product-2
Workcell-2 removes a capability
Workcell-2 can Assemble Product-3
Workcell-1 adds another capability

Figure 5.21: Compensation of the workcells capabilities variation via holons interaction.

1- **Workcells capabilities:** workcells capabilities are the first source of external disturbances that are compensated by the proposed HCA. In the context of reconfigurable concept, a cooperative workcell can have more than one capability. These capabilities vary when adding a new capability or removing an old capability during the production. Therefore, the HCA must update all the workcells capabilities, in order to match this variation with the customer orders. In other words, if the MOM always knows the workcells capabilities, it will be able to distribute the customer orders over the workcells based on their present capabilities. Figure 5.21 shows an interaction example that contains two workcells. At the very beginning of the production, the SH of every workcell must inform the enterprise MOM about their capabilities. For example, workcell-1 can assemble product-1 and workcell-2 can assemble product-2, as it can be seen from step-1 and step-2 of the interaction. In step-3 and step-4, two customized orders are received by the MOM. Accordingly, by matching the customer orders to the workcells capabilities, the MOM distributes the customer orders over workcells in step-5 and step-6 of the interaction. In step-7, workcell-2 informs the MOM that it cannot assemble product-2 anymore. While in step-8, workcell-1 informs the MOM that it can assemble a new product that is product-3.

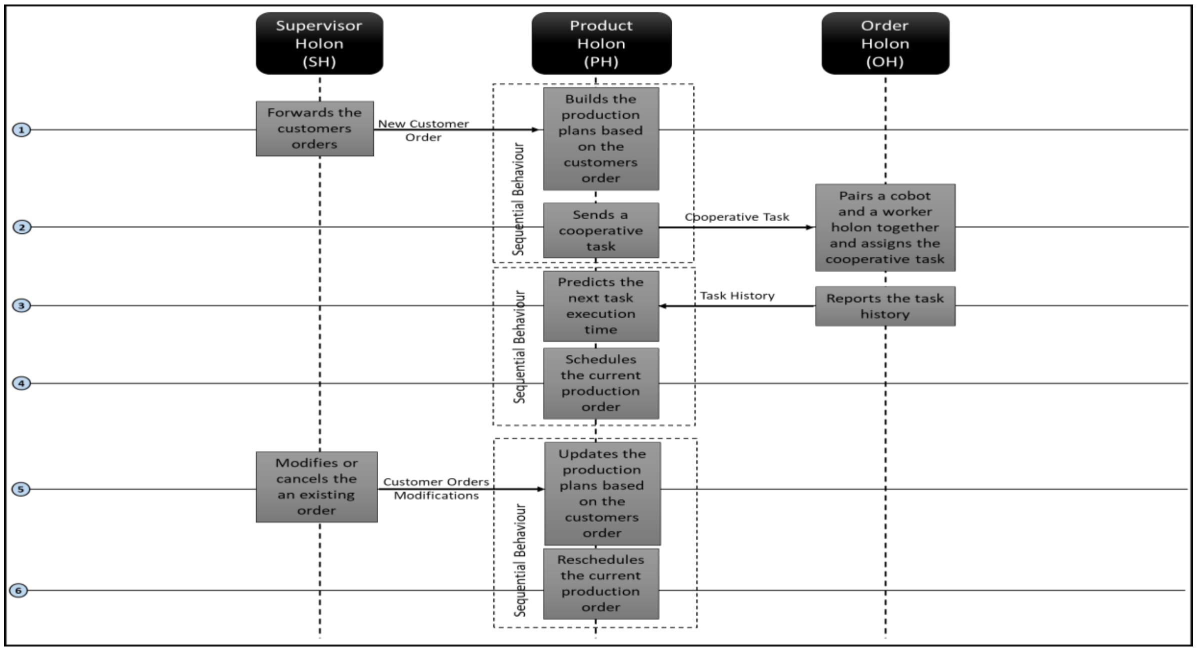

Supervisor Holon (SH)
Product Holon (PH)
Order Holon (OH)
1
Forwards the customers orders
New Customer Order
Builds the production plans based on the customers order
Sequential Behaviour
2
Sends a cooperative task
Cooperative Task
Pairs a cobot and a worker holon together and assigns the cooperative task
3
Predicts the next task execution time
Task History
Reports the task history
Sequential Behaviour
4
Schedules the current production order
5
Modifies or cancels the an existing order
Customer Orders Modifications
Updates the production plans based on the customers order
Sequential Behaviour
6
Reschedules the current production order

Figure 5.22: Compensation of the production customization variation via holons interaction.

2- **Production customization:** production customization is the second source of external disturbances that are compensated by the proposed HCA. In the context of reconfigurable concept, a cooperative workcell must be able to receive, cancel, or modify the customer orders during the real production time. Therefore, the HCA must update all the customer orders, in order to schedule and manage the production. Figure 5.22 shows an interaction example that contains one workcell. In step-1 of the interaction, the SH forwards all the customer orders to PH. In step-2, the PH forms a cooperative task and sends it to the OH. Then, the OH assigns the cooperative task to a cooperative pair and reports the task history to the PH, as shown in step-3. The PH uses the task history to predict the next task T_P. Therefore, it schedules the production in step-4 of the interaction. When a new customer order arrives or an old customer order is cancelled or modified, the PH updates and reschedules the production list, as shown in step-5 and step-6 of the interaction.

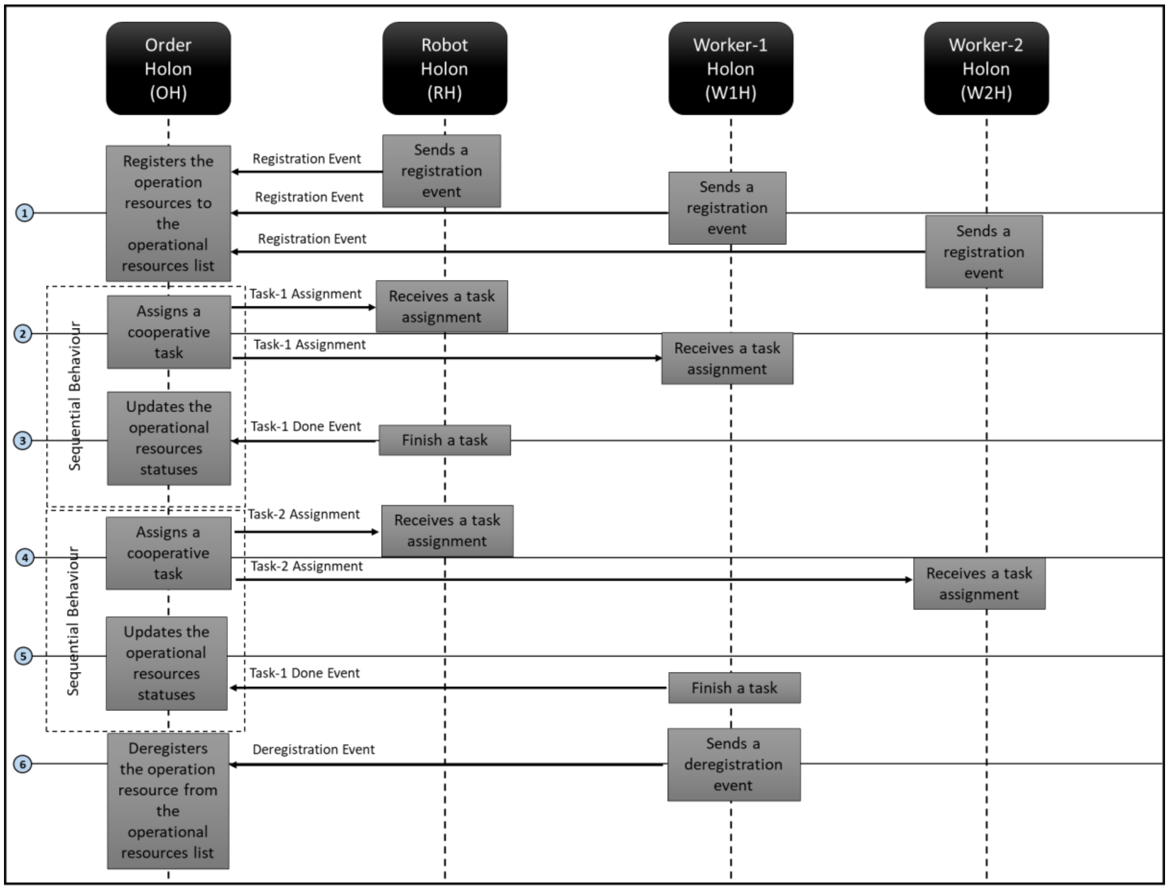

Order Holon (OH)
Robot Holon (RH)
Worker-1 Holon (W1H)
Worker-2 Holon (W2H)
Registers the operation resources to the operational resources list
Registration Event
Sends a registration event
Registration Event
Sends a registration event
Registration Event
Sends a registration event
Sequential Behaviour
Assigns a cooperative task
Task-1 Assignment
Receives a task assignment
Task-1 Assignment
Receives a task assignment
Updates the operational resources statuses
Task-1 Done Event
Finish a task
Sequential Behaviour
Assigns a cooperative task
Task-2 Assignment
Receives a task assignment
Task-2 Assignment
Receives a task assignment
Updates the operational resources statuses
Task-1 Done Event
Finish a task
Deregisters the operation resource from the operational resources list
Deregistration Event
Sends a deregistration event
1
2
3
4
5
6

Figure 5.23: Compensation of the resources availability variation via holons interaction.

3- **Operational resources availability:** operational resources availability is the first source of internal disturbances that are compensated by the proposed HCA. The availability of the operational resources varies due to different events. The first event is generated when registering a new operational resource to the workcell, or deregistering an existing operational resource from the workcell. Registering event happens if a new operational resource joins the workcell. While, deregistering event must happen if an existing operational resource permanently leaves the workcell. The second event is generated when a registered operational resource is unavailable. For example, a worker could be unavailable during the rests or the breaks, or a robot is unavailable during its maintenance or reprogramming. The third even is generated when the operational resource is available but busy while performing a task. Accordingly, the HCA must be able to recognize and distinguish these events from each other. Thus, it can assign the cooperative task. Figure 5.23 shows an interaction scenario which assumes one cooperative workcell that contains two workers and one robot. In step-1 of the interaction, all the operational resources at the workcell (i.e., W1H, W2H, and RH) are registering to the OH. When the OH knows the availability of the operational resources, it assigns the cooperative tasks for them. Therefore, the OH pairs between W1H and RH and assigns a cooperative task-1, as it can be seen in step-2 of the interaction. Then, the OH updates the status of W1H and the RH from free to busy. When the OH receives task-1 done event from RH in step-3, it pairs between W2H and RH and assigns a cooperative task-2, as it can be seen in step-4 of the interaction. In step-5 of the interaction, W1H sends task-1 done event to inform the OH that task-1 is done. Therefore, the OH changes W1H status from busy to free. In step-6, worker-1 decides to leave the workcell, therefore it sends a deregistration event to the OH. Thus, the OH removes worker-1 from the operational resource list.

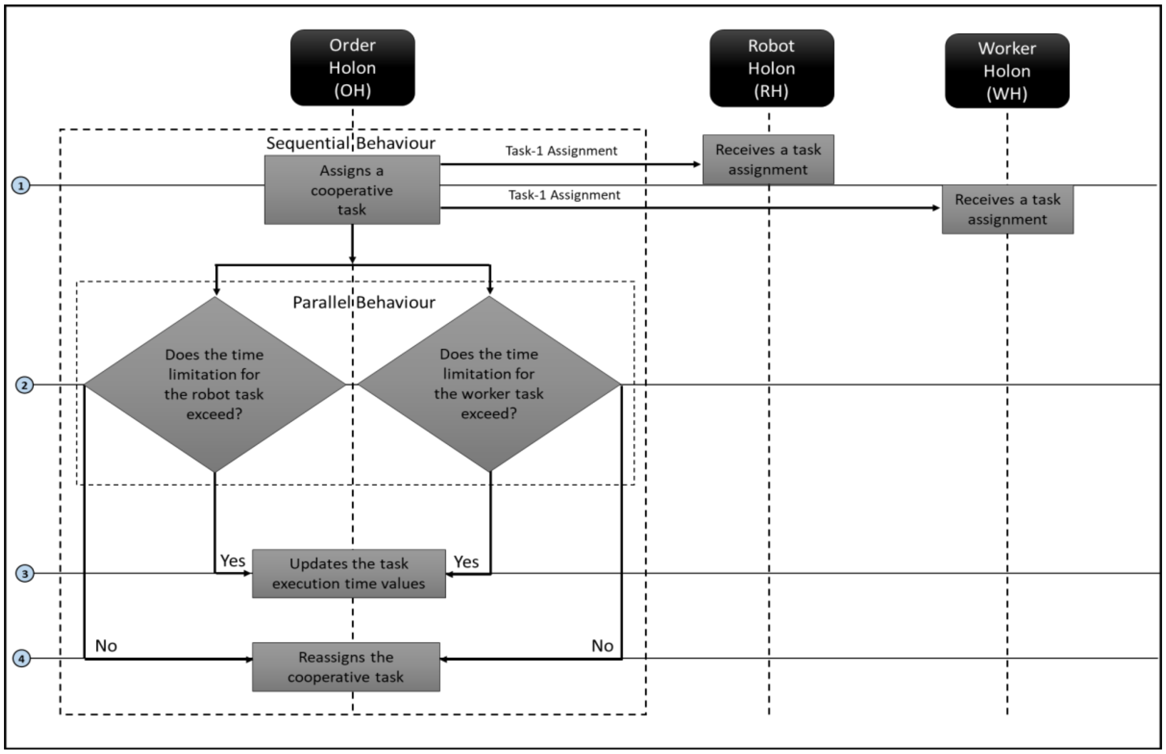

Order Holon (OH)
Robot Holon (RH)
Worker Holon (WH)
Sequential Behaviour
Assigns a cooperative task
Task-1 Assignment
Receives a task assignment
Task-1 Assignment
Receives a task assignment
Parallel Behaviour
Does the time limitation for the robot task exceed?
Does the time limitation for the worker task exceed?
Yes
Updates the task execution time values
Yes
No
Reassigns the cooperative task
No
1
2
3
4

Figure 5.24: Compensation of the task processing time variation via holons interaction.

4- **Task processing time:** task T_P is the second source of internal disturbances that are compensated by the proposed HCA. The WT_P is stochastic. This is because the worker is always taking a different amount of time to execute the same task. This factor is magnified when considering different tasks that are performed with many workers. In the robot case, the **R**obot **P**rocessing **T**ime (**RT_P**) can be more stable. This is because the robot's speed is reliable and can be known in advance, therefore the RT_P is known in advance. Another problem regarding the task T_P may appear during the production for both the worker and the robot. This problem is the delay of the task T_P. This problem appears when assigning task to an operational resource, however the operational resource would not finish the task. For example, a robot could be stuck but the control system does not realize that. Therefore, a task assignment would be sent to the robot, but never get a task done event. The same scenario can happen with the worker as well. The solution of the task delay problem is that the HCA estimates the maximum time needed to perform a specific task before assigning it. Therefore, if the operational resource did not respond within a time limitation after the task assignment. The HCA reassigns the task, and reports the problem with this operational resources. To estimate a task T_P, the HCA records the previous T_P history. Therefore, it learns the task T_P trend, and be able to diagnose the operational resources failures. Figure 5.24 shows an interaction scenario which assumes one cooperative workcell that contains one worker and one robot. In step-1 of the interaction, the OH assigns a cooperative task-1 for both the WH and the RH. Then, the OH launches a parallel behaviour which contains two different time limitation values for both the worker's task and the robot task, as shown in step-2 of the interaction. If both of the operational resources finished their tasks within the time limitation. The OH would update the real value of the time which has been taken to perform the task from both the worker and the robot. Otherwise, if one of the cooperative operational resources did not respond within the T_P limitation. The OH resource must report an operational resource failure and reassign the task to another available operational resources, or waits till the failure is fixed.

5.4.3 Cooperative Enterprise Performance Measuring

Performance measuring is the process of quantifying the effectiveness and the efficiency of the past operations [NMP+00]. KPIs such as efficiency, availability, and productivity are commonly used to reflect the industrial operation performance. KPIs give a clear insight of the potential improvements at the industrial enterprise. Furthermore, the HCA can use many of these indicators during the decision-making process. ISO 22400 standard defines 34 quantitative and qualitative KPIs, which measure the enterprise strategic success factors [ISO14]. These KPIs are divided into four categories that are production KPIs, quality KPIs, maintenance KPIs, and inventory KPIs as shown in Figure 5.27.

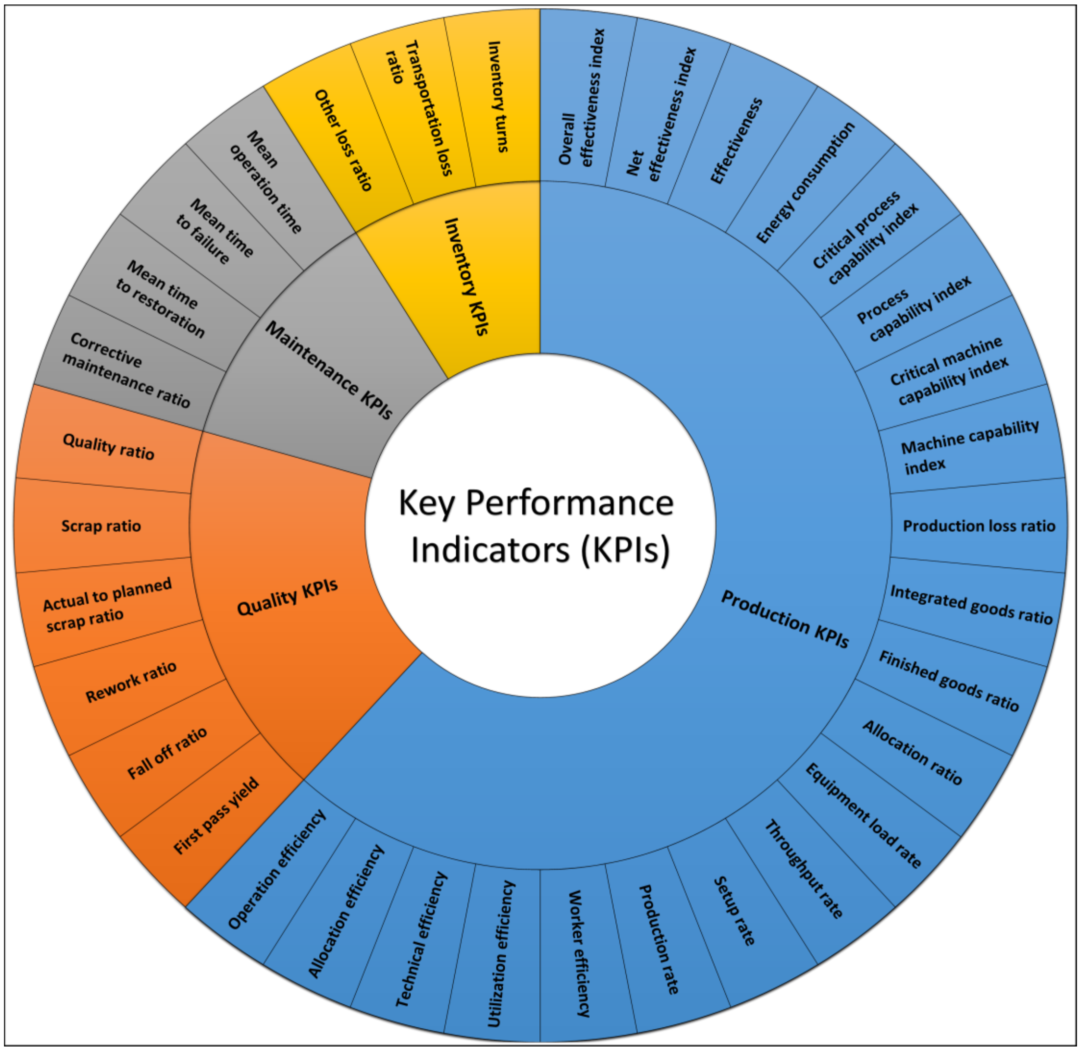


Figure 5.25: Key Performance Indicators (KPIs) due ISO 22400 [Joh14], [WHP13].

As shown in Figure 5.25, the KPIs are generic and broad. Thus, this dissertation spots only the most related to cooperative manufacturing, which are focused on production. In order to obtain these production KPIs, it is essential to define the basic production elements that are used to calculate the production KPIs [KZL+16]. These elements are the following:

- **Actual Operation Time (T_{AO}):** the actual time taken by the resource to perform the needed tasks to produce a production order.
- **Setup Time (T_S):** the time taken by the resource be setup before producing a production order - if the setup is needed.
- **Processing Time (T_P):** the time taken by the resource to produce a production order, including the setup time.

$$T_P = T_{AO} + T_S \tag{5.3}$$

- **Down Time (T_D):** the time wasted by the resource due to breaks, rests, maintenance, programming, etc.
- **Busy Time (T_B):** the time that the resource is busy performing a specific production order, including the down time.

$$T_B = T_P + T_D \tag{5.4}$$

- **Idle Time (T_I):** the time when the resource is idle due to the starvation delays.
- **Overall Operation Time (T_{OO}):** the overall time taken by the resource to finish a production order, including the setup time, the down time, and the idle time.

$$T_{OO} = T_B + T_I \tag{5.5}$$

Thus, based on the above basic production elements, the following production KPIs can be calculated:

1- **Operation Efficiency (E_O):** the percentage that represents the actual resource operation time with respect to the overall needed time to finish the production orders. Operation Efficiency is also known as the availability.

$$E_O = \frac{T_{AO}}{T_{OO}} \text{ X } 100\ \% \tag{5.6}$$

2- **Allocation Efficiency (E_A):** the percentage that represents the actual usage of the resource with respect to the overall needed time to finish the production orders.

$$E_A = \frac{T_B}{T_{OO}} \text{ X } 100\ \% \tag{5.7}$$

3- **Technical Efficiency (E_T):** the percentage that represents the wasted time due to the resource down time.

$$E_T = \frac{T_{AO}}{T_{AO} + T_D} \text{ X } 100\ \% \tag{5.8}$$

4- **Utilization Efficiency (E_U):** the percentage that represents the productivity of the resource.

$$E_U = \frac{T_{AO}}{T_B} \text{ X } 100\ \% \tag{5.9}$$

5- **Production Rate (R_P):** the quantity of products that produced per a fixed time interval. Alternatively, the amount of time that is taken to finish one product.

$$R_P \text{ per hour} = \frac{\text{Number of Finished Products}}{\text{hour}} \tag{5.10}$$

The previous production KPIs can be applied over any operational resource within the cooperative workcell (i.e. robot, worker, or machine) while considering the difference in the nature of the terms with respect to every resource. For example, T_D in the robot case could result of its maintenance. However, in the worker case, T_D could be a result of breaks and rests. In case of the worker or the robot, T_S is irrelevant. Therefore, it can be always assumed it equals to zero. Thus, the relationship among the production time element can be found from Figure 5.26.

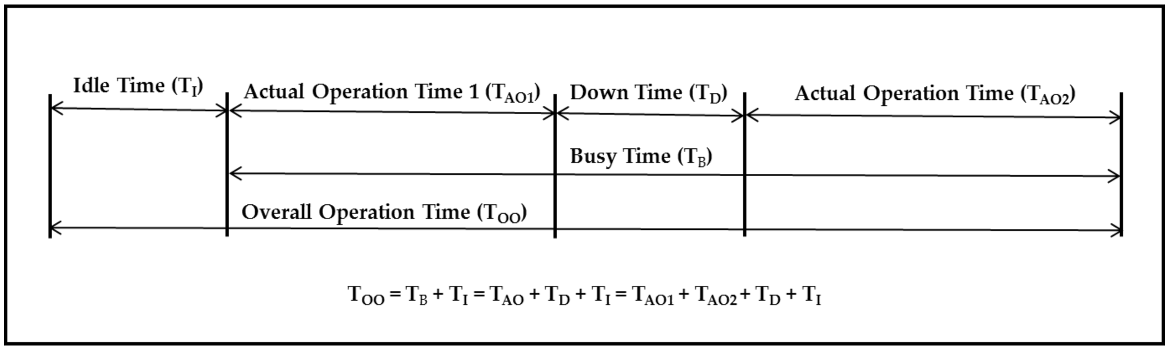

Idle Time (T_I)
Actual Operation Time 1 (T_{AO1})
Down Time (T_D)
Actual Operation Time (T_{AO2})
Busy Time (T_B)
Overall Operation Time (T_{OO})
$T_{OO} = T_B + T_I = T_{AO} + T_D + T_I = T_{AO1} + T_{AO2} + T_D + T_I$

Figure 5.26: Relation among the production time elements.

Assuming that the T_S for the worker or the robot is zero, it can be concluded that E_T and E_U are equal to each other. Furthermore, the previously stated production KPIs can be dedicated to either the worker or the robot as shown in Table 5.3.

Table 5.3: Robot and worker KPIs.

KPI \ Resource	Worker	Robot
Operation Efficiently	$WE_O = \frac{WT_{AO}}{WT_{OO}}$ X 100 % = $\frac{WT_{AO}}{WT_I + WT_D + WT_{AO}}$ X 100 %	$RE_O = \frac{RT_{AO}}{RT_{OO}}$ X 100 % = $\frac{RT_{AO}}{RT_I + RT_D + RT_{AO}}$ X 100 %
Allocation Efficiency	$WE_A = \frac{WT_B}{WT_{OO}}$ X 100 % = $\frac{WT_D + WT_{AO}}{WT_I + WT_D + WT_{AO}}$ X 100 %	$RE_A = \frac{RT_B}{RT_{OO}}$ X 100 % = $\frac{RT_D + RT_{AO}}{RT_I + RT_D + RT_{AO}}$ X 100 %
Technical Efficiency = Utilization Efficiency	$WE_T = WE_U = \frac{WT_{AO}}{WT_{AO} + WT_D}$ X 100 % = $\frac{WT_{AO}}{WT_B}$ X 100 %	$RE_T = RE_U = \frac{RT_{AO}}{RT_{AO} + RT_D}$ X 100 % = $\frac{RT_{AO}}{RT_B}$ X 100 %
Production Rate	WRP = $\frac{\text{number of finished products by the worker}}{\text{hour}}$	RRP = $\frac{\text{number of finished products by the cobot}}{\text{hour}}$

Where:

- RE_O is Robot Operation Efficiency
- WE_O is Worker Operation Efficiency
- RE_A is Robot Allocation Efficiency
- WE_A is Worker Allocation Efficiency
- RE_T is Robot Technical Efficiency
- WE_T is Worker Technical Efficiency

- RE_U is Robot Utility Efficiency
- WE_U is Worker Utility Efficiency
- RR_P is Robot Production Rate
- WR_P is Worker Production Rate
- RT_{AO} is Robot Actual Operation Time
- WT_{AO} is Worker Actual Operation Time
- RT_{OO} is Robot Overall Operation Time
- WT_{OO} is Worker Overall Operation Time
- RT_B is Robot Busy Time
- WT_B is Worker Busy Time
- RT_D is Robot Down Time
- WT_D is Worker Down Time
- RT_I is Robot Idle Time
- WT_I is Worker Idle Time

The production KPIs of a cooperative workcell are calculated by the SH, as discussed in section 5.2.4. Therefore, the SH sends the workcell KPIs to the ERP, whom calculates the whole enterprise KPIs. Because the enterprise scale is beyond the limits of this dissertation, it was sufficient to derive the previous formulas in Table 5.3, which can be used in further related research to monitor the cooperative enterprise performance. However, some of the derived KPIs such as the WR_P is very important within the cooperative workcell scope. For example, the WR_P is used by the OH to take a task assignment decision in the case study in section 6.3. In addition, the WR_P will be used by the PH to schedule the production orders in real production time in the case study in section 6.4.

Chapter 6 - Implementation and Case-Examples

"The Value of an Idea lies in the Using of it."

Thomas A. Edison

6.1 *Overview*

This chapter provides three case studies, which support the developed solution concept in chapter 4 and chapter 5. As the cooperative manufacturing is an open-end problem, this chapter focuses on the cooperative workcell implementation. Inasmuch as the cooperative workcell is the essential building unit of the cooperative manufacturing paradigm. The first case study proves that the proposed solution concept is feasible to be deployed by the available hardware and software in industrial automation field. Therefore, the case study ensures the viability of the proposed solution. The second case study shifts the focus to the knowledge representation in the cooperative workcell. Therefore, an ontology conceptual model of the cooperative workcell has been developed via this case study. The final case study shows the capability of the proposed solution to adapt and reschedule the cooperative workcell in the real production time. The three case studies are only examples, which validate the proposed solution concept. Nevertheless, the proposed solution is designed to be generic to fit the context of the cooperative manufacturing. Therefore, it can be applied over other case studies as well.

6.2 *Case Study 1: Solution Feasibility*

This case study is focused on building the cooperative workcell based on the concept that has been clarified in section 5.3. Thus, the case study constructs the cooperative workcell physical layer via combining the robot hardware plus the required sensors and GUI for the worker. The I/O events and data that are generated during a cooperative manufacturing scenario will be considered as well. These I/Os are transformed into information that is exchanged among the proposed HCA. The main goal of this case study is to test the feasibility of the concept over a real manufacturing hardware and automation devices. Therefore, the knowledge representation has been excluded during the implementation, as the complexity of the manufacturing information would distract the essential focus of the case study.

6.2.1 Case Study Description

6.2.1.1 Cooperative Workcell Physical Layer

The hardware components that have been used to construct the case study are shown in Figure 6.1. These components are as follows:

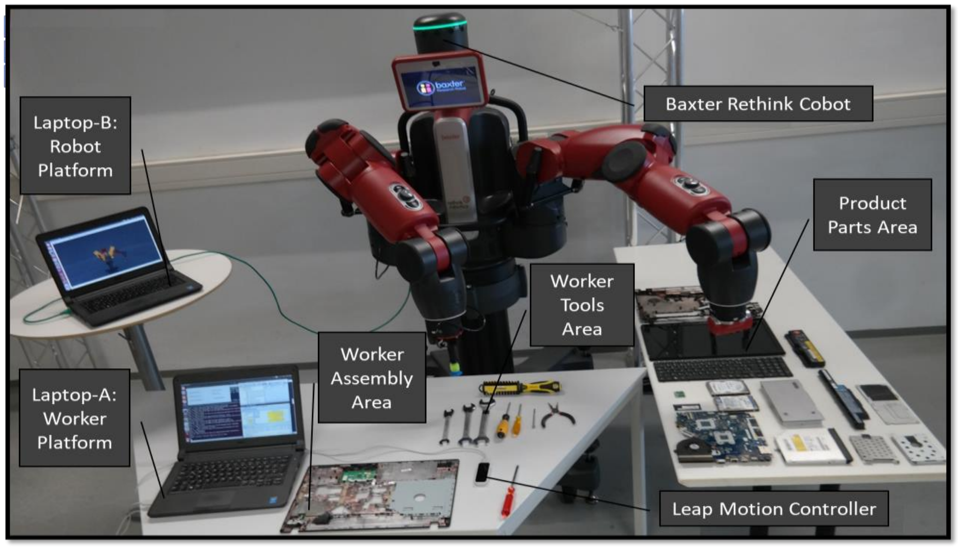


Figure 6.1: Cooperative workcell physical layer test environment.

1- Baxter dual-arm cobot from Rethink robotics.
2- Leap Motion recognition sensor.
3- Laptop-A to contain the worker platform.
4- Laptop-B to contain the robot platform.
5- Product parts (cooperative assembly of a customized laptop is assumed).
6- Wireless router to connect the worker platform to the robot platform wirelessly.

The test environment contains three areas within the reach of Baxter. These areas are dedicated for the product parts, the worker's tools, and the assembly operation. Different types of a laptop parts are available at the parts area. Therefore, different laptop configurations can be assembled.

6.2.1.2 Cooperative Workcell Holonic Control Layer

The case study targets a micro or small enterprise, where few cooperative workcells are required. Therefore, the coordination among the workcells is not a high necessity. Moreover, it is assumed that the worker has enough time to teach the robot the required positions of the pick and place operations before or during the assembly. Also, there is no MOM software to automatically generate the product recipe. Therefore, the worker needs to manually create and control the product recipe.

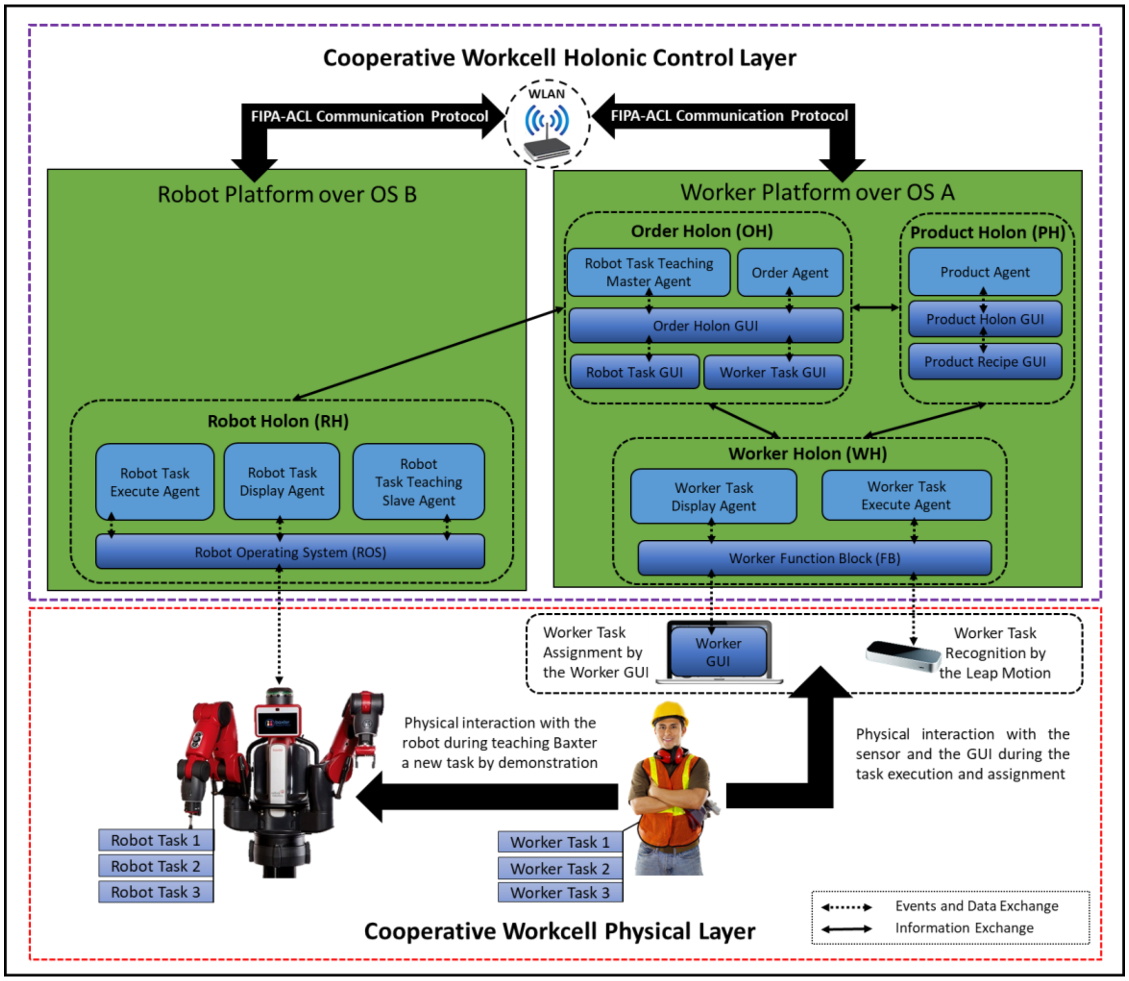
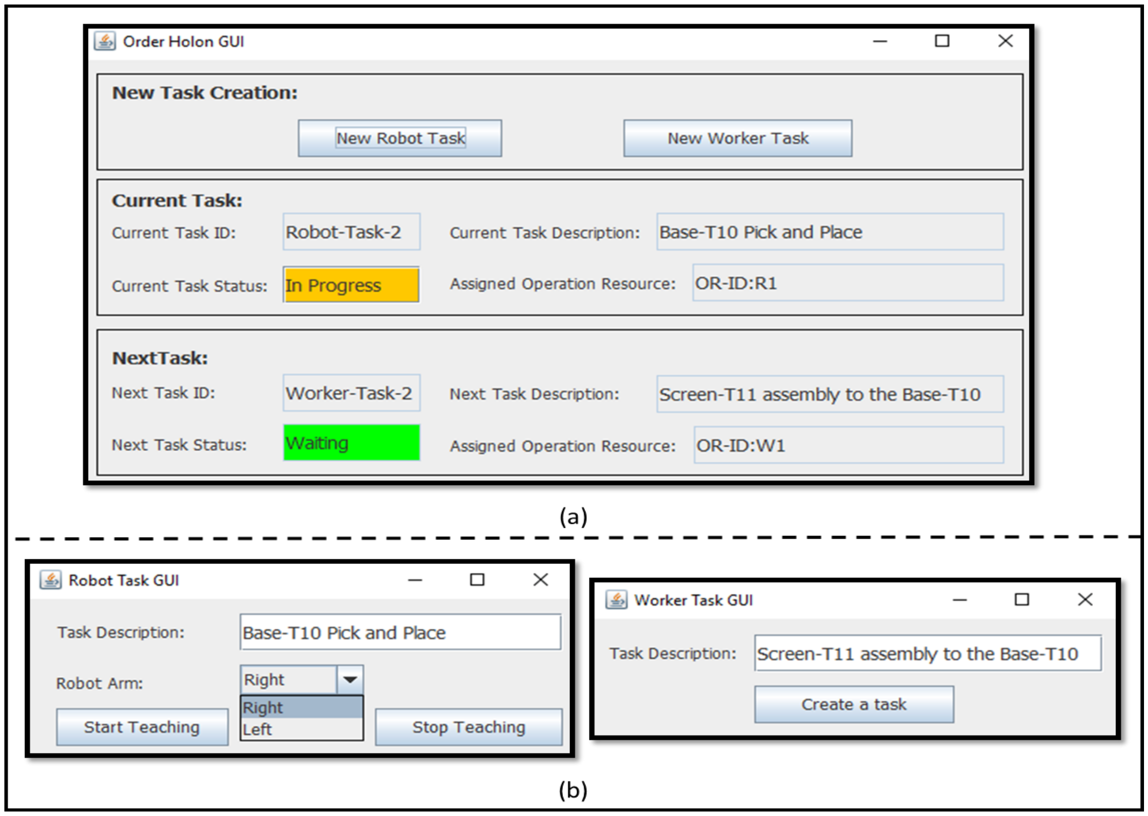

Cooperative Workcell Holonic Control Layer
FIPA-ACL Communication Protocol
WLAN
FIPA-ACL Communication Protocol
Robot Platform over OS B
Worker Platform over OS A
Order Holon (OH)
Robot Task Teaching Master Agent
Order Agent
Order Holon GUI
Robot Task GUI
Worker Task GUI
Product Holon (PH)
Product Agent
Product Holon GUI
Product Recipe GUI
Robot Holon (RH)
Robot Task Execute Agent
Robot Task Display Agent
Robot Task Teaching Slave Agent
Robot Operating System (ROS)
Worker Holon (WH)
Worker Task Display Agent
Worker Task Execute Agent
Worker Function Block (FB)
Worker Task Assignment by the Worker GUI
Worker GUI
Worker Task Recognition by the Leap Motion
Physical interaction with the robot during teaching Baxter a new task by demonstration
Physical interaction with the sensor and the GUI during the task execution and assignment
Robot Task 1
Robot Task 2
Robot Task 3
Worker Task 1
Worker Task 2
Worker Task 3
Events and Data Exchange
Information Exchange
Cooperative Workcell Physical Layer

Figure 6.2: Case study schematic.

The schematic in Figure 6.2 applies the proposed HCA over the test environment. Four holons are implemented over the cooperative workcell control layer. Three holons are deployed over laptop-A which contains the worker platform. These holons are a WH, OH, and PH. While, the RH is deployed over laptop-B which contains the robot platform. The worker platform and the robot platform communicate together via IEEE 802.11-wireless Ethernet protocol over the wireless router. The main reason to connect the two platforms wirelessly is to provide the mobility for the worker to move during many occasions such as teaching the robot a new task. The WH contains two agents, which are the worker-task-execute agent and the worker-task-display agent. These two agents are encapsulated inside the worker FB, as will be discussed in details in section 6.2.3.1. The worker-task-execute agent is responsible for controlling and monitoring the Leap Motion sensor. The worker-task-display agent is responsible for controlling and monitoring the worker GUI over laptop-A screen. The worker FB provides a physical interface to connect to the Leap Motion and the worker GUI to the MAS.

6.2.1.3 *Cooperative Workcell GUIs*


Order Holon GUI

New Task Creation:

New Robot Task | New Worker Task

Current Task:

Current Task ID: Robot-Task-2 — Current Task Description: Base-T10 Pick and Place

Current Task Status: In Progress — Assigned Operation Resource: OR-ID:R1

NextTask:

Next Task ID: Worker-Task-2 — Next Task Description: Screen-T11 assembly to the Base-T10

Next Task Status: Waiting — Assigned Operation Resource: OR-ID:W1

(a)

Robot Task GUI

Task Description: Base-T10 Pick and Place

Robot Arm: Right (Right, Left)

Start Teaching | Stop Teaching

Worker Task GUI

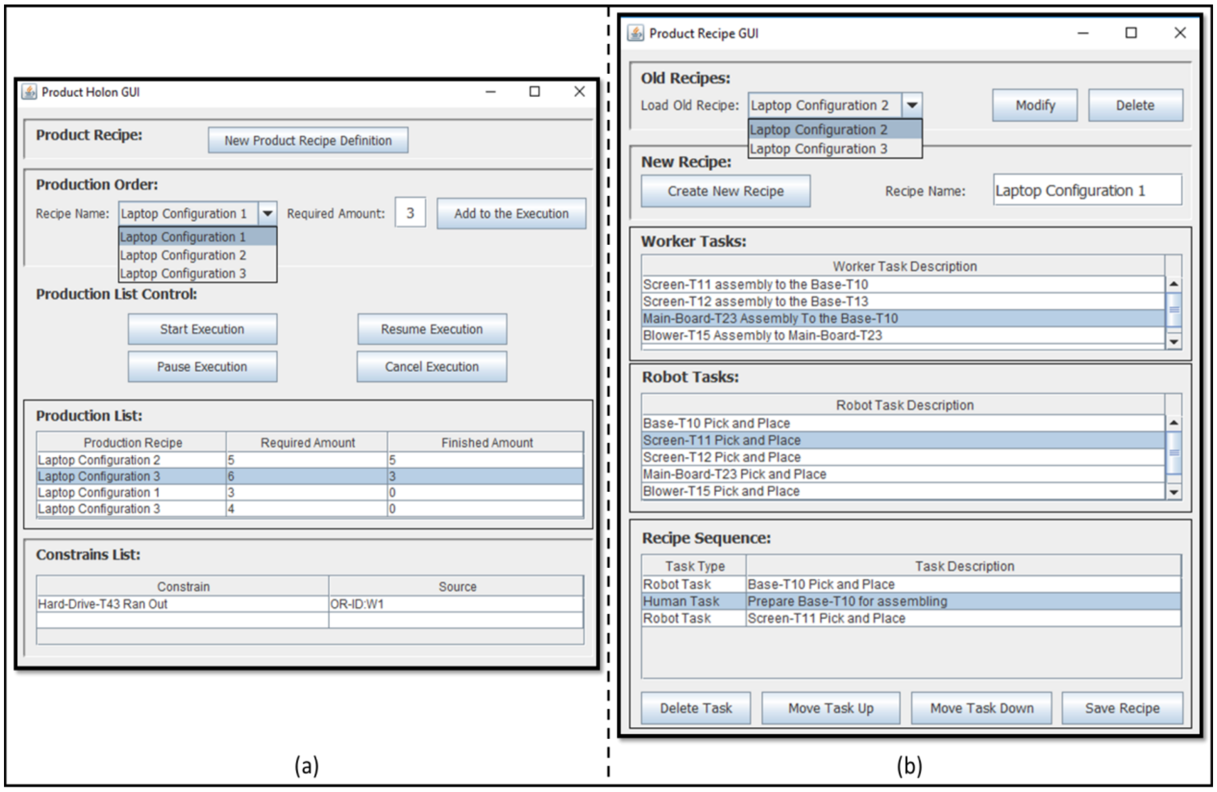
Task Description: Screen-T11 assembly to the Base-T10

Create a task

(b)

Figure 6.3: (a) OH GUI – (b) Robot and worker's task GUIs.

The OH contains two agents which are the order agent, and the robot-task-teaching-master agent. The order agent is responsible for the cooperative tasks assignment. While, the robot-task-teaching-master agent is only active when the worker is teaching a new task to Baxter. The order holon GUI is used to generate a new robot task or a new worker's task, as shown in Figure 6.3a and Figure 6.3b. In addition, the GUIs show the current and the next task information. Baxter teaching task is done by recording and storing the motion profile of a pick and place operation. Thus, this record can be reused during building an assembly recipe. The creation of a robot task will be explained in details in section 6.2.3. A worker's task is a simple description of an assembly step that is done by the worker, as shown in Figure 6.3b. A worker's task might have three statuses, which are waiting, in progress, or done. The Leap Motion sensor distinguishes between these three statuses. Detecting the worker pick and place gesture means that the worker's task changed from waiting to in progress. When the worker finishes the task, he explicitly interacts with the Leap Motion sensor by the swipe right gesture to indicate a task done status. The worker's task status recognition will be explained in details in section 6.2.2.2.



Recipe Name: Laptop Configuration 1 — Required Amount: 3

Laptop Configuration 1
Laptop Configuration 2
Laptop Configuration 3

Production List:

Production Recipe	Required Amount	Finished Amount
Laptop Configuration 2	5	5
Laptop Configuration 3	6	3
Laptop Configuration 1	3	0
Laptop Configuration 3	4	0

Constrains List:

Constrain	Source
Hard-Drive-T43 Ran Out	OR-ID:W1

Load Old Recipe: Laptop Configuration 2

Laptop Configuration 2
Laptop Configuration 3

Recipe Name: Laptop Configuration 1

Worker Tasks:

Worker Task Description
Screen-T11 assembly to the Base-T10
Screen-T12 assembly to the Base-T13
Main-Board-T23 Assembly To the Base-T10
Blower-T15 Assembly to Main-Board-T23

Robot Tasks:

Robot Task Description
Base-T10 Pick and Place
Screen-T11 Pick and Place
Screen-T12 Pick and Place
Main-Board-T23 Pick and Place
Blower-T15 Pick and Place

Recipe Sequence:

Task Type	Task Description
Robot Task	Base-T10 Pick and Place
Human Task	Prepare Base-T10 for assembling
Robot Task	Screen-T11 Pick and Place

Figure 6.4: (a) PH GUI – (b) Product recipe GUI.

The PH over the worker platform is composed of a product agent and a product holon GUI. The product holon GUI is used to create a new product recipe, as shown in Figure 6.4a. While, the product agent is responsible for creating and executing a production order list. The production orders execution is done via FCFS scheduling, as the optimal scheduling is not the focus of this case study. Additionally, the PH receives the production constrains instead of the SH, as the SH only exists in case of many cooperative workcells. A product recipe means a laptop assembly precedence plan. A laptop assembly plan is done by combining a sequence of the worker's tasks with the robot tasks, as shown in Figure 6.4b. The product holon GUI is connected to a database that contains all the production recipes. Therefore, it can be also used to modify or delete a previous recipe. The RH is located over laptop-B and contains three agents, which are the robot-task-execute agent, the robot-task-display agent, and the robot-task-teaching-slave agent. The RH interacts with the WH in order to teach Baxter a new task, and interacts with the OH in order to assign a task to Baxter. The physical component of the RH is Baxter ROS that is installed over Baxter hardware. Integrating Baxter ROS and JADE agents will be explained in details in section 6.2.3.2.

6.2.2 Physical Layer Implementation

6.2.2.1 Baxter Cobot

Baxter is a dual-arm cobot as every arm represents a seven DOF articulated industrial robot, as shown Figure 6.5a. During this case study, a pneumatic end effector has been attached to Baxter left arm, while a two-finger electrical gripper is attached to the right arm. The reachability of one of Baxter articulated arms is in the range of 1.0 m, which gives Baxter the ability to cover a large workspace, as shown in Figure 6.5b. The maximum payload of one of Baxter's arms is 2.3 kg. The two arms are programmable within the same ROS code that is located over Baxter's controller. It is therefore possible to synchronize the two arms movements and build a behaviour in which the arms are operated either independently from each other, or in coordination with each other. Moreover, Baxter applies an anti-self-collision mechanism to compensate its movements to avoid the two arms collision. Baxter display is one of the methods to interact with the worker during the operation. Baxter display can rotate 360^0, so it can follow the worker during his motion.

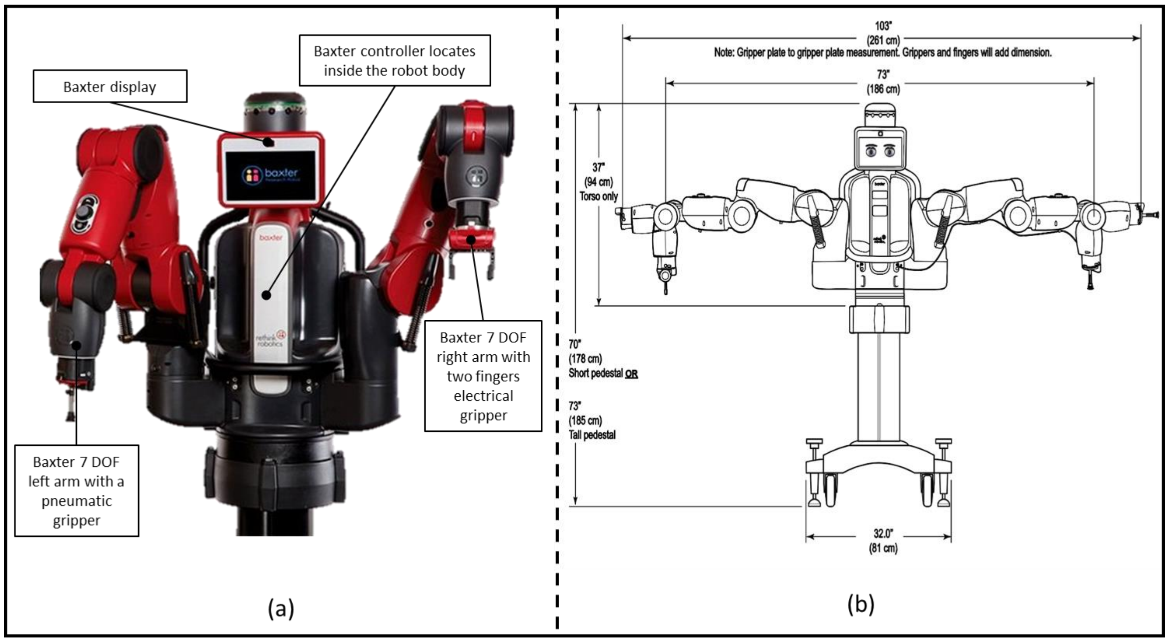

Baxter display
Baxter controller locates inside the robot body
Baxter 7 DOF right arm with two fingers electrical gripper
Baxter 7 DOF left arm with a pneumatic gripper
(a)
103" (261 cm)
Note: Gripper plate to gripper plate measurement. Grippers and fingers will add dimension.
73" (186 cm)
37" (94 cm) Torso only
70" (178 cm) Short pedestal OR
73" (185 cm) Tall pedestal
32.0" (81 cm)
(b)

Figure 6.5: (a) Baxter hardware – (b) Baxter workspace.

Baxter is slower than the conventional IR as its movements are elastic and non-hazard to the human worker. Therefore, the robot was designed to particularly target the SMEs. This is because the SMEs do no need to automate complicated processes. Thus, Baxter can be used as a multi-task shared resource by automating many simple low value-added tasks. Ultimately, the overall productivity of an SME will be enhanced.

6.2.2.2 *Gestures Recognition via Leap Motion*

Gesture recognition is a booming field that has many different applications. Gesture recognition is a common method to achieve the HMI. The HMI developed from using wired devices such as the mouse and the keyboard to the touch screens and nowadays is evolving towards the gestures interaction. The gesture recognition sensors can be categorized as vision-based or non-vision-based sensors, as shown in Table 6.1.

Table 6.1: Comparison between non-vision and vision-based gesture recognition techniques [NPW+14].

Non-vision-based Sensors This type of devices use various technologies to detect motions, such as accelerometers, multi-touch screens, EMG sensors and other, which include different types of detectors.	**Wearable**	This kind of device is in the form of garment, which includes sensors needed to recognize arrangement and motions of examined part of the body. It often occur in the form of gloves, armband or the whole outfit.
	Biomechanical	Type of device, which use biomechanical techniques such as electromyography, to measure parameters of gesture.
	Inertial	These devices measure the variation of the earth magnetic field in order to detect the motion. This kind of devices use accelerometers and gyroscopes for measurement.
	Haptics	Various kinds of touch screens.
	Electromagnetic	These devices measure the variation of an artificial electromagnetic field generated by wireless networks, electronic devices or produced by themselves.
Vision-based Sensors Vision-based sensors include one or several cameras and provide data stream from the captured video sequences. Processing of frame is based on filtering, analyzing and data interpreting.	**Video Cameras**	Gesture recognition techniques based on data derived from monocular camera using detection methods such as color or shape based techniques, learning detectors from pixel values or 3D model-based detection.
	Stereo Cameras	Techniques based on captured images from two cameras, which provide an approximation of the recorded data to a 3D model representation.
	Invasive Techniques	Systems which require using of body markers such as color gloves, LED lights. Play Station Move Controller is an example.
	Active Techniques	Requires the projection of some form of structured light.

Vision-based sensors that apply active techniques are lately getting more focus in research. Microsoft Kinect and Leap Motion are two successful examples of these sensors that are available in the commercial market. Microsoft Kinect is more suitable to detect the body gestures, while Leap Motion is designed to detect the hand gestures [Bon14]. Thus, Leap Motion is more suitable to interact with the HCA during the cooperative manufacturing. The Leap Motion is a sensor that aims to translate the hand movements into computer commands. The Leap dimensions are 8.0 cm in length and 3.0 cm in width, and it can be connected to the computer using a USB connection as shown in Figure 6.6a. The sensing range is a hemispherical volume that extends to 60.0 cm in radius as shown in Figure 6.6b. Through two monochromatic infrared cameras and three infrared LEDs the device observes its sensing volume. The infrared LEDs emit a 3D pattern of infrared light dots. Simultaneously, the cameras reconstruct the reflected data in a rate of 300 frames per second. The constructed data is transferred to the host computer via the USB connection, where they are parsed and processed by the Leap Motion software [Lea16].

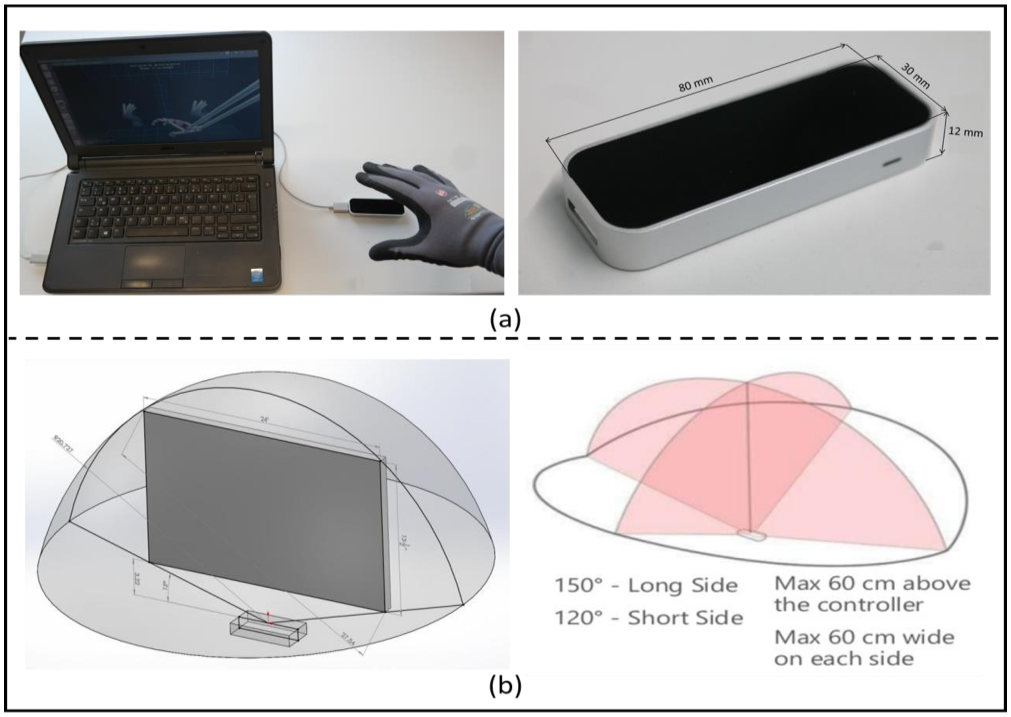


Figure 6.6: (a) Leap controller dimensions – (b) Leap controller workspace [Lea16].

The Leap Motion sensor has been used in a similar case study, but in the field of surgery in [PTL+15]. The goal of this research is to allow the surgeon to interface with a digital imaging software during the surgery without the need to take off the elastic gloves. In other words, finding a touchless replacement of the computer mouse.

Table 6.2: Comparison between Microsoft Kinect and Leap Motion characteristics [PTL+15].

Characteristics	Microsoft Kinect	Leap Motion
Manufacturer	• Microsoft	• Leap Motion Inc.
Dimensions	• 249 mm x 66 mm x 67 mm	• 80 mm x 30 mm x 12 mm
Technology	• 1 infrared transmitter • 1 RGB camera • 4 directional microphones	• 3 infrared transmitter • 2 infrared cameras (1.3 MP)
Image refresh rate	• 9 to 30 Hz	• 300 Hz
Recognition	• Body Movements • Facial recognition • Vocal recognition	• Hand movements • Fingers movements
Precision	• Centimeters	• Millimeters
Field of vision	• Horizontal 57^o • Vertical 43^o	• Anteroposterior 120^o • Left-right 150^o
Captor's range	• 1.2 : 3.5 m	• 0.002 : 0.61 m
Workspace's floor surface	• 6.0 m^2	• 1.16 m^2

Based on the comparison above, the following disadvantages of Kinect in the context cooperative manufacturing can be addressed:

- A considerable sensor dimensions and workspace surface, as the Kinect operates in the range of 6.0 m^2. Thus, using Kinect in cooperative manufacturing will practically waste the shop floor surface. As one worker needs a surrounding area of 6.0 m^2. Otherwise, an interference will occur. Furthermore, the surrounding area must be clear and clean, which is hard to achieve in a manufacturing environment.
- Entire upper body is not an appropriate method to interact with the HCA during the manufacturing. In addition, a repeatable upper body gesture will take long time and is exhausting.
- The worker must get out of the interaction zone to avoid any non-intended interaction that may happen during the manufacturing activities.
- The Kinect software configuration is extremely complex. Which could be very hard for a worker with low programming skills to deal with it.

The advantages of the Kinect in cooperative manufacturing are few in comparison with the Leap Motion for the following reasons:

- The Leap Motion has reduced sensor dimensions and covering volume (0.227 m^3), which fit the context of cooperative manufacturing.
- Hand gestures are more appropriate and less exhausting method of interaction during the manufacturing.
- Better technical specifications in comparison with the Kinect (e.g., Leap Motion has two infrared cameras of 1.3 MP against one infrared camera of 0.3 megapixels in case of the Kinect. The image refresh rate of the Leap Motion is 300 Hz in comparison with 30 Hz in case of the Kinect).
- Greater precision and reliability, the standard deviation in static measurements is between 0.0081 and 0.49 mm. and accuracy below 0.2 mm. This accuracy is much higher than the Kinect which can reach to 4.0 cm [WBR+13]
- Easier to configure, compatible with all OSs, and cheaper than the Kinect.

However, the disadvantage of the Leap Motion sensor is the need of the high processing power due to the high precision of the two infrared cameras.

Table 6.3: Workers' hand gestures meaning.

Worker Hand Gesture	Meaning
Pick	Worker Task Started
Place	Worker Task in Progress
Swipe right	Worker Task Done
Swipe Left	Worker is Unavailable
Lean Backward	Worker Task is Paused
Lean Forward	Work is Resumed
Tool	Worker needs Assistant

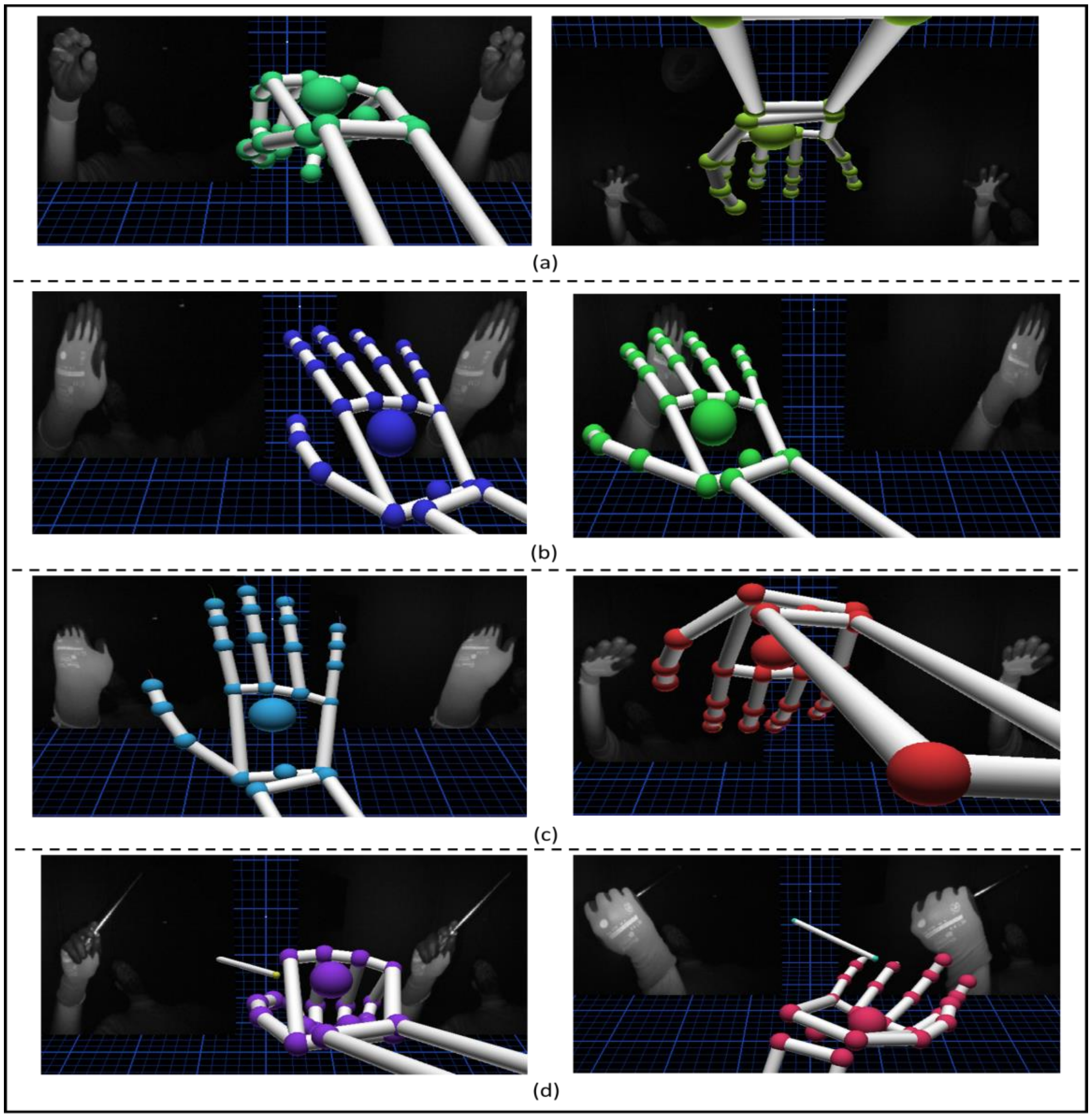
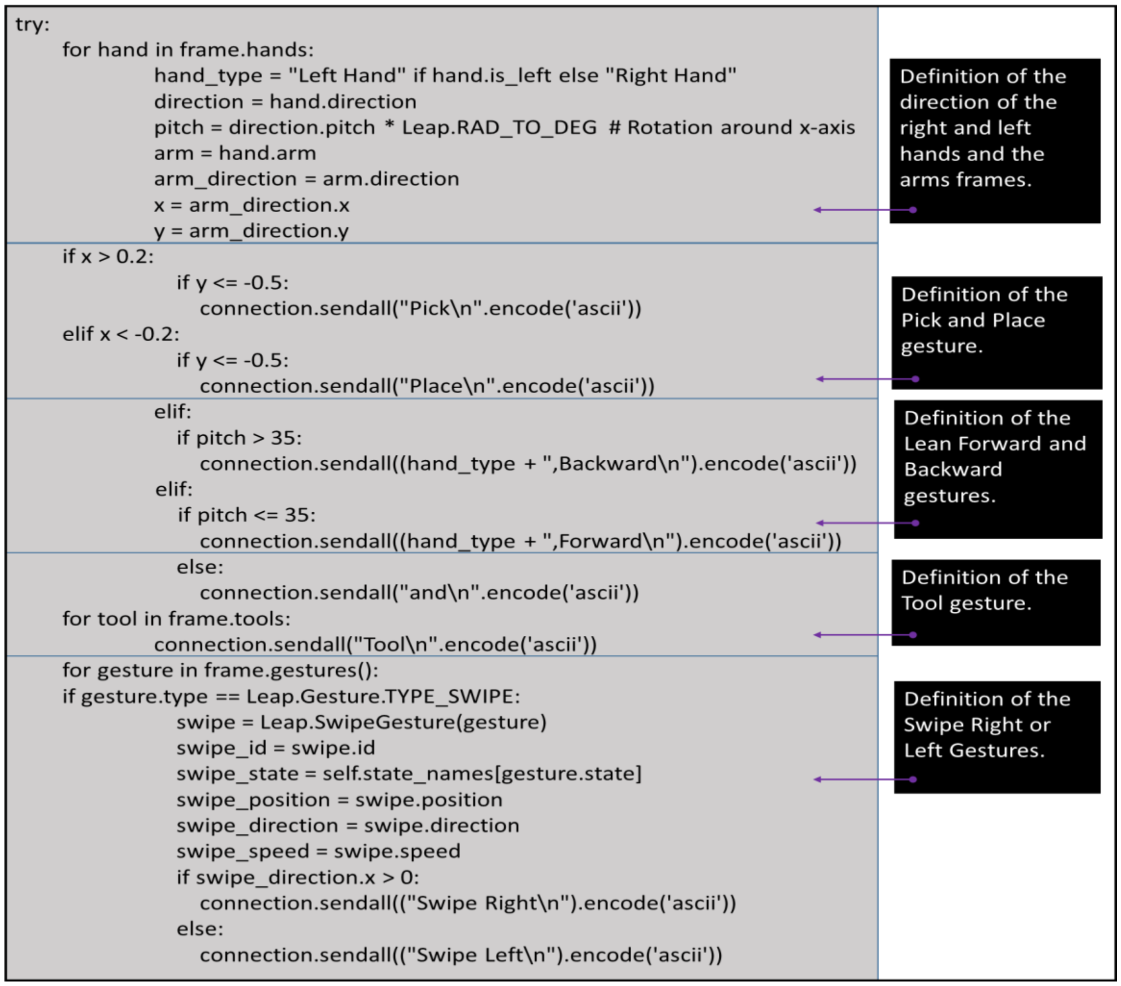


Figure 6.7: (a) Pick and place gestures – (b) Swipe right and left gestures – (c) Lean backward and forward gestures – (d) Tool gesture.

During this case study, the Leap Motion has been used as an input device to the WH. Therefore, the worker can interact with HCA via various hand gestures, without taking off the safety gloves. The seven gestures in Table 6.3 have been used to achieve this interaction. The pick gesture which is shown at the left side of Figure 6.7a, is used to inform the WH that the worker's task has started. While the place gesture, which is shown at the right side of Figure 6.7a, is used to inform the WH that the worker is busy and the task is in progress. The swipe right gesture, which is shown at the left side of Figure 6.7b, is used to inform the WH that the worker's task is done. While the swipe left gesture, which is shown at the right side of Figure 6.7b, is used to inform the WH that the worker is unavailable. The lean backward gesture, which is shown at the left side of Figure 6.7c, is used to inform the WH that the worker's task is paused. While the lean forward gesture, which is shown at the right side of Figure 6.7c, is used to inform the WH that the worker's task is resumed. The tool detection, which is shown in Figure 6.7d, is used when the worker needs to assign a new task to the Baxter while operation. For example, when a screwdriver tool is detected, Baxter would bring a specific amount of screws that are associated with this tool. Programming the Leap Motion means to write the code that defines the gestures. Therefore, the code continuously runs as a server that is waiting for one of these predefined gestures to be detected. The Leap Motion affords a variety of programming languages to write the gestures code. Python language has been used during this implementation to deploy the Leap server, as shown in Figure 6.8.



```
try:
    for hand in frame.hands:
            hand_type = "Left Hand" if hand.is_left else "Right Hand"
            direction = hand.direction
            pitch = direction.pitch * Leap.RAD_TO_DEG  # Rotation around x-axis
            arm = hand.arm
            arm_direction = arm.direction
            x = arm_direction.x
            y = arm_direction.y
    if x > 0.2:
              if y <= -0.5:
                connection.sendall("Pick\n".encode('ascii'))
    elif x < -0.2:
              if y <= -0.5:
                connection.sendall("Place\n".encode('ascii'))
            elif:
              if pitch > 35:
                connection.sendall((hand_type + ",Backward\n").encode('ascii'))
            elif:
              if pitch <= 35:
                connection.sendall((hand_type + ",Forward\n").encode('ascii'))
              else:
                connection.sendall("and\n".encode('ascii'))
    for tool in frame.tools:
            connection.sendall("Tool\n".encode('ascii'))
    for gesture in frame.gestures():
    if gesture.type == Leap.Gesture.TYPE_SWIPE:
              swipe = Leap.SwipeGesture(gesture)
              swipe_id = swipe.id
              swipe_state = self.state_names[gesture.state]
              swipe_position = swipe.position
              swipe_direction = swipe.direction
              swipe_speed = swipe.speed
              if swipe_direction.x > 0:
                connection.sendall(("Swipe Right\n").encode('ascii'))
              else:
                connection.sendall(("Swipe Left\n").encode('ascii'))
```

Definition of the direction of the right and left hands and the arms frames.
Definition of the Pick and Place gesture.
Definition of the Lean Forward and Backward gestures.
Definition of the Tool gesture.
Definition of the Swipe Right or Left Gestures.

Figure 6.8: Python code to define the workers' hand gestures.

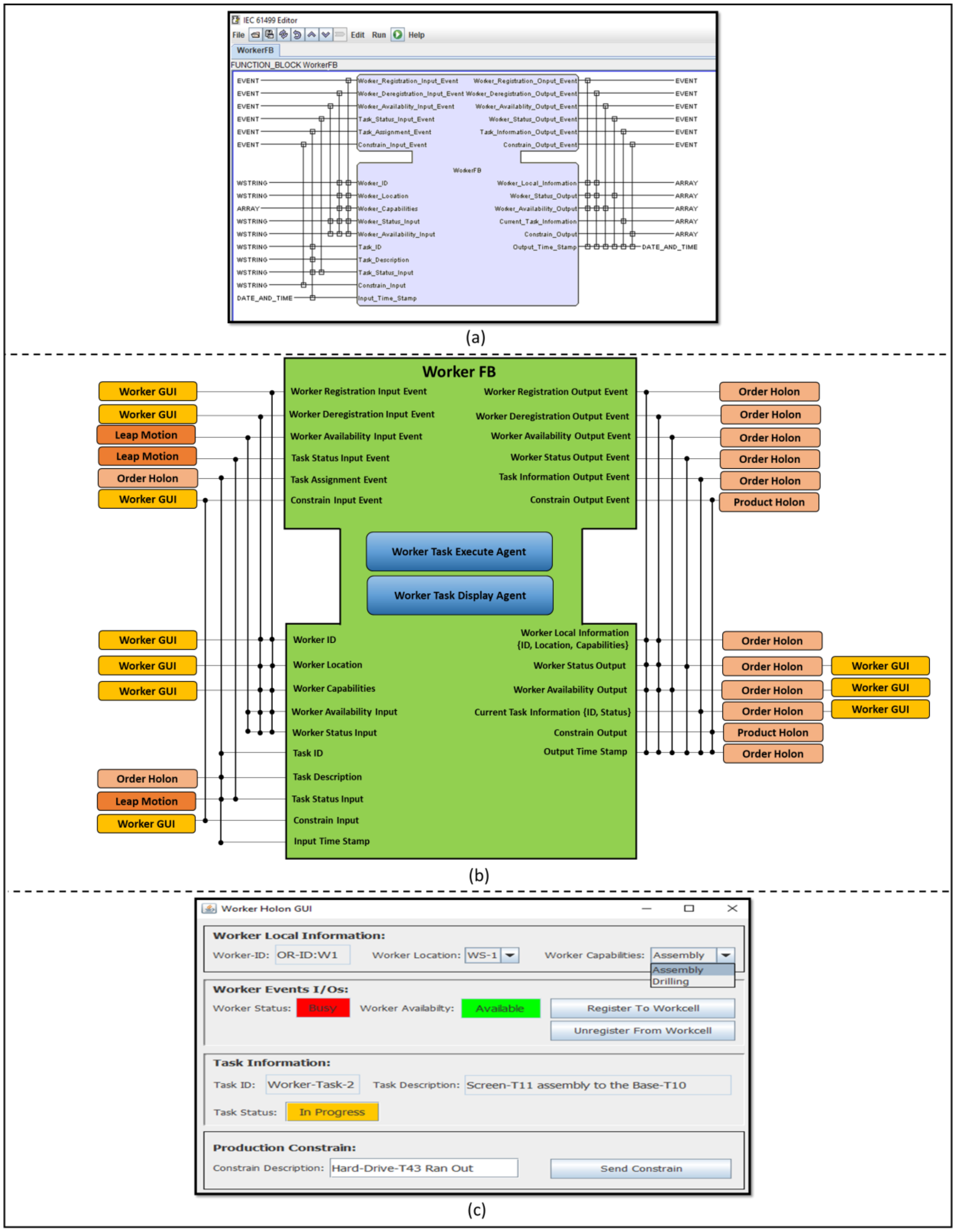
6.2.3 Control Layer Implementation

6.2.3.1 Worker FB

The worker FB represents the physical interface that connects the hardware to the MAS. IEC 61499 FB is a very feasible method to implement the physical component of the WH. This is because IEC 61499 standard supports the distributed industrial controllers such as the PAC, as mentioned earlier in section 4.1.1.1. IEC 61499 FB distinguishes between three terms, which are events, data, and algorithms. The environment that has been used to implement the worker FB is Holobloc FBDK, which is shown in Figure 6.9a. FBDK uses Java to code the FB algorithm, which in this case denotes the agent behaviours and interaction. As JADE is used to implement the agents, this means that also the agents are developed as a Java code. Accordingly, no problem to integrate the worker agents and the worker FB, as shown in Figure 6.9b. Therefore, many software integration problems are avoided by using FBDK along with JADE to develop the worker holon.

The events and the data of the worker FB can be directly developed via the FBDK graphical editor. FBDK maps between the input events and the input data, as well as the output events and the output data. When an input event triggers an algorithm, it passes the associated input data to that algorithm. When an algorithm finishes the data processing, it triggers one or more output events and passes the processed data to the output. The source of the input events and data values comes from the input sensors, the GUIs, or the output events and data from the other holons. The destination of the output events and data is either the output controllers, the GUIs, or the input events and data of the other holons. In case of the worker FB, the input sources are the Leap Motion, the worker GUI, and the OH. While the output sources of the worker are the OH, PH, and the worker GUI. The worker GUI has been developed via Java, as an input and output interface to the worker, as shown in Figure 6.9c.

The first two input events at the worker FB are the worker registration and deregistration, which are mapped with the worker local information such as the worker-ID, location, and capabilities. The worker registration/deregistration events are generated from the worker GUI in this case study, however they could be also generated from the Leap Motion by defining two extra gestures. The process of registration and deregistration is discussed in section 5.2.1. The worker registration/deregistration events are also setting the worker's availability and status to their appropriate values. The worker's availability and the task status input events are connected to the Leap Motion sensor. The values of the worker's availability and the task status input data change based on the detected hand gesture from the worker as discussed in the previous section.



Worker Local Information:

Worker-ID: OR-ID:W1 Worker Location: WS-1 Worker Capabilities: Assembly (Assembly, Drilling)

Worker Events I/Os:

Worker Status: Busy Worker Availabilty: Available

Task Information:

Task ID: Worker-Task-2 Task Description: Screen-T11 assembly to the Base-T10

Task Status: In Progress

Production Constrain:

Constrain Description: Hard-Drive-T43 Ran Out

Figure 6.9: (a) Worker FB design over FBDK – (b) Worker FB schematic – (c) Worker FB interface.

The task assignment event is generated by the OH and mapped to the current task information, which has been discussed in details in section 5.2.1. A simple task description via a string data type is enough to achieve the goal of this case study. Therefore, the current task information appears to the worker via his GUI. Every new task assignment must be marked by a time stamp, which is useful to calculate the overall time that has been taken to finish this task. The last input event is the constrain event which is generated by the worker via his GUI when it is needed. A constrain event could be also generated from a sensor which is dedicated to detecting a specific constrain, however this depends on the controlled process requirements and priorities. The constrain description is created by the worker from his GUI in a string format and sent to the PH to be solved. The output events and data are generated by the two agents that are encapsulated within the worker FB. All the outputs from the worker FB go to the OH for further processing, except one output that is the production constrains which goes to the PH. In a large-scale enterprise, the production constrains are the responsibility of the SH as discussed before in section 5.2.4. Nevertheless, as the SH does not exist in this implementation, the production constrains are delegated to the PH. Additionally, some outputs are showed over the worker GUI such as the status, the availability, and the task status. The time stamp output is very useful to mark all the outputs from the worker FB. Specially, in case of task information output that indicates a task done status, the output time stamp is subtracted from the input time stamp to calculate the task execution time.

6.2.3.2 ROS and JADE Integration

ROS represents Baxter physical interface that connects Baxter hardware to the MAS, as explained earlier in section 4.1.1.2. Integrating the worker FB and JADE did not present a technical problem as FBDK and JADE could be encapsulated together in one JAVA code. However, in case of integrating ROS that is coded via Python and JADE that is coded via Java, the software integration problem appears. ROS provides a **S**oftware **D**evelopment **K**it (**SDK**), which has a software interface that simplifies the process of developing applications for Baxter. Baxter interface provides information about the robot current state (e.g. whether the robot is enabled or disabled), the output of different sensors, the current state of the actuators of the robot (e.g., the current angles of the robot arms), and provides the privilege to control the hardware of the robot. In a similar manner of JADE, ROS divides a complex problem into smaller problem solvers that are called nodes. Thus, every ROS node solves a specific problem. Three ROS nodes have been implemented during this case study. Those nodes are task-recording node, task-execute node, and task-display node, which can be seen at the bottom of Figure 6.10.

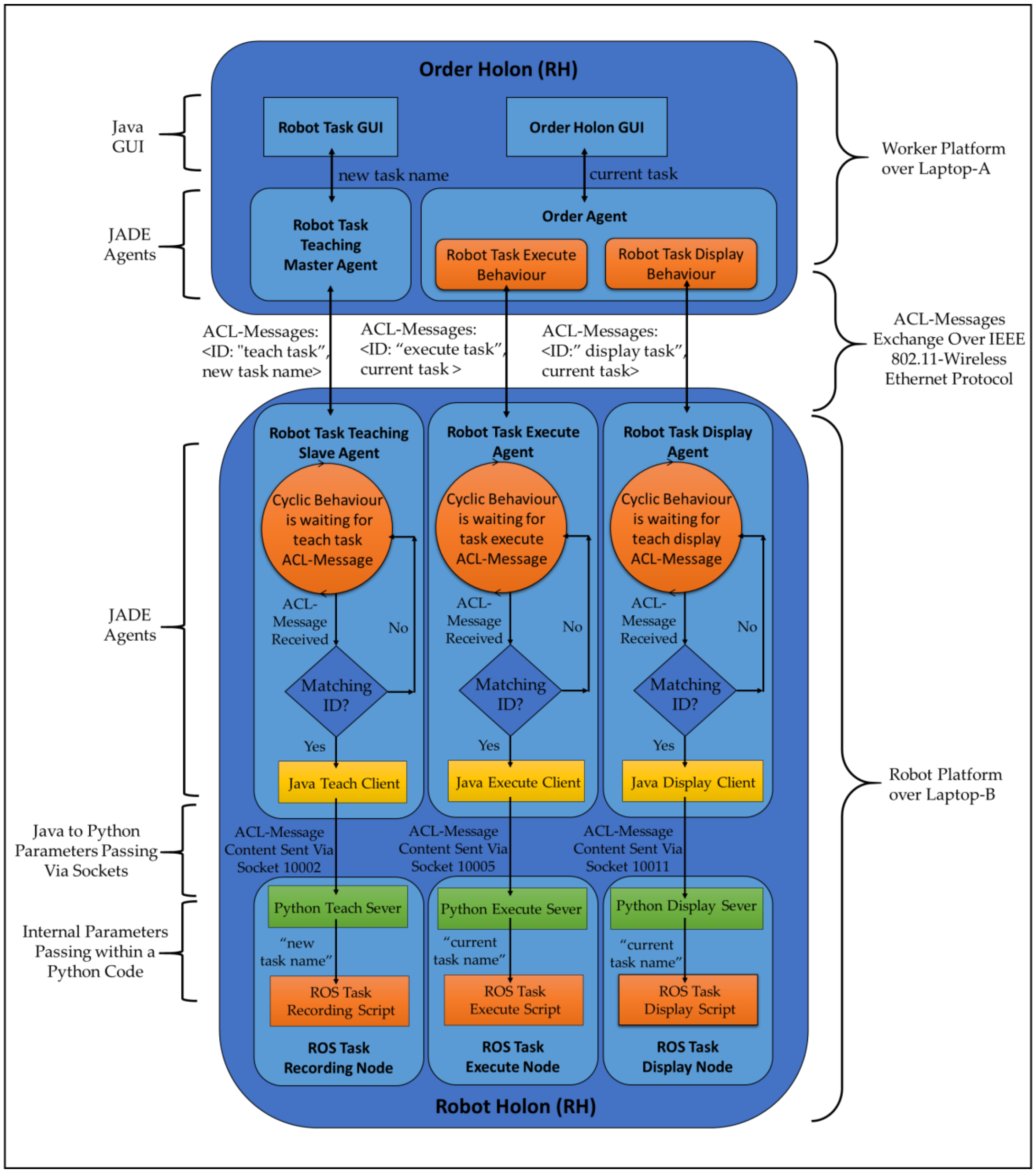

Order Holon (RH)
Robot Task GUI
Order Holon GUI
new task name
current task
Robot Task Teaching Master Agent
Order Agent
Robot Task Execute Behaviour
Robot Task Display Behaviour
Java GUI
JADE Agents
Worker Platform over Laptop-A
ACL-Messages: <ID: "teach task", new task name>
ACL-Messages: <ID: "execute task", current task >
ACL-Messages: <ID:" display task", current task>
ACL-Messages Exchange Over IEEE 802.11-Wireless Ethernet Protocol
Robot Task Teaching Slave Agent
Robot Task Execute Agent
Robot Task Display Agent
Cyclic Behaviour is waiting for teach task ACL-Message
Cyclic Behaviour is waiting for task execute ACL-Message
Cyclic Behaviour is waiting for teach display ACL-Message
ACL-Message Received
No
Matching ID?
Yes
Java Teach Client
Java Execute Client
Java Display Client
JADE Agents
Robot Platform over Laptop-B
ACL-Message Content Sent Via Socket 10002
ACL-Message Content Sent Via Socket 10005
ACL-Message Content Sent Via Socket 10011
Java to Python Parameters Passing Via Sockets
Python Teach Sever
Python Execute Sever
Python Display Sever
"new task name"
"current task name"
ROS Task Recording Script
ROS Task Execute Script
ROS Task Display Script
Internal Parameters Passing within a Python Code
ROS Task Recording Node
ROS Task Execute Node
ROS Task Display Node
Robot Holon (RH)

Figure 6.10: JADE and ROS integration.

Over the RH, ROS task-recording node is responsible for recording Baxter motion profile while it is in teaching mode. In other words, recoding the robot task. ROS task-execute node commands one of Baxter arms to repeat a previously recorded task, when the OH assigns this task to Baxter. ROS task-display node commands Baxter display to show the current assigned task. Each of these ROS nodes is linked with an associated JADE agent, which are robot-task-teaching-slave agent, robot-task-execute agent, and robot-task-display agent respectively. Each of the previously mentioned agents implements a cyclic behaviour to receive the ACL-Messages from the OH (i.e., Robot-task-teaching-master agent or order agent). An ACL-Message represents a command to Baxter. The type of this command is filtered based on the conversation-ID of the ACL-Message (as explained in section 4.1.2). The content of an ACL-Message contains the necessary parameters to perform the command such as the new task name or the current task name. A new task name is used to record a new robot task while a current task name is used to repeat or display this task.

Socket programming has been used to solve the problem of parameters exchanging between ROS nodes and JADE agents (i.e., ROS and JADE integration). Socket programming provides a communication protocol between different platforms. Those platforms can locate on the same device or on two separated devices. One of the platforms is considered as a server while the other is the client. The server is always running and ready to establish a connection with the client. When the client connects to the server, the server creates a socket between itself and the client. This socket is considered as the communication channel between them, where they can exchange data with each other. The solution that is shown in Figure 6.10 deploys Python servers and Java clients that can be summarized as follows:

- Python servers: those servers are continuously waiting for the clients to connect and exchange data. They are the intermediate connection points between JADE and ROS nodes.
- Java clients: those clients connect to the running servers when it is needed to exchange data. After the data transmission is done and the process is complete, the client disconnects itself from the server in order to have the chance to receive another request.

Each agent over the RH creates a socket to transfer the received ACL-Message content from the OH to its corresponding Python server as described before. Then, those servers pass the ACL-Message content to ROS nodes. Therefore, the received messages from the OH are translated into a series of actions that can be understood and executed by Baxter. The description of the mechanisms that are followed by those three agents are summarized as follows:

- Robot-task-teaching-slave agent: deploys a cyclic behaviour that waits an ACL-Message from the robot-task-teaching-master agent to start recording a new task. The ACL-Message contains the name of a new task. Thus, the agent initializes a unique socket “10002” to communicate with the corresponding Python server, which receives the name of the new task and then uses ROS task-record node to record Baxter arm motion profile.

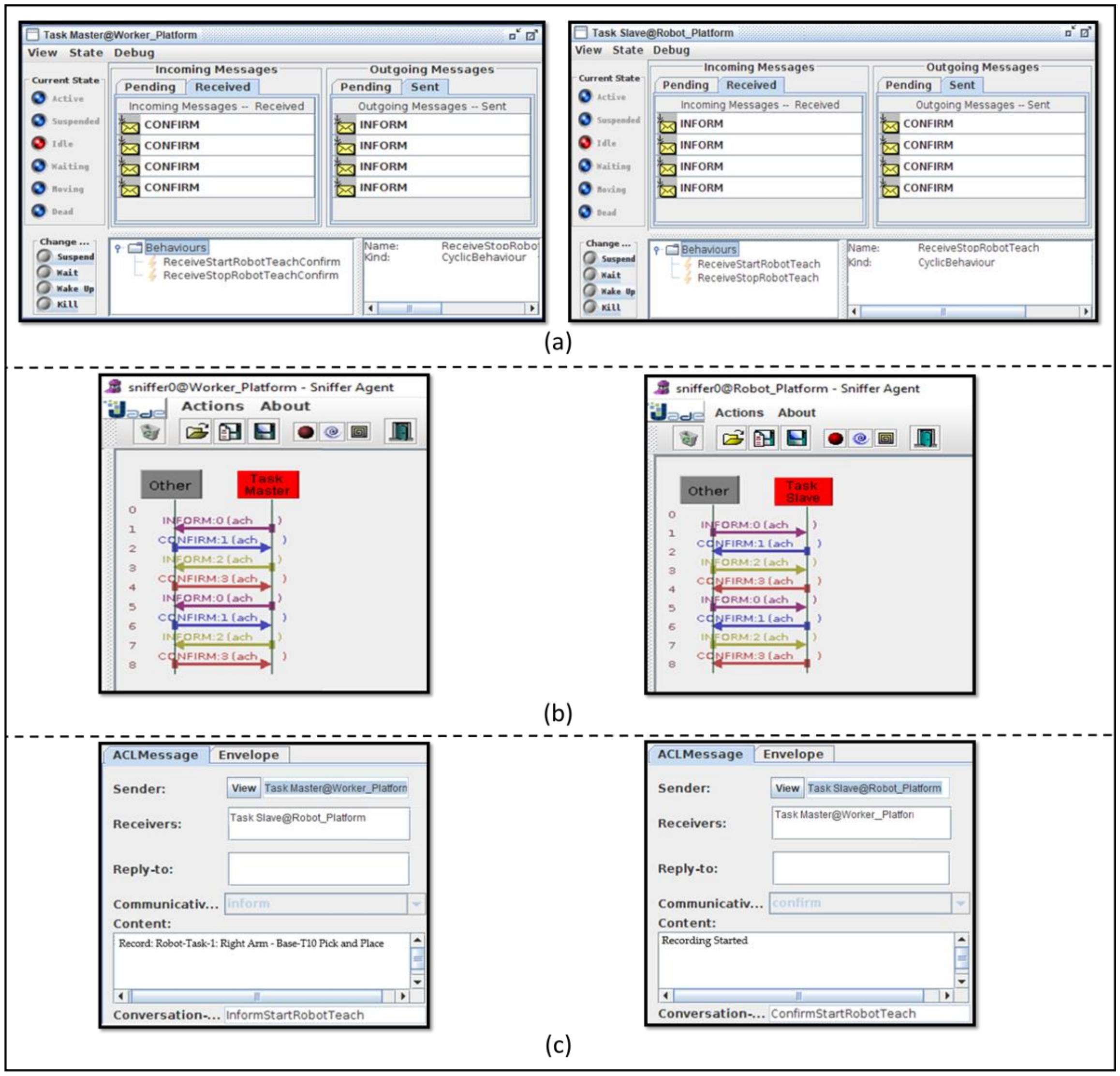
- Robot-task-execute agent: deploys a cyclic behaviour that waits an ACL-Message from the order agent to execute a previously recorded robot task. The ACL-Message contains the name of the task, which should be executed. Thus, the agent initializes a unique socket “10005” to communicate with the corresponding Python server, which receives the name of the task, and then uses ROS task-execute node to execute this task.
- Robot-task-display agent: deploys a cyclic behaviour that waits an ACL-Message from the order agent to display the name of the current robot task. The ACL-Message contains the name of the current task, which should be displayed. Thus, the agent initializes a unique socket “10011” to communicate with the corresponding Python server, which receives the name of the task, and then uses ROS task-display node to display this task.

In this case study, the integration between JADE agents and ROS nodes has been achieved via Java to Python parameters passing via socket programming. Particularly, the same method can be applied to integrate JADE agents with another non-Java software. Generally, the same concept can be applied to integrate any two different programming languages.

6.2.3 Holons Interaction

An ACL-Message exchange is the very first step to achieve JADE agent interaction with another. Therefore, this section is focused on the mechanism of ACL-Messages exchange. This mechanism will be used to implement all the next case studies within this chapter. Thus, it will be discussed in more details during this case study.

JADE is not only a tool to code and deploy an autonomous agent, but also it provides a graphical interface which is used to debug the MAS during its development. The first tool that is used to debug the interaction is JADE Introspector that is shown in Figure 6.11a. In this sub-figure, JADE Introspector shows four successful sending/receiving processes. A sending/receiving process has been explained details in section 4.2.3. An ACL-Message sending means that one JADE agent is constructing an ACL-Message, then it uses one of its behaviours to send this message. An ACL-Message receiving means that another JADE agent receives this ACL-Message and then directs it to the right behaviour for further processing. JADE agent directs the ACL-Messages by filtering them based on some parameters such as the communicative-act or the conversation-ID that have been explained in details in section 4.1.2.



Task Master@Worker_Platform

Incoming Messages -- Received	Outgoing Messages -- Sent
CONFIRM	INFORM
CONFIRM	INFORM
CONFIRM	INFORM
CONFIRM	INFORM

Behaviours
ReceiveStartRobotTeachConfirm
ReceiveStopRobotTeachConfirm

Name: ReceiveStopRobo
Kind: CyclicBehaviour

Task Slave@Robot_Platform

Incoming Messages -- Received	Outgoing Messages -- Sent
INFORM	CONFIRM
INFORM	CONFIRM
INFORM	CONFIRM
INFORM	CONFIRM

Behaviours
ReceiveStartRobotTeach
ReceiveStopRobotTeach

Name: ReceiveStopRobotTeach
Kind: CyclicBehaviour

sniffer0@Worker_Platform - Sniffer Agent

```
0
1  INFORM:0 (ach )
2  CONFIRM:1 (ach )
3  INFORM:2 (ach )
4  CONFIRM:3 (ach )
5  INFORM:0 (ach )
6  CONFIRM:1 (ach )
7  INFORM:2 (ach )
8  CONFIRM:3 (ach )
```

sniffer0@Robot_Platform - Sniffer Agent

```
0
1  INFORM:0 (ach )
2  CONFIRM:1 (ach )
3  INFORM:2 (ach )
4  CONFIRM:3 (ach )
5  INFORM:0 (ach )
6  CONFIRM:1 (ach )
7  INFORM:2 (ach )
8  CONFIRM:3 (ach )
```

Sender:	Task Master@Worker_Platform
Receivers:	Task Slave@Robot_Platform
Reply-to:	
Communicativ...	inform
Content:	Record: Robot-Task-1: Right Arm - Base-T10 Pick and Place
Conversation-...	InformStartRobotTeach

Sender:	Task Slave@Robot_Platform
Receivers:	Task Master@Worker_Platfon
Reply-to:	
Communicativ...	confirm
Content:	Recording Started
Conversation-...	ConfirmStartRobotTeach

Figure 6.11: Holons interaction while teaching Baxter a new task (a) Via JADE introspector tool – (b) Via JADE sniffer tool – (c) Via FIPA ACL-Message.

Figure 6.11a shows an interaction between the robot-task-teaching-master agent over the worker platform (i.e., Task-Master@Worker_Platform) and the robot-task-teaching-slave agent over the robot platform (i.e., Task-Slave@Robot_Platform). The purpose of this interaction is to teach Baxter a new task. Four Inform-Messages have been sent from the task-master agent to the task-slave agent. Each time the task-slave agent receives an Inform-Message, it answers the message back with a Confirm-Message. The Inform/Confirm process is similar to the handshaking pattern that is often used in reliable communication protocols. However, it depends on the MAS developer to implement this mechanism. During the entire case studies implementation, this mechanism has been considered, to ensure the communication reliability and to debug and diagnose the MAS.

The interaction occurs over a wireless network. This is more obvious to be seen via JADE Sniffer tool that is shown Figure 6.11b. At the left half of Figure 6.11b, the Task-Master@Worker_Platform sends an Inform-Message to the Task-Slave@Robot_Platform. This Inform-Message is received, as shown at the right half of Figure 6.11b. Thus, the Task-Slave@Robot_Platform replies with a Confirm-Message that is received by the Task-Master@Worker_Platform. After that, the same scenario is repeated for three times.

At the left half of Figure 6.11c, the Inform-Message that is sent by the task master agent is shown. This ACL-Message is generated to initialize a Baxter new recording task. The name of Baxter new task is a part of the message content. The content of the Inform-Message is parsed by the task-slave agent. Therefore, it commands the ROS over Baxter to be ready to record a new motion profile using the right arm. The record starts by pressing a start-teaching button at the robot task GUI which has been shown previously in Figure 6.3a. Finally, when the recoding ends by pressing stop-teaching button at the robot task GUI, the task-slave agent sends a Confirm-Message to indicate the end of the recording process. This Confirm-Message can be seen at the right half of Figure 6.11c.

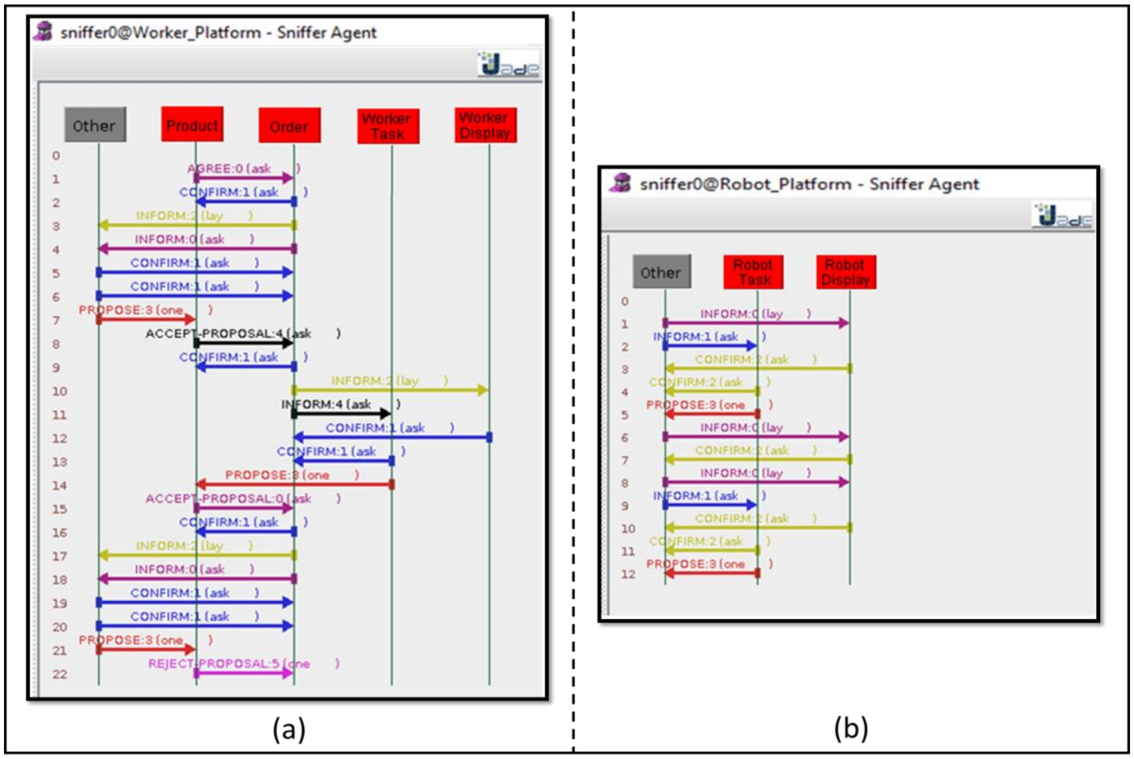


Figure 6.12: JADE interaction while executing a cooperative assembly scenario composed of three tasks (a) Over worker platform – (b) Over robot platform.

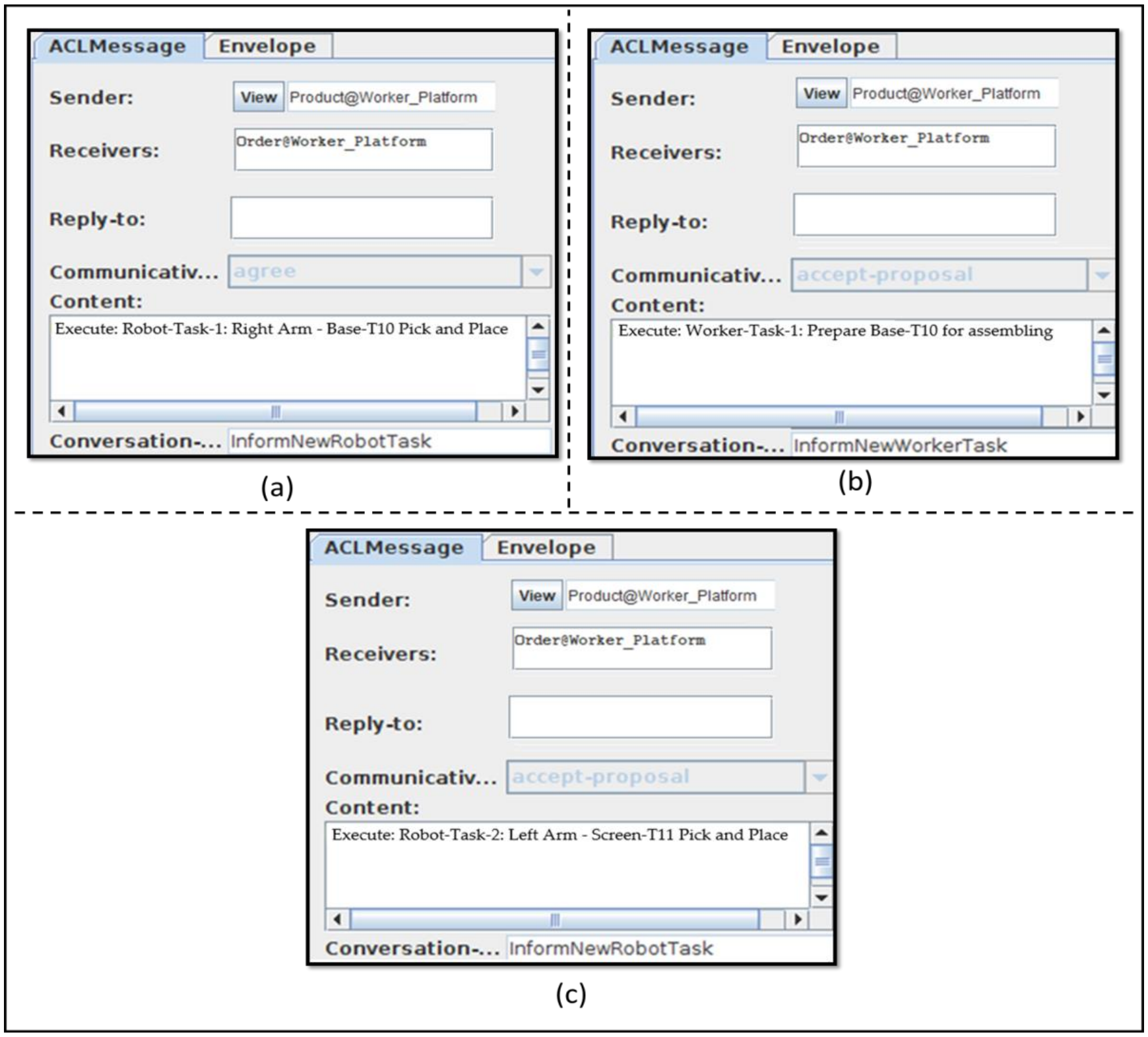
In order to see the interaction among all JADE agents within the case study solution, a cooperative assembly scenario that composes of three steps is assumed. After teaching

Baxter the necessary motion profiles to perform two pick and place operations, a production recipe can be defined by adding a worker's task between Baxter's two tasks. Simply, Baxter handles the base of a laptop product to the worker. The worker prepares the laptop base to be assembled, then swipes right to indicate that his task has been finished. Then, Baxter automatically handles the next part, which is the screen. This scenario is defined via the product recipe GUI that has be shown earlier in Figure 6.4b. By pressing the start-execution button at the product GUI which has been shown in Figure 6.4a, the interaction between the agents starts to achieve this recipe.

By starting the agent interaction that is shown in Figure 6.12, an Agree-Message is sent from the product agent to the order to indicate the start of a new product order execution. The message content holds the task details as shown in Figure 6.13a. The content of the Agree-Message indicates that this task belongs to the Baxter right arm. Therefore, the order agent assigns this task to Baxter right arm (i.e., robot-task agent) and displays that task on Baxter display (i.e., robot-display agent). This can be seen via lines 3, 4, 5, and 6 in Figure 6.12a. The same messages can be seen in lines 1, 2, 3, and 4 in Figure 6.12b. When the robot task agent finishes its task, it sends a Propose-Message to the product agent to confirm that this task has finished. This can be seen in line 4 of Figure 6.12b and line 7 of Figure 6.12a.

When the product agent receives the Propose-Message, it checks if there are still tasks left at the recipe. Thus, it finds that the next task is the worker's task. Therefore, it sends an Accept-Message that holds the worker's task details as shown in Figure 6.13b. Therefore, the order agent assigns this task to worker (i.e., worker-task agent) and displays that task on worker GUI (i.e., worker-display agent). This can be seen via lines 10, 11, 12, and 13 in Figure 6.12a. When the worker finishes his task, he performs the swipe right gesture to indicate that the task has been accomplished. Therefore, the worker's task sends a Propose-Message the product agent to confirm that this task has finished. This can be seen in line 14 of Figure 6.12a.

When the product agent receives the Propose-Message, it checks if there are still tasks left at the recipe. Thus, it finds that the last task is a robot task. Thus, it sends an Accept-Proposal-Message that holds the robot task details as shown in Figure 6.13c. Therefore, the same scenario to assign Baxter task will be carried on. Except this time, the task will be assigned for the left arm. After Baxter finishes its task, the robot-task agent will send a Propose-Message to the product agent. However, the product agent finds that the product recipe is finished. Therefore, it sends a Reject-Proposal-Message to the order agent to inform that this product has finished. The Reject-Proposal-Message can be seen in line 22 of Figure 6.12a.



(a)

ACLMessage | Envelope

Sender: View Product@Worker_Platform

Receivers: Order@Worker_Platform

Reply-to:

Communicativ... agree

Content:

Execute: Robot-Task-1: Right Arm - Base-T10 Pick and Place

Conversation-... InformNewRobotTask

(b)

ACLMessage | Envelope

Sender: View Product@Worker_Platform

Receivers: Order@Worker_Platform

Reply-to:

Communicativ... accept-proposal

Content:

Execute: Worker-Task-1: Prepare Base-T10 for assembling

Conversation-... InformNewWorkerTask

(c)

ACLMessage | Envelope

Sender: View Product@Worker_Platform

Receivers: Order@Worker_Platform

Reply-to:

Communicativ... accept-proposal

Content:

Execute: Robot-Task-2: Left Arm - Screen-T11 Pick and Place

Conversation-... InformNewRobotTask

Figure 6.13: ACL-Messages to start tasks execution (a) Robot task-1 – (b) Worker's task-1 – (c) Robot task-2.

6.3 *Case Study 2: Cooperative Workcell Knowledge Model*

This case study is focused on the knowledge in cooperative manufacturing. More specifically, the knowledge representation by the holon to describe itself and the manufacturing domain. Then, the knowledge exchange among the different holons of the proposed HCA. Finally, the knowledge reasoning to derive new facts and obtain decisions during the manufacturing. FIPA-SL language has been used during the implementation to build the knowledge schema (refer to section 4.2.2). Hence JADE agents use instances of these schemes to communicate and exchange the ontology-based messages (refer to section 4.2.3). Thus, the knowledge exchange is obtained, each agent passes the knowledge to Drools RE for further reasoning (refer to section 4.1.3).

6.3.1 Case Study Description

6.3.1.1 *Cooperative Workcell Layout*

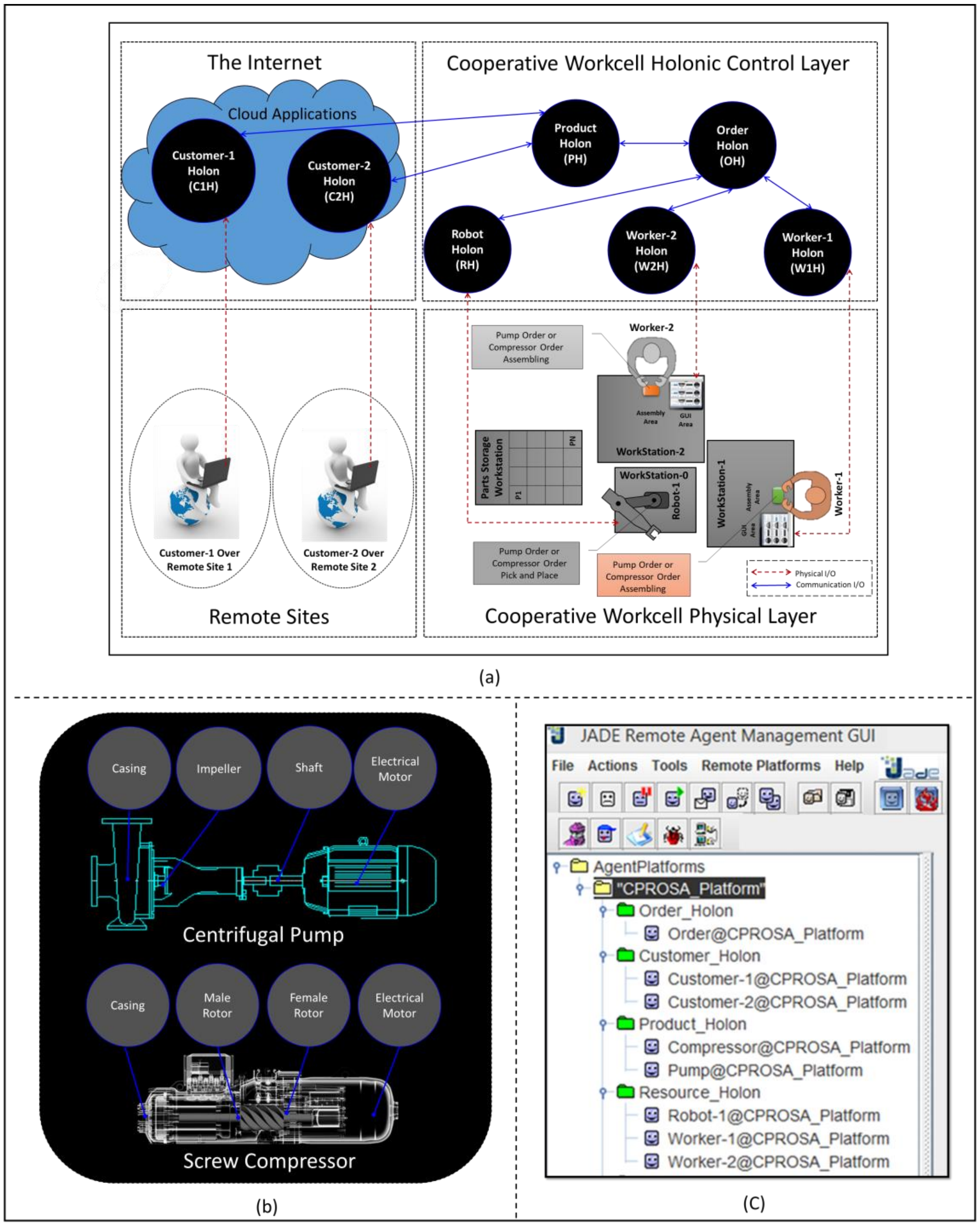
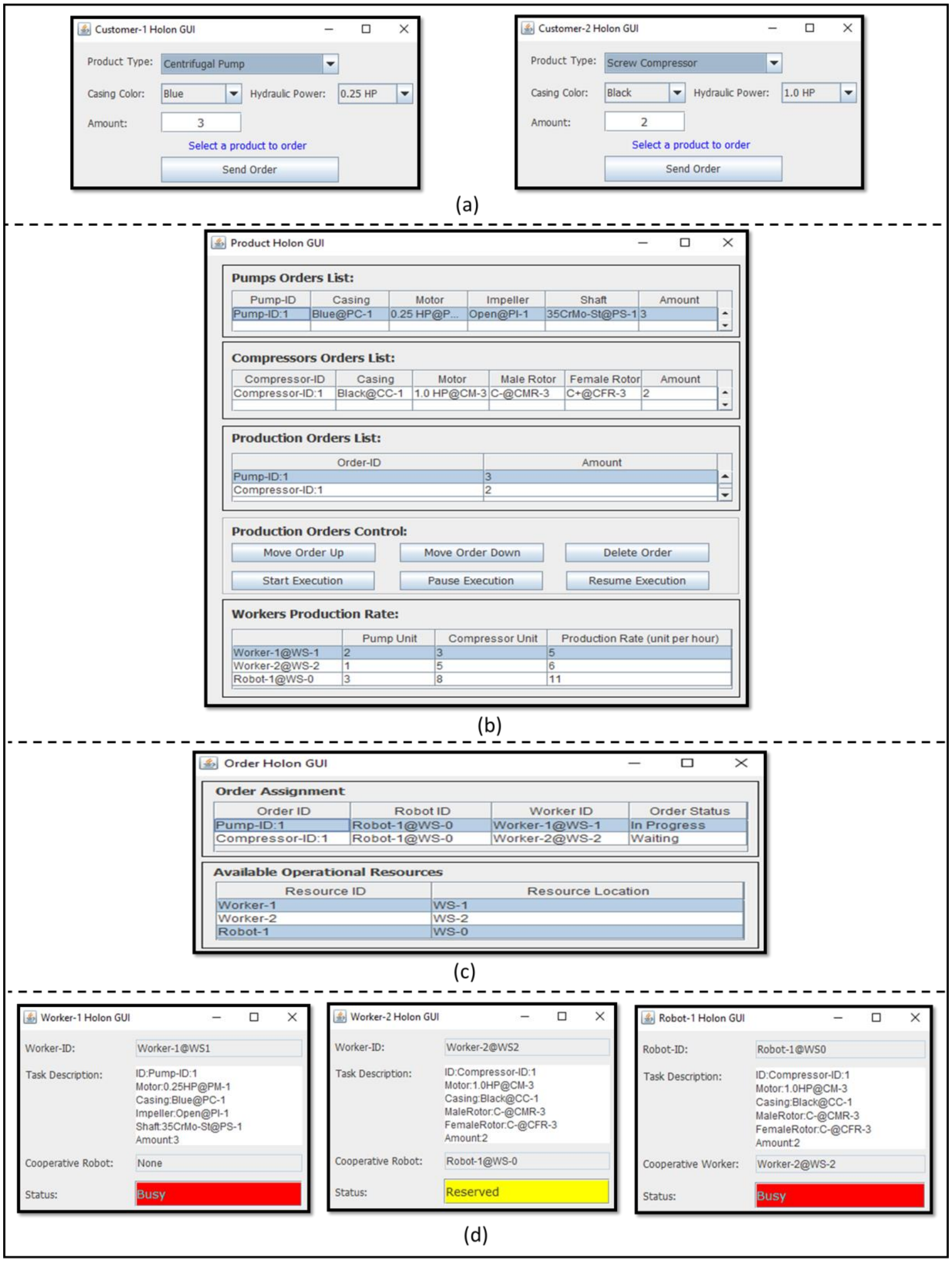


Figure 6.14: (a) Case study layout – (b) Case study product family – (c) Case study deployment over JADE.

During this case study, a cooperative assembly scenario has been assumed as shown in Figure 6.14a. The physical layer of the cooperative workcell includes two workers in cooperation with one robot. The robot locates at the centre of the cooperative workcell over workstation-0. While, worker-1 locates over workstation-1 and worker-2 locates over workstation-2. In addition to another workstation which contains different parts of the same product family. Each of these parts is in a predefined position that has been taught in advance to the robot. Two products are within the product family, which are a centrifugal pump and a screw compressor as shown in Figure 6.14b. The idea of the cooperative scenario is the ability of the robot to handle the parts of the product to the workers, while the two workers focus only on the assembly operations. Based on PROSA model example, three ORHs have been developed which are: **W**orker-**1** **H**olon (**W1H**), **W**orker-**2** **H**olon (**W2H**), and RH, in addition to the OH and the PH. Furthermore, **C**ustomer-**1** **H**olon (**C1H**) and **C**ustomer-**2** **H**olon (**C2H**) have been added to the proposed HCA to build the customer orders. C1H and C2H are meant to link between the customer over a remote site and the HCA solution. For instance, the CHs can be deployed as a cloud application over the internet. JADE framework has been used during this case study to implement the HCA, as shown Figure 6.14c.

6.3.1.2 Cooperative Workcell GUIs

The implementation feasibility of the proposed HCA has been considered in the previous case study. Therefore, during this case study, the hardware has been substituted by a group of GUIs that are responsible for the I/O data and events generation, as shown in Figure 6.15. Two similar GUIs are dedicated to present the CHs, as shown in Figure 6.15a. The GUI of the CH provides a tool to order a specific product with certain features (i.e., parts). The customer selects the basic features and defines the needed amount of the product then sends the production order to the PH. Based on the selected features of the production order, the PH defines the required production recipe to build the production order. Then, it stacks the production orders together to be executed based on FCFS, as shown in Figure 6.15c. The PH also is able to rearrange the orders or modify them before sending them to the OH. The OH is responsible for receiving the production orders from the PH. Thus, the OH assigns the production orders to the available cooperative resources, as shown in Figure 6.15d. The GUIs that are dedicated to the WHs and the RH are used to simulate the existence of the workers and the robot, as shown in Figure 6.15d.



Customer-1 Holon GUI

Product Type:	Centrifugal Pump
Casing Color:	Blue
Hydraulic Power:	0.25 HP
Amount:	3

Select a product to order

Customer-2 Holon GUI

Product Type:	Screw Compressor
Casing Color:	Black
Hydraulic Power:	1.0 HP
Amount:	2

Select a product to order

Product Holon GUI

Pumps Orders List:

Pump-ID	Casing	Motor	Impeller	Shaft	Amount
Pump-ID:1	Blue@PC-1	0.25 HP@P...	Open@PI-1	35CrMo-St@PS-1	3

Compressors Orders List:

Compressor-ID	Casing	Motor	Male Rotor	Female Rotor	Amount
Compressor-ID:1	Black@CC-1	1.0 HP@CM-3	C-@CMR-3	C+@CFR-3	2

Production Orders List:

Order-ID	Amount
Pump-ID:1	3
Compressor-ID:1	2

Production Orders Control:

Workers Production Rate:

	Pump Unit	Compressor Unit	Production Rate (unit per hour)
Worker-1@WS-1	2	3	5
Worker-2@WS-2	1	5	6
Robot-1@WS-0	3	8	11

Order Holon GUI

Order Assignment

Order ID	Robot ID	Worker ID	Order Status
Pump-ID:1	Robot-1@WS-0	Worker-1@WS-1	In Progress
Compressor-ID:1	Robot-1@WS-0	Worker-2@WS-2	Waiting

Available Operational Resources

Resource ID	Resource Location
Worker-1	WS-1
Worker-2	WS-2
Robot-1	WS-0

Worker-1 Holon GUI

Worker-ID:	Worker-1@WS1
Task Description:	ID:Pump-ID:1 Motor:0.25HP@PM-1 Casing:Blue@PC-1 Impeller:Open@PI-1 Shaft:35CrMo-St@PS-1 Amount:3
Cooperative Robot:	None
Status:	Busy

Worker-2 Holon GUI

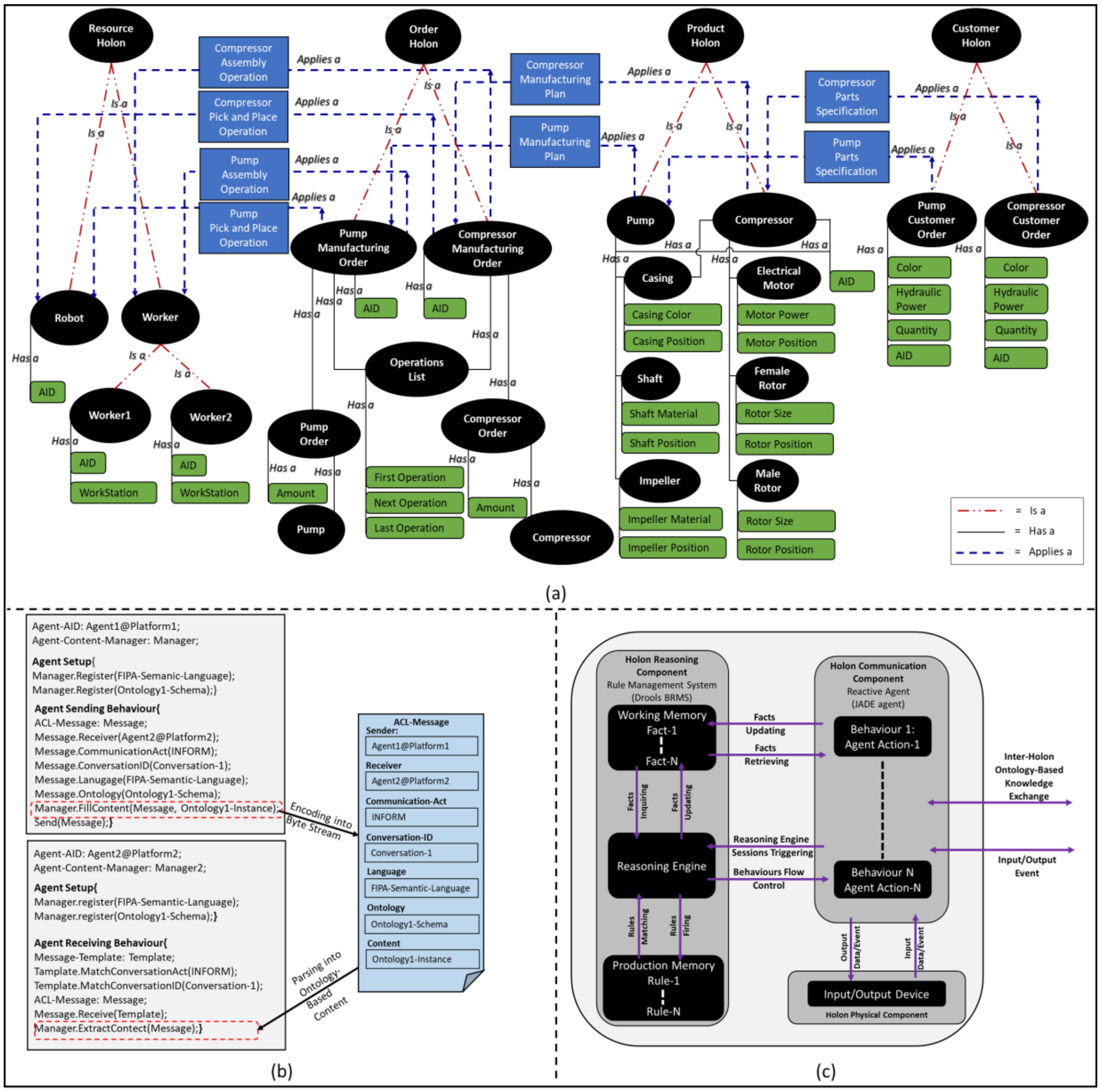
Worker-ID:	Worker-2@WS2
Task Description:	ID:Compressor-ID:1 Motor:1.0HP@CM-3 Casing:Black@CC-1 MaleRotor:C-@CMR-3 FemaleRotor:C-@CFR-3 Amount:2
Cooperative Robot:	Robot-1@WS-0
Status:	Reserved

Robot-1 Holon GUI

Robot-ID:	Robot-1@WS0
Task Description:	ID:Compressor-ID:1 Motor:1.0HP@CM-3 Casing:Black@CC-1 MaleRotor:C-@CMR-3 FemaleRotor:C-@CFR-3 Amount:2
Cooperative Worker:	Worker-2@WS-2
Status:	Busy

Figure 6.15: GUIs for (a) CHs – (b) PH – (c) OH – (d) ORHs.

6.3.2 Case Study Ontology Model



```
Agent-AID: Agent1@Platform1;
Agent-Content-Manager: Manager;

Agent Setup{
Manager.Register(FIPA-Semanic-Language);
Manager.Register(Ontology1-Schema);}

Agent Sending Behaviour{
ACL-Message: Message;
Message.Receiver(Agent2@Platform2);
Message.CommunicationAct(INFORM);
Message.ConversationID(Conversation-1);
Message.Lanugage(FIPA-Semantic-Language);
Message.Ontology(Ontology1-Schema);
Manager.FillContent(Message, Ontology1-Instance);
Send(Message);}
```

```
Agent-AID: Agent2@Platform2;
Agent-Content-Manager: Manager2;

Agent Setup{
Manager.register(FIPA-Semantic-Language);
Manager.register(Ontology1-Schema);}

Agent Receiving Behaviour{
Message-Template: Template;
Tamplate.MatchConversationAct(INFORM);
Template.MatchConversationID(Conversation-1);
ACL-Message: Message;
Message.Receive(Template);
Manager.ExtractContect(Message);}
```

Figure 6.16: (a) Case study ontology model – (b) ACL-Message exchange based on FIPA ontology – (c) JADE and Drools RE integration.

The previously described case study is presented via an ontology model which describes every element in the HCA solution, as shown in Figure 6.16a. The model can be seen as a connected group of schemas. The schema is an abstraction of an object within the manufacturing environment, as will be discussed in details in the following sections. The relation between two terms schemes, or between a term schema and an action schema can be defined as a predicate schema:

- **Is a:** a relation to express the inheritance of a child schema to its parent's parameters and methods. In another words, a child schema is an extension to a parent schema.
- **Has a:** a relation to express a possession of a term schema to a parameter or to another term schema. In another words, a relation between a whole and a part. This relation is different from the previous one, as part does not inherit the properties of the whole.
- **Applies a:** a relation to express an axiom between two schemas. In other words, a relation to define an action or predicate schema that can be done by a term schema.

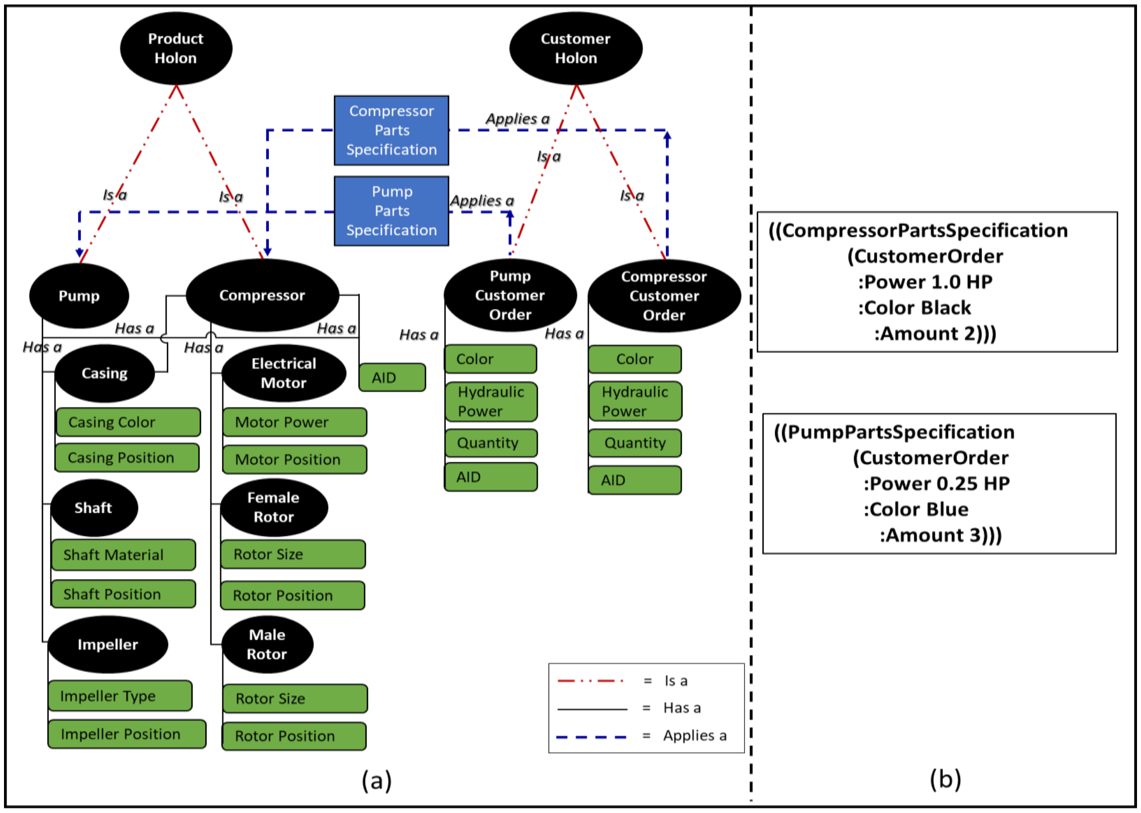
All the schemes in the case study ontology model are written in FIPA-SL. Therefore, all the JADE agents within the HCA solution must define FIPA-SL as the language of conversation. This definition typically takes place during JADE agent setup method, as shown in Figure 6.16b. Other languages such as **LEAP** (**L**ightweight **E**xtensible **A**gent **P**latform) can be also used in JADE as a conversation language. However, JADE-LEAP is not a human readable that makes it unsuitable for this cooperative scenario. Programming JADE agent based on ontology schemas is similar to OOP, in the sense that both the agents must agree about the same schemas before any conversation. Then, every JADE agent builds an instant of this schema when it needs to talk with another JADE agent, as shown in the example in Figure 6.16b.

After the conversation between two JADE agents occurs, JADE agent may need to extract facts from the ontology-based ACL-Message. This extraction is achieved by one of the agent behaviours that is triggered by an incoming input event. In order to store and update these facts, JADE agent feeds these facts to Drools working memory and fires Drools RE. Therefore, a reasoning session is conducted which derives new facts based on matching the rules on Drools production memory. Ultimately, new facts are updated at Drools working memory. Accordingly, JADE agent builds new ontology-based ACL-Message with these new facts when it is needed. Integrating JADE and Drools RE does not represent any technical problem from the coding point of view. As both the frameworks are writing in JAVA. Integrating JADE and Drools RE can be summarized in Figure 6.16c.

Due to the case study ontology model, the proposed cooperative assembly scenario can be divided into three stages. The first stage is dedicated to specifying the parts of the production order. In this stage, the customer orders are received by the PH. Then, based on the minimum information that has been selected by the customer, the PH specifies the position and the other parameters of every part. Thus, the workers and the robot know the parts to use to assemble this customer order. In the second stage of the cooperative assembly scenario, the PH plans the manufacturing by selecting the necessary operation to be carried over every production order. Then, the PH sends this production plan (i.e., recipe) to the OH. Thus, the OH distributes and organises the manufacturing operations among the available cooperative resources.

6.3.2.1 *Parts Specification Ontology*

Figure 6.17a, is composed of the three types of JADE schemas that have been discussed in details in section 4.2.3. Term and action schemas are linked together with the predicate schemes that have been explained earlier. The term schemas that have been used during this stage of the ontology model are as follows:



```
((CompressorPartsSpecification
    (CustomerOrder
      :Power 1.0 HP
      :Color Black
        :Amount 2)))
```

```
((PumpPartsSpecification
    (CustomerOrder
      :Power 0.25 HP
      :Color Blue
        :Amount 3)))
```

Figure 6.17: (a) Product parts specification ontology – (b) ACL-Messages contain the customer orders.

- **Compressor-Customer-Order:** a schema that encapsulates some attributes such as the required compressor color, the needed hydraulic power, and the required amount of units. In addition, it contains an AID as every compressor-customer- order is an agent that needs a unique ID.
- **Pump-Customer-Order:** a schema that encapsulates some attributes such as the required pump color, the needed hydraulic power, and the required amount of units. In addition, it contains an AID as every pump-customer-order is an agent that needs a unique ID.
- **Casing:** a shared part between the pump and the compressor. The casing schema contains two attributes, which are the casing color, and its position at the parts storage workstation.

- **Electrical-Motor:** a shared part between the pump and the compressor. The motor schema contains two attributes, which are the motor electrical power, and its position at the parts storage workstation.
- **Shaft:** a unique part of the pump. The shaft schema contains two attributes that are the shaft material, and its position at the storage workstation.
- **Impeller:** a unique part of the pump. The impeller schema contains two attributes that are the impeller type, and position at the features workspace or storage.
- **Female-Rotor:** a unique part of the compressor. The female-rotor schema contains two attributes, which are the rotor size, and its position at the storage workstation.
- **Male-Rotor:** a unique part of the compressor. The male-rotor schema contains two attributes, which are the rotor size, and its position at the parts storage workstation.
- **Compressor:** a concept schema that encapsulates many other schemas under it, those schemas are the casing, electrical-motor, female-rotor, and male-rotor. Every compressor is an agent that needs an AID attribute as well.
- **Pump:** a concept schema which encapsulates many other schemas under it, those schemas are the casing, electrical-motor, shaft, and impeller. Every pump is an agent that needs an AID attribute as well.

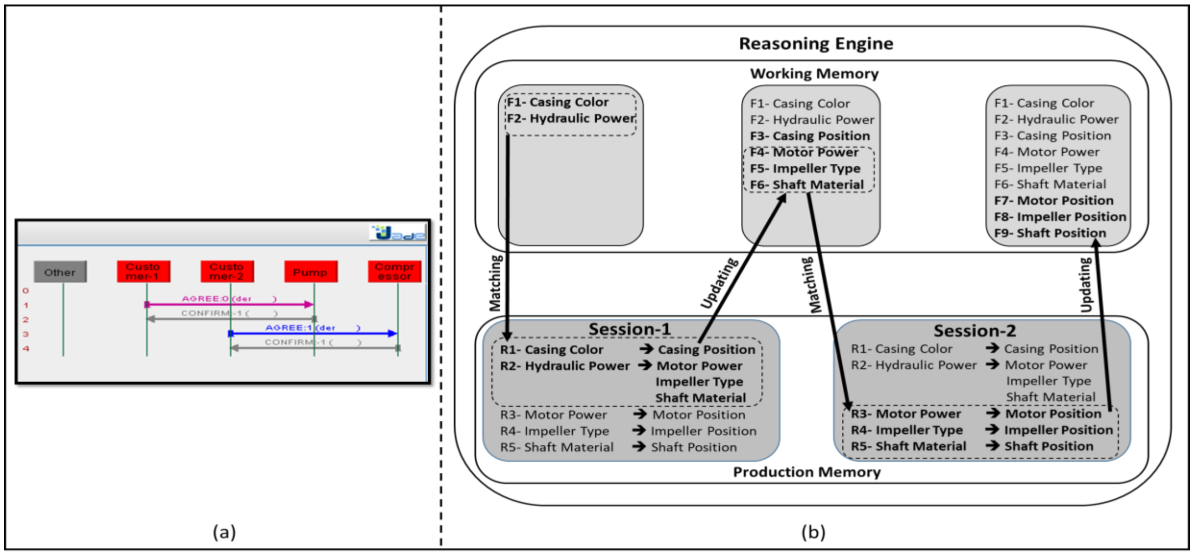


Figure 6.18: (a) JADE interaction between the CHs and the PH – (b) Drools RE mechanism.

The action schemas that have been used during the parts specification ontology are:

- **Compressor-Parts-Specification:** this action schema expects a Compressor-Customer-Order concept schema as an input, and it can be deployed by either C1H or C2H. An instance of this action schema can be seen at the ACL-Message content in at the top of Figure 6.17b.

- **Pump-Parts-Specification:** this action schema expects a Pump-Customer-Order concept schema as an input, and it is deployed by either C1H or C2H. An instance of this action schema can be seen at the ACL-Message at the bottom of Figure 6.17b.

Figure 6.18a shows JADE interaction scenario among the CHs (i.e., customer-1 agent and customer-2 agent) and the PH (i.e., pump agent and compressor agent). In this scenario, customer-1 agent sends an Agree -Message to the pump agent. The content of this Agree-Message is the Pump-Customer-Order that has been shown in Figure 6.17b. When the Agree-Message is received by the pump agent, the pump agent confirms the receiving by sending a Confirm-Message back to customer-1 agent. When the pump agent receives this Agree-Message, it applies the Pump-Parts-Specification action schema to reason the contents of that message.

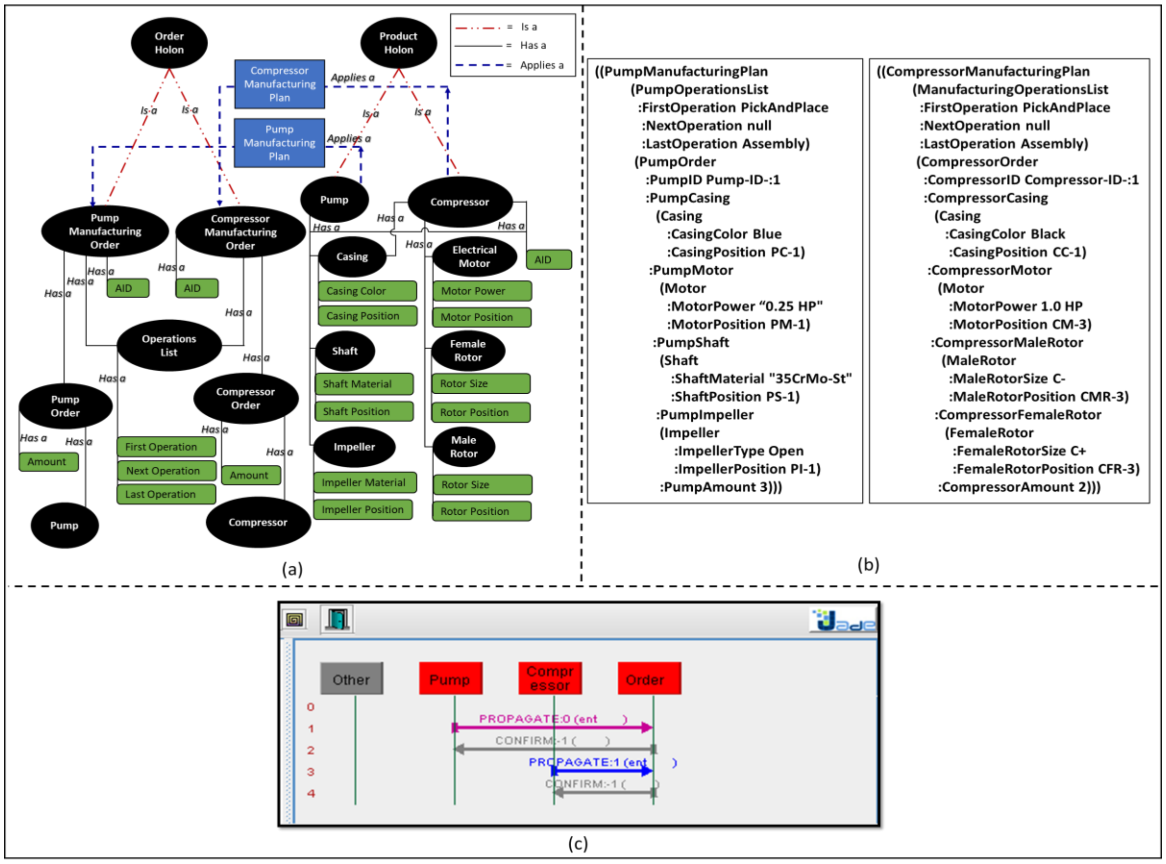
The reasoning mechanism is done as a forward chaining by Drools RE via two different sessions as shown in Figure 6.18b. Two facts can be extracted from the received customer order Agree-Message, which are the casing color and the hydraulic power. Those two facts are inserted into reasoning session-1. The casing color fact derives new fact which is the casing position. While, the hydraulic power fact derives other three facts which are the motor power, the impeller type, and the shaft material. Those new facts are being updated to Drools working memory. When Drools realizes that new facts in the working memory are matching some of the rules in the production memory, it triggers the reasoning session-2. In this session, the motor power fact derives the motor position fact, the impeller type fact derives the impeller position, and the shaft material derives the shaft position. The new facts at Drools working memory can be used to formulate the pump manufacturing plan, as it will be shown in the next section. The same previous mechanism is followed again between cusomter-2 agent and the compressor agent to derive the compressor-parts-specification.

6.3.2.2 Manufacturing Planning Ontology

Figure 6.19a contains the term and the action schemas that has been used to build the manufacturing planning ontology. The term schemas that have been used during this stage of the ontology model are as follows:

- **Compressor-Order:** a schema that extends the compressor schema by adding the required amount of compressor units.
- **Pump-Order:** a schema that extends the pump schema by adding the required amount of pump units.
- **Operations-List:** a schema that includes a list of operations that can be used to manufacture either a pump or a compressor. The schema can be used to manufacture a product that needs three operations or less.

- **Compressor-Manufacturing-Order:** a schema which combines a Compressor-Order schema and an Operations-List schema. In addition, it has an AID attribute as it acts as an agent.
- **Pump-Manufacturing-Order:** a schema which combines a Pump-Order schema and an Operations-List schema. In addition, it has an AID attribute as it acts as an agent.



```
((PumpManufacturingPlan
    (PumpOperationsList
      :FirstOperation PickAndPlace
      :NextOperation null
      :LastOperation Assembly)
    (PumpOrder
      :PumpID Pump-ID-:1
      :PumpCasing
        (Casing
          :CasingColor Blue
          :CasingPosition PC-1)
      :PumpMotor
        (Motor
          :MotorPower "0.25 HP"
          :MotorPosition PM-1)
      :PumpShaft
        (Shaft
          :ShaftMaterial "35CrMo-St"
          :ShaftPosition PS-1)
      :PumpImpeller
        (Impeller
          :ImpellerType Open
          :ImpellerPosition PI-1)
      :PumpAmount 3)))
```

```
((CompressorManufacturingPlan
    (ManufacturingOperationsList
      :FirstOperation PickAndPlace
      :NextOperation null
      :LastOperation Assembly)
    (CompressorOrder
      :CompressorID Compressor-ID-:1
      :CompressorCasing
        (Casing
          :CasingColor Black
          :CasingPosition CC-1)
      :CompressorMotor
        (Motor
          :MotorPower 1.0 HP
          :MotorPosition CM-3)
      :CompressorMaleRotor
        (MaleRotor
          :MaleRotorSize C-
          :MaleRotorPosition CMR-3)
      :CompressorFemaleRotor
        (FemaleRotor
          :FemaleRotorSize C+
          :FemaleRotorPosition CFR-3)
      :CompressorAmount 2)))
```

Figure 6.19: (a) Manufacturing planning ontology – (b) ACL-Messages contain the manufacturing plan - (c) JADE Interaction between the PH and the OH.

The action schemas that have been used during the manufacturing planning ontology are as follows:

- **Compressor-Manufacturing-Plan:** this action schema expects a Compressor-Order and a Compressor-Operations-List concept schemas as inputs, and it is deployed by the compressor agent. An instance of this action schema can be seen at the ACL-Message at the right side of Figure 6.19b.
- **Pump-Manufacturing-Plan:** this action schema expects a Pump-Order and a Pump-Operations-List concept schema as inputs, and it is deployed by the pump agent. An instance of this action schema can be seen at the ACL-Message at the left side of Figure 6.19b.

Figure 6.19c shows JADE interaction scenario among the PH (i.e., pump agent and compressor agent) and the OH (i.e., order agent). In this scenario, the pump agent sends a Propagate-Message to the order agent. The contents of this Propagate-Message is the Pump-Manufacturing-Plan that has been shown in Figure 6.19b. When the Propagate-Message is received by the order agent, the order agent confirms the receiving by sending a Confirm-Message back to the pump agent. The Pump-Manufacturing-Plan is built upon the new facts that have been derived during Drools reasoning that has been shown earlier.

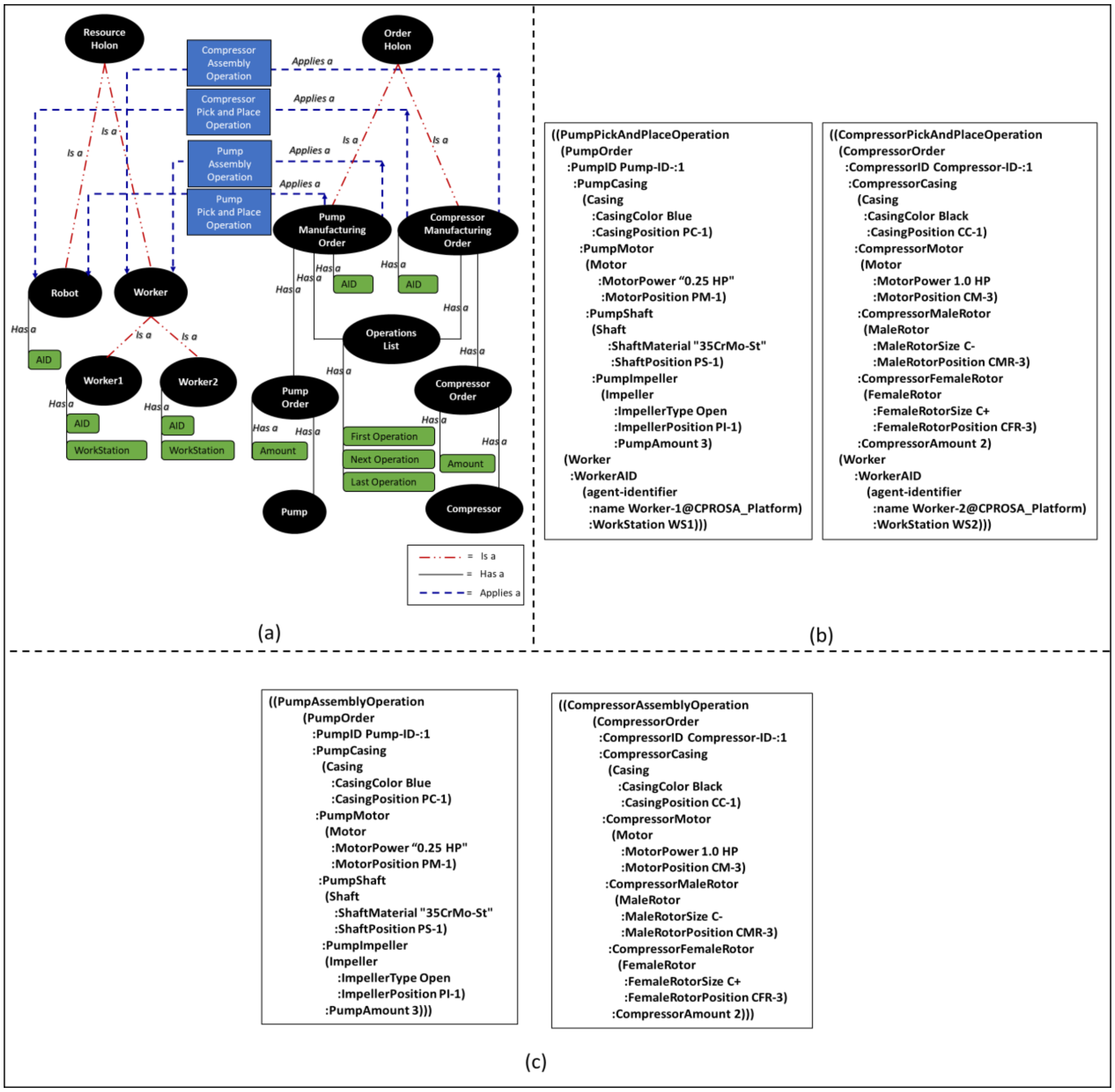
The same previous mechanism is followed again between the compressor agent and the order agent to build the Compressor-Manufacturing-Plan.

6.3.2.3 Manufacturing Execution Ontology

Figure 6.20a contains the term and the action schemas that has been used to build the manufacturing execution ontology. The term schemas that have been used during this stage of the ontology model are as follows:

- **Worker:** a schema that contains two attributes, the first one is the worker AID as it acts as an agent, and the second is the worker location within the workcell (i.e., workstation). The worker GUI provides the worker with the assigned task and inquires the task done event (see Figure 6.15d). Two instances of the WH exist in this case study scenario. A WH can have three statuses. A free status when there are no product orders or the production is not started. A reserve status when the worker is waiting to get the first product parts. A busy status lasts while the robot is handling the orders until the worker triggers the task-done button.
- **Robot:** a schema that contains one attribute that is the robot AID as it acts as an agent. The robot schema does not have a workstation attribute, because there is only one robot that is responsible for the pick and place. Therefore, the location of the robot is irrelevant. However, in case of more than one robot this attribute could be important. The robot GUI shows the robot assigned task and the status of the robot (see Figure 6.15d). The robot can have two statuses. A free status when there are no product orders or the production is not started. A busy status when the robot is handling the production orders. As this case study does not involve a real robot hardware, it has been assumed that every pick and place operation takes a constant time of 2.0 minutes. Therefore, a software timer is implemented by the RH to simulate the pick and place operation.

The action schemas that have been used during the parts specification ontology are as follows:



```
((PumpPickAndPlaceOperation
  (PumpOrder
   :PumpID Pump-ID-:1
    :PumpCasing
     (Casing
      :CasingColor Blue
      :CasingPosition PC-1)
    :PumpMotor
     (Motor
      :MotorPower "0.25 HP"
      :MotorPosition PM-1)
    :PumpShaft
     (Shaft
      :ShaftMaterial "35CrMo-St"
      :ShaftPosition PS-1)
    :PumpImpeller
     (Impeller
      :ImpellerType Open
      :ImpellerPosition PI-1)
      :PumpAmount 3)
  (Worker
   :WorkerAID
    (agent-identifier
     :name Worker-1@CPROSA_Platform)
     :WorkStation WS1)))
```

```
((CompressorPickAndPlaceOperation
  (CompressorOrder
   :CompressorID Compressor-ID-:1
    :CompressorCasing
     (Casing
      :CasingColor Black
      :CasingPosition CC-1)
    :CompressorMotor
     (Motor
      :MotorPower 1.0 HP
      :MotorPosition CM-3)
    :CompressorMaleRotor
     (MaleRotor
      :MaleRotorSize C-
      :MaleRotorPosition CMR-3)
    :CompressorFemaleRotor
     (FemaleRotor
      :FemaleRotorSize C+
      :FemaleRotorPosition CFR-3)
    :CompressorAmount 2)
  (Worker
   :WorkerAID
    (agent-identifier
     :name Worker-2@CPROSA_Platform)
     :WorkStation WS2)))
```

```
((PumpAssemblyOperation
  (PumpOrder
   :PumpID Pump-ID-:1
   :PumpCasing
    (Casing
     :CasingColor Blue
     :CasingPosition PC-1)
   :PumpMotor
    (Motor
     :MotorPower "0.25 HP"
     :MotorPosition PM-1)
   :PumpShaft
    (Shaft
     :ShaftMaterial "35CrMo-St"
     :ShaftPosition PS-1)
   :PumpImpeller
    (Impeller
     :ImpellerType Open
     :ImpellerPosition PI-1)
   :PumpAmount 3)))
```

```
((CompressorAssemblyOperation
  (CompressorOrder
   :CompressorID Compressor-ID-:1
   :CompressorCasing
    (Casing
     :CasingColor Black
     :CasingPosition CC-1)
   :CompressorMotor
    (Motor
     :MotorPower 1.0 HP
     :MotorPosition CM-3)
   :CompressorMaleRotor
    (MaleRotor
     :MaleRotorSize C-
     :MaleRotorPosition CMR-3)
   :CompressorFemaleRotor
    (FemaleRotor
     :FemaleRotorSize C+
     :FemaleRotorPosition CFR-3)
   :CompressorAmount 2)))
```

Figure 6.20: (a) Manufacturing execution ontology – (b) ACL-Messages contain pick and place operations – (c) ACL-Messages contain assembly operations.

- **Compressor-Pick-And-Place-Operation:** this action schema expects two concept schemas as inputs; the first input is the Compressor-Order concept schema which contains the detailed specifications of the compressor. Therefore, the robot can use this information especially the compressor parts positions to perform the pick operation. The second concept schema input is the target worker. Therefore, the robot can use the worker workstation location to place the compressor parts at this location. This action schema is deployed by the OH to assign a task to the RH. A detailed example of this operation can be seen at the right side Figure 6.20b.

- **Pump-Pick-And-Place-Operation:** this action schema expects two concept schemas as inputs; the first input is the Pump-Order concept schema, which contains the detailed specifications of the pump. Therefore, the robot can use this information, especially the pump parts positions, to perform the pick operation. The second concept schema input is the target worker. Therefore, the robot can use the worker workstation location to place the pump parts at this location. This action schema is deployed by the OH to assign a task to the RH. A detailed example of this operation can be seen at can be seen at the left side Figure 6.20b.
- **Compressor-Assembly-Operation:** this action schema expects one input that is the Compressor-Order concept schema. This operation informs the worker with the required knowhow knowledge to build a customized compressor. Moreover, it informs the worker with the required amount of compressor units. This action schema is deployed by the OH to assign a task to a WH. A detailed example of this operation can be seen at the right side of Figure 6.20c.
- **Pump-Assembly-Operation:** this action schema expects one input that is the Pump-Order concept schema. This operation informs the worker with the required knowhow knowledge to build a customized pump. Moreover, it informs the worker with the required amount of pump units. This action schema is deployed by the OH to assign a task to a WH. A detailed example of this operation can be seen at the left side of Figure 6.20c.

6.3.3 Holons Interaction

Figure 6.21a shows JADE interaction scenario among the OH and the ORHs (i.e., worker-1 agent, worker-2 agent, and robot agent). During this interaction, the manufacturing operations are assigned to the operational resources based on their status. As shown in lines 1-4 of Figure 6.21a, the orders agent sends two Request-Messages which are replied by two Confirm-Messages. The first Request-Message assigns a Pump-Pick-And-Place-Operation to the robot agent. The second Request-Message assigns a Pump-Assembly-Operation to worker-1 agent. The pump-order has been processed first by the PH as it is the first product order at the order list (refer to Figure 6.15b). In line 5 of Figure 6.21a, the robot agent sends an Inform-Ref-Message to worker-1 agent to inform that the first pump parts have been placed. Then, the robot agent sends two Inform-If-Messages to the orders agent and worker-1 agent to inform that it finished handling the entire required pump (i.e., three pump units by referring to Figure 6.15). The two Inform-If-Messages can be seen in lines 6, and 7 of Figure 6.21a. The same interaction mechanism in lines 9-13 is followed to assign the compressor-order manufacturing operations to worker-2 agent and the robot agent. Lines 14, and 15 of Figure 6.21a show the Inform-Messages that denotes a task done event that is generated from worker-1 and worker-2 agents to the orders agent.

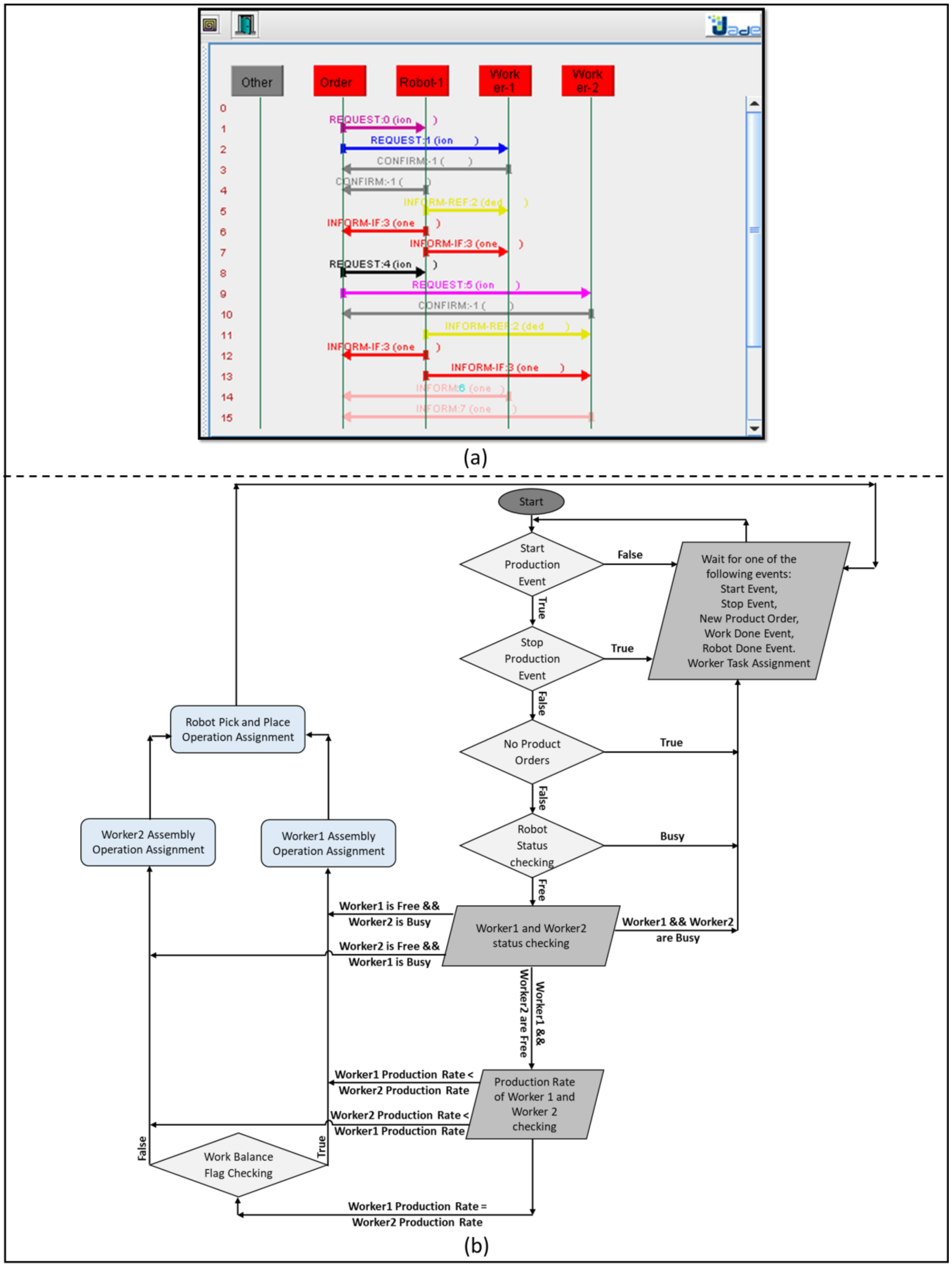

Other
Order
Robot-1
Worker-1
Worker-2
REQUEST:0 (ion)
REQUEST:1 (ion)
CONFIRM:-1 ( )
CONFIRM:-1 ( )
INFORM-REF:2 (ded)
INFORM-IF:3 (one)
INFORM-IF:3 (one)
REQUEST:4 (ion)
REQUEST:5 (ion)
CONFIRM:-1 ( )
INFORM-REF:2 (ded)
INFORM-IF:3 (one)
INFORM-IF:3 (one)
INFORM:6 (one)
INFORM:7 (one)
(a)
Start
Start Production Event
False
True
Wait for one of the following events: Start Event, Stop Event, New Product Order, Work Done Event, Robot Done Event. Worker Task Assignment
Stop Production Event
True
False
No Product Orders
True
False
Robot Status checking
Busy
Free
Robot Pick and Place Operation Assignment
Worker2 Assembly Operation Assignment
Worker1 Assembly Operation Assignment
Worker1 is Free && Worker2 is Busy
Worker2 is Free && Worker1 is Busy
Worker1 and Worker2 status checking
Worker1 && Worker2 are Busy
Worker1 && Worker2 are Free
Production Rate of Worker 1 and Worker 2 checking
Worker1 Production Rate < Worker2 Production Rate
Worker2 Production Rate < Worker1 Production Rate
Work Balance Flag Checking
False
True
Worker1 Production Rate = Worker2 Production Rate
(b)

Figure 6.21: (a) JADE interaction between the OH and the ORHs – (b) An algorithm implement via Drools to control the interaction between the OH and the ORHs.

Reasoning the interaction between the OH and the ORHs is done by Drools RE. The reasoning rules that have been used to control this interaction are summarized in the algorithm shown in Figure 6.21b, as it is simpler to sum all the rules in one flow chart. The algorithm begins by checking if the start production condition is true, and if the stop production condition is false. If this condition is true, the algorithm checks if there are production orders that are waiting for execution. If not, Drools RE waits a new event to be received by the OH to start the algorithm again. After checking all the previously mentioned conditions, Drools RE checks the status of the robot agent to assign the first order.

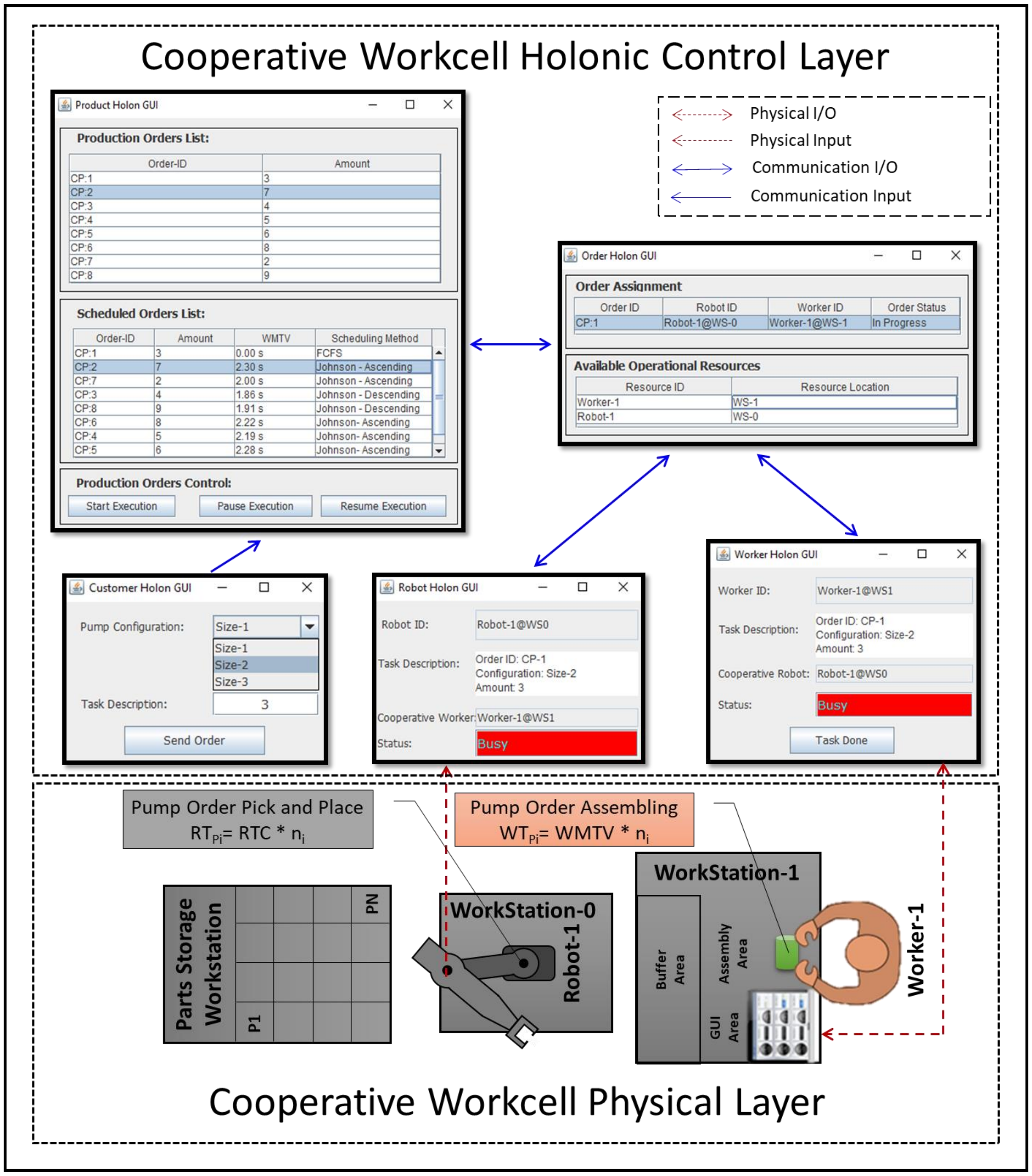
IF the robot agent is free, Drools RE checks the status of both workers. If worker-1 agent is free and worker-2 agent is busy, Drools RE assigns the Assembly-Operation to worker-1 agent and starts the algorithm again. If worker-1 agent is busy and worker-2 agent is free, Drools RE assigns the Assembly-Operation to worker-2 agent and starts the algorithm again. If both the workers are busy, Drools RE waits a new event to be received by the OH to start the algorithm again. If both the workers are free, Drools RE checks the worker with the highest production rate. As the production rate of the worker is continuously monitored by the PH as it was shown in Figure 6.15b. Drools RE uses the production rate fact to solve the case that two workers are free at the same time. Therefore, it assigns the task to the worker with less production rate. Hence, the algorithm tries to obtain the balance in distributing the tasks between the two workers.

In case that both the workers are free and they have been equal production rate, Drools RE assigns randomly to one of them, only the first time this case happens. Then, the next time the same case happens, Drools RE assigns to the other worker. After that, Drools RE keeps switching between the two workers every time this case happens.

6.4 Case Study 3: Cooperative Workcell Scheduling

This case study is focused on the production scheduling during the cooperative manufacturing. This will be discussed via a case study that is shown in Figure 6.22. The case study is based on modifying the previously discussed case study in section 6.3. However, in order to simplify the example and focus on its purpose, the knowledge representation is excluded from this case study. As the production scheduling was not the concern of the case study in section 6.3, the production orders have been assigned to the workers and the robot based on FCFS method. Nevertheless, in this case study, Johnson's scheduling that has been explained earlier in section 5.3.3 will be deployed by the proposed HCA. More specifically, Johnson's scheduling will be converted to a set of Drools rules that fit the case study and implemented by the proposed HCA.

6.4.1 Case Study Description



Production Orders List:

Order-ID	Amount
CP:1	3
CP:2	7
CP:3	4
CP:4	5
CP:5	6
CP:6	8
CP:7	2
CP:8	9

Scheduled Orders List:

Order-ID	Amount	WMTV	Scheduling Method
CP:1	3	0.00 s	FCFS
CP:2	7	2.30 s	Johnson - Ascending
CP:7	2	2.00 s	Johnson - Ascending
CP:3	4	1.86 s	Johnson - Descending
CP:8	9	1.91 s	Johnson - Descending
CP:6	8	2.22 s	Johnson- Ascending
CP:4	5	2.19 s	Johnson- Ascending
CP:5	6	2.28 s	Johnson- Ascending

Production Orders Control:

Order Assignment

Order ID	Robot ID	Worker ID	Order Status
CP:1	Robot-1@WS-0	Worker-1@WS-1	In Progress

Available Operational Resources

Resource ID	Resource Location
Worker-1	WS-1
Robot-1	WS-0

Pump Configuration: Size-1
Size-1
Size-2
Size-3
Task Description: 3

Robot ID: Robot-1@WS0
Task Description: Order ID: CP-1
Configuration: Size-2
Amount: 3
Cooperative Worker: Worker-1@WS1
Status: Busy

Worker ID: Worker-1@WS1
Task Description: Order ID: CP-1
Configuration: Size-2
Amount: 3
Cooperative Robot: Robot-1@WS0
Status: Busy

Figure 6.22: Case study layout structure.

The physical layer of the cooperative workcell includes one worker in cooperation with one robot. Both the worker and the cobot perform a pump assembly scenario. The role of the cobot is to handle the pump parts to be assembled by the worker. The number of the required pumps are varied due to the customer's need. As the cobot's speed can be adjusted in advance, the time taken from the robot to handle a specific amount of pumps is assumed to be known. However, in the worker case, the time to assemble a production order is continuously varying. Therefore, this time will be estimated by the PH based on the previous tasks history, then will be used to schedule the production orders list.

The control layer of the cooperative workcell provides a set of GUIs that are used to test different scheduling cases. One GUI is dedicated for the customer who sends a different order to the PH. The customer orders are initially stacked in a production list via the PH. The PH schedules this production list every time a new customer order is received and every time the worker finishes an assigned task. The scheduling will be explained in more details later via Drools decision table. After scheduling, the PH sends the production order on the top of the list to the OH. The OH assigns the production order as a pick and place task to the robot, whose status turns to be busy. The robot places the production order at the buffer area to be assembled by the worker when he is free.

6.4.2 Workcell Scheduling

6.4.2.1 Drools Decision-Making Table

An important advantage of Drools is that it can be accessed by the different holons. This means that the working memory facts can be updated and retrieved, and Drools RE can be fired by all the holons. That gives the proposed solution the privilege to take a collective decision. Drools production memory stores a set of rules that decide its actions due to the input events and the overall status of the workcell. These rules are summarized in Table 6.4 and explained as follows:

- **Rule-1:** if a new order event is received from the customer, while the worker did not assemble any order yet and the robot is in free status, then the production list will be scheduled based on FCFS, and the scheduled production order will be assigned as a pick and place task to the robot. This rule takes place at the very beginning of the production, or if the worker is still assembling the first production order while the robot finished handling all the parts.
- **Rule-2:** the same as rule-1 except that the robot is busy. Thus, the production list will be scheduled based on FCFS, and waits the robot to be in free status to assign a robot task. This rule takes place while the robot is still handling the very first production order.
- **Rule-3:** if a new order event is received from the customer, while the worker finished assembling at least one production order and the robot is in free status, then the production list will be scheduled based on Johnson's rules, and the scheduled production order will be assigned as a pick and place task to the robot. This rule takes place after the WMTV has been calculated and the robot is free.

Table 6.4: Drools decision-making table.

Rule	Event	Cooperative Workcell State			Action	State Explanation
		Worker Done Counter	Cobot Status	Worker Status		
Rule-1	**New Customer Order**	Equal 0	Free	Don't Care	• FCFS scheduling • Robot task assignment	• New order event is received at the very beginning of the production scenario. • New order event is received after the cobot is done with all the previously assigned tasks, and the worker has not finished the first assigned task.
Rule-2	**New Customer Order**	Equal 0	Busy	Don't Care	• FCFS scheduling	• New order event is received while the cobot is executing one previously assigned task, and the worker has not finished the first assigned task.
Rule-3	**New Customer Order**	Greater than or Equal to 0	Free	Don't Care	• WMTV Calculation • Johnson scheduling • Robot task assignment	• New order event is received after the cobot is done with all the previously assigned task, and the worker has finished at least the first assigned order.
Rule-4	**New Customer Order**	Greater than or Equal to 0	Busy	Don't Care	• WMTV Calculation • Johnson scheduling	• New order event is received while the cobot is executing a previously assigned task, and the worker has finished at least the first assigned task.
Rule-5	**Robot Task Done**	Equal 0	Don't Care	Free	• FCFS scheduling • Worker task assignment	• The cobot finished one assigned order while the worker is free and has not finished any previous task.
Rule-6	**Robot Task Done**	Equal 0	Don't Care	Busy	• FCFS scheduling	• The robot finished one assigned order while the worker is executing the first assigned order.
Rule-7	**Robot Task Done**	Greater than or Equal to 0	Don't Care	Free	• WMTV Calculation • Johnson scheduling • Worker task assignment	• The robot finished one assigned order while the worker is free and has finished at least one assigned order.
Rule-8	**Robot Task Done**	Greater than or Equal to 0	Don't Care	Busy	• WMTV Calculation • Johnson scheduling	• The robot finished one assigned order while the worker is busy and has finished at least one assigned order.

- **Rule-4:** the same as rule-3 except that the robot is in busy status. Thus, the production list will be scheduled based on Johnson's rules, and waits the robot to be in free status to assign a robot task. This rule takes place while the robot is still handling a pump order and the WMTV has been calculated.
- **Rule-5:** if the robot task done event is received from the robot, while the worker did not assemble any orders yet and still in a free status, then the production list will be scheduled based on FCFS, and the scheduled production order will be assigned as an assembly task to the worker. This rule takes place at the very beginning of the production when the robot handles the first production order and the worker did not start the assembly task yet.
- **Rule-6:** the same as rule-5 except that the worker is busy. Thus, the production list

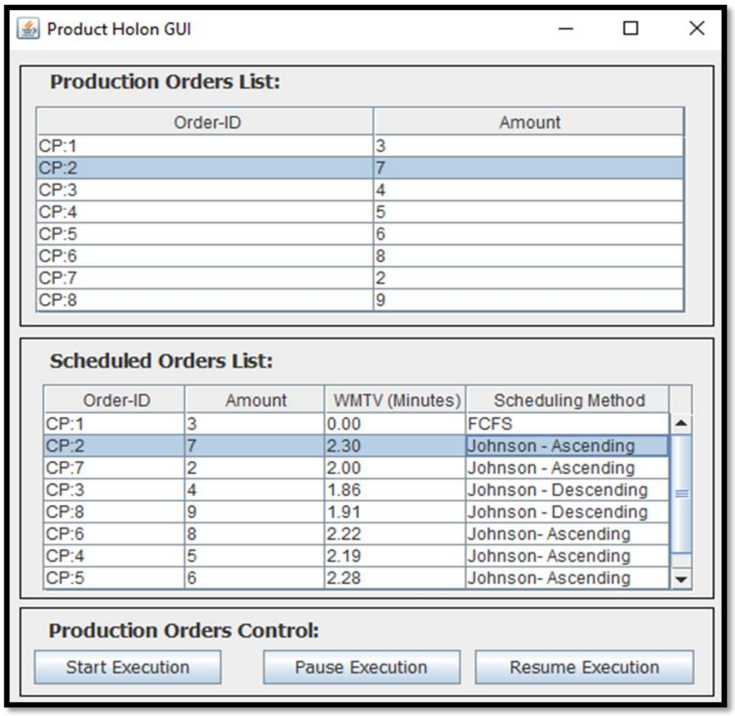
will be scheduled based on FCFS, and waits the worker to be in free status to assign a worker's task. This rule takes place at the very beginning of the production when the robot handles the first production order and the worker is still performing the first assembling task.

- **Rule-7:** if the robot task done event is received from the robot, while the worker finished at least to assemble one pump order and in a free status, then the production list will be scheduled based on Johnson's rules, and the scheduled production order will be assigned as an assembly task to the worker. This rule takes place after the WMTV has been calculated and the worker is free.
- **Rule-8:** the same as rule-7 except that the worker is in busy status. Thus, the production list will be scheduled based on Johnson's rules, and waits the worker to be in free status to assign a worker's task. This rule takes place while the worker is still assembling a production order and the WMTV has been calculated.

6.4.2.2 Scheduling Results

Figure 6.23 shows the case study scheduling results due to the WMTV calculation that is explained in Table 6.5. There are 8 received orders in Figure 6.23a which start with order-ID (CP:1) and end with order-ID (CP:8). Moreover, the value of RTC is assumed to be 2.0 min. By starting the production execution, the value of WMTV equals to 0.0 min, thus CP:1 is scheduled as FCFS. When the robot was done with handling CP:1, the value of WMTV was still 0.0 min as the worker had not assembled any order yet. Therefore, CP:2 was scheduled as the next order via FCFS.

When the robot was done with handling CP:2, the value of WMTV could be calculated. The details of calculating WMTV can be seen via the first raw in Table 6.5. The WMTV is calculated after 20.0 min from starting the production. This is the exact time when the robot finished handling CP:2 to the buffer. At this this time, the number of units that were already assembled by the worker was the overall number of units in CP:1 (i.e., 3 units) plus another 3 units from CP:2. Therefore, WMTV can be obtain by subtracting 6 min from 20 min of the first starving delay, then dividing the result by 6 units. The value of WMTV was 2.3 min that is greater than RTC (i.e., 2.0 min). Therefore, the production orders were scheduled based on Johnson's rules by ascending the values of the required unit, as explained in details in section 5.3.3. Accordingly, CP:7 was scheduled next, as it contains the lowest number of units (i.e., 2 units).


Product Holon GUI

Production Orders List:

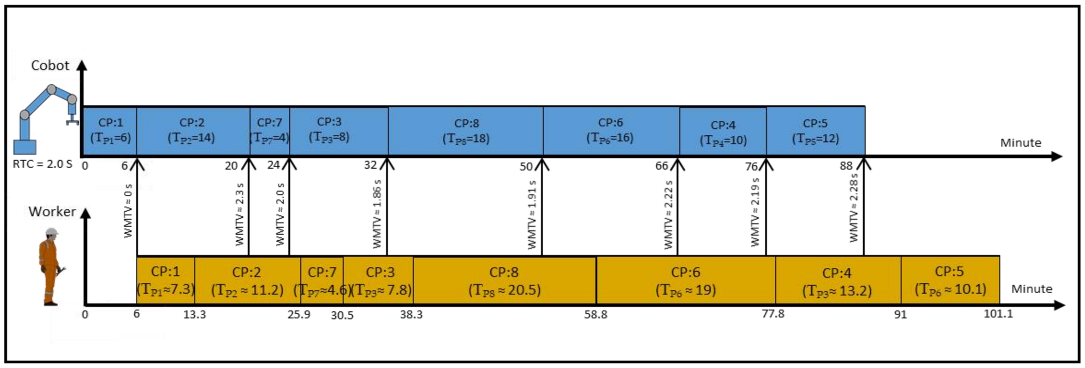
Order-ID	Amount
CP:1	3
CP:2	7
CP:3	4
CP:4	5
CP:5	6
CP:6	8
CP:7	2
CP:8	9

Scheduled Orders List:

Order-ID	Amount	WMTV (Minutes)	Scheduling Method
CP:1	3	0.00	FCFS
CP:2	7	2.30	Johnson - Ascending
CP:7	2	2.00	Johnson - Ascending
CP:3	4	1.86	Johnson - Descending
CP:8	9	1.91	Johnson - Descending
CP:6	8	2.22	Johnson- Ascending
CP:4	5	2.19	Johnson- Ascending
CP:5	6	2.28	Johnson- Ascending

Production Orders Control:

(a)



(b)

Figure 6.23: Case study scheduling (a) Results via the PH-GUI – (b) Results via a gantt Chart.

When the robot was done with handling CP:7, the worker was still assembling CP:2, as shown in Figure 6.23b. Thus, the value of WMTV could be calculated in the same manner mentioned above as 2.0 min. As the value of WMTV equals to the value RTC, thus the remaining orders could be rescheduled based on Johnson's rules by ascending or descending the values of the required unit. In this case, rescheduling the remaining production orders was done based on ascending basis. Accordingly, CP:3 was scheduled next, as it contains the lowest number of units (i.e., 4 units).

Table 6.5: Case study scheduling through WMTV calculation.

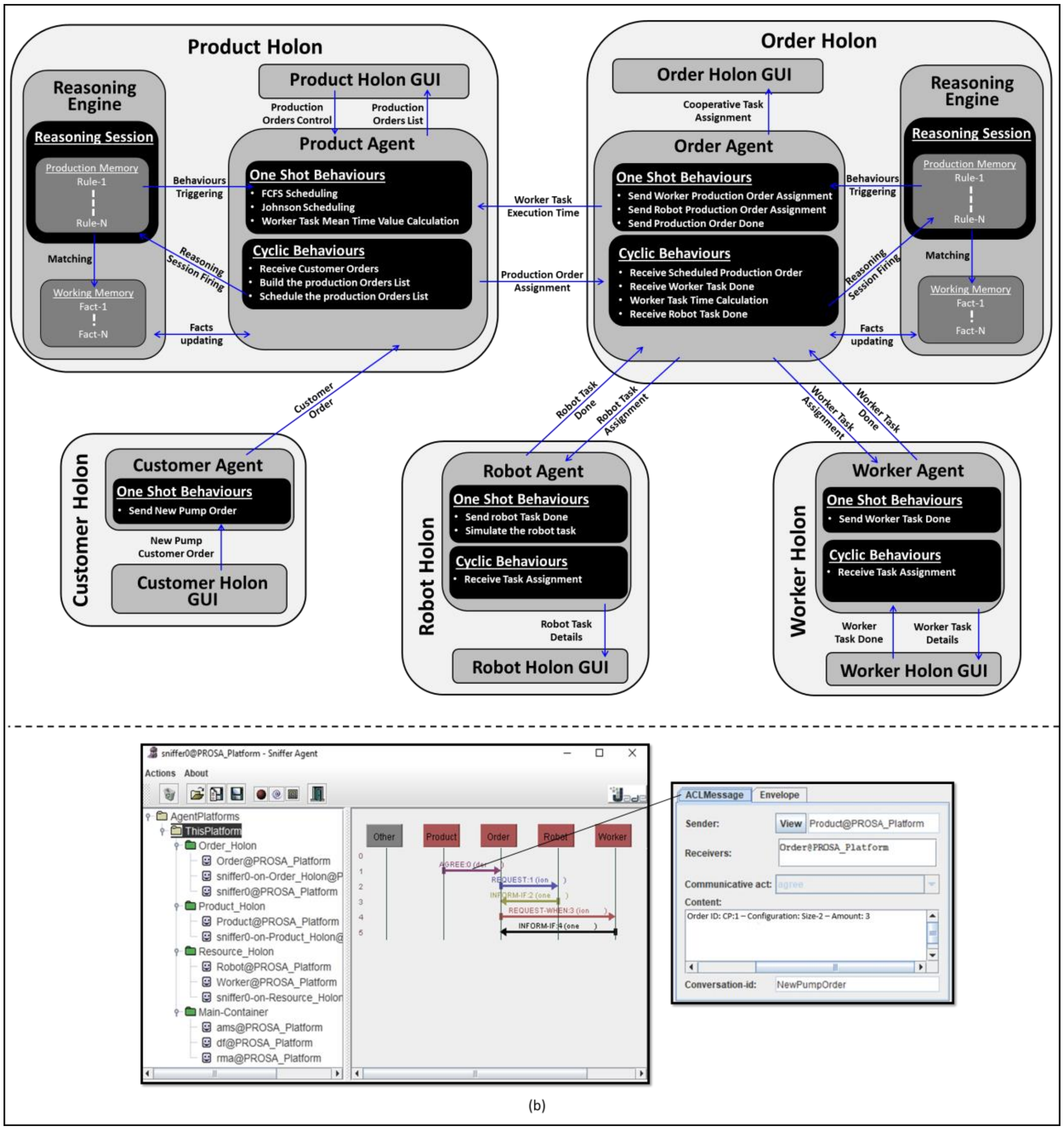
Order handled By the Cobot	Number of Assembled Units by the worker	WMTV (Minutes)	Next Order Scheduling Method	Next Scheduling Order
CP:1 (3 Units)	0	0.0	FCFS	CP:2 (7 Units)
CP:2 (7 Units)	3 + 3 = 6 Units	$\frac{20-6}{6} = 2.3$	Johnson (Ascending)	CP:7 (2 Units)
CP:7 (2 Units)	6 + 3 = 9 Units	$\frac{24-6}{9} = 2.0$	Johnson (Ascending)	CP:3 (4 Units)
CP:3 (4 Units)	10 + 2 + 2 = 14 Units	$\frac{32-6}{14} = 1.86$	Johnson (Descending)	CP:8 (9 Units)
CP:8 (9 Units)	16 + 7 = 23 Units	$\frac{50-6}{23} = 1.91$	Johnson (Descending)	CP:6 (8 Units)
CP:6 (8 Units)	25 + 2 = 27 Units	$\frac{66-6}{27} = 2.22$	Johnson (Ascending)	CP:4 (5 Units)
CP:4 (5 Units)	25 + 7 = 32 Units	$\frac{76-6}{32} = 2.19$	Johnson (Ascending)	CP:5 (6 Units)
CP:5 (6 Units)	33 + 3 = 36 Units	$\frac{88-6}{36} = 2.28$	Johnson (Ascending)	

When the robot was done with handling CP:3, the worker had done with assembling CP:2 and CP:7, as shown in Figure 6.23b. Thus, the value of WMTV could be calculated as 1.86 min. As the value of WMTV is less than RTC (i.e., 2.0 min). Therefore, the production orders were scheduled based on Johnson's rules by descending the values of the required unit, as explained in details in section 5.3.3. Accordingly, CP:8 was scheduled next, as it contains the highest number of units (i.e., 9 units). When the robot was done with handling CP:8, the value of WMTV was 1.91 min which is still less than RTC. Thus, rescheduling the remaining production orders was done on descending basis. Accordingly, CP:6 was scheduled next, as it contains the highest number of units (i.e., 8 units).

When the robot was done with handling CP:6, the value of WMTV was 2.22 min which is greater than RTC. Thus, rescheduling the remaining production orders was done on ascending basis. Accordingly, CP:4 was scheduled next, as it contains the lowest number of units (i.e., 5 units). Subsequently, CP:5 was scheduled after, as it is the last remaining production order.

6.4.3 Holons Interaction

Figure 6.24a shows the interaction among the case study holons. The CH implements a one-shot behaviour that sends the customer order in form of an Agree-Message to the PH. The PH implements a cyclic behaviour that receives the customer orders and transforms them into production orders list. When the PH receives a customer order, it schedules the existing production list either based on FCFS or Johnson's scheduling.



Sender: Product@PROSA_Platform
Receivers: Order@PROSA_Platform
Communicative act: agree
Content: Order ID: CP:1 – Configuration: Size-2 – Amount: 3
Conversation-id: NewPumpOrder

Figure 6.24: Case study holons interaction (a) Schematic drawing – (b) Product order assignment.

Scheduling the production occurs by firing Drools RE via a cyclic behaviour that accepts the customer order Agree-Message. The Drools RE contains the rules that define which scheduling method is used. At the very beginning, the PH uses FCFS scheduling because the value of the WMTV is still unknown. Therefore, the PH selects the first production order on the top of the list to be executed. This production order can be seen via the product holon GUI in Figure 6.22, as well as its assignment via an Agree-Message as shown in Figure 6.24b.

When the OH receives the production order, it assigns the order as a pick and place task to the robot and an assembly task to the worker. Because this case study does not involve a real robot hardware, it has been assumed that RTC of the robot to handle one pump is 2.0 minutes. Therefore, the RH simulates the overall time needed by the robot to handle the production order (i.e., RT_{Pi}). When RT_{Pi} is elapsed, the RH sends a task done event to the OH. While in the worker case, the worker holon GUI can be used to send a task done event by pressing task-done button. By this way, the variation in the WT_P can be simulated. When the OH receives the worker's task done event, it can calculate the time taken from the worker to assemble one pump by dividing the task overall time over the number of required units. Then, it sends this time to the PH. Therefore, the PH is able to calculate the WMTV and schedule the production list based on Johnson's rules. Then, it sends the production order on the top of the scheduled list to be executed. Furthermore, the value of WMTV is continuously updated every time the worker finishes an assigned task.

Chapter 7 - Summary, Outcomes, and Outlook

"The scientific man does not aim at an immediate result. He does not expect that his advanced ideas will be ready taken up. His work is like that of the planter – for the future. His duty is to lay the foundation for those who are to come and point the way."

Nikola Tesla

7.1 Summary

This dissertation combines two important subjects of research, which are the cooperative manufacturing and the HCA. The link between the two subjects comes from the need to control the cooperative workcell to achieve the smart manufacturing goals. The dissertation shows the roles of the cobot and the worker within the cooperative workcell, and how they can complement each other. Therefore, the dissertation emphasises the idea to increase the productivity by providing the cobot to the worker as a smart tool. Simultaneously, the flexibility of the worker to use his intuitive experience and logical judgment is a big advantage in the cooperative workcell. Different well-known cobots were studied to understand their capabilities. The advantages of the cobot come in handy when automating repeatable, physically harsh, or hazard tasks. Some aspects of the cobot such as the maximum weight payload are still in progress, due to the novelty of the field. The selection of the cobot must consider its capabilities and limitations, in addition to the purpose and the application of the cobot within the cooperative workcell. The dissertation highlights that the research in safety in cooperative manufacturing is far advanced than in the cooperative manufacturing control system. The current research in cooperative manufacturing control focuses only on the implementation aspect, without considering a conceptual solution to build the implementation upon. Moreover, the current research does not consider the existing solutions in smart manufacturing systems such as the FMS and the RMS. The literature review of this dissertation shows that the HCA is the most convenient conceptual solution for similar problems of the cooperative workcell. The implementation of the HCA differs from one research to another, but it is generally conducted via IEC 61499 FB or autonomous software agent technologies.

The proposed HCA by this dissertation starts from the software component (i.e. the holon) design. A new perspective of the holon structure is introduced via this dissertation. This perspective divides the holon into three layers which are physical data and events, information communication, and knowledge reasoning. Each layer implements a different technology that serves its purpose. The selection of the implemented technologies within every layer is based on the literature review. Other technologies such as the industrial web services could be implemented as the holon's communication layer as well. However, the dissertation aims at providing a software model as a guideline to be followed by the other researchers. Therefore, the implemented technologies within this software model can be alternated based on the designer's selection and the case study. The holon software model is developed to be generic, therefore the next step was to define an object-oriented model of the holon based on its purpose within the cooperative workcell. According to the holonic concept, three basic holon models which are the ORH, the PH, and the OH are defined to build the cooperative workcell HCA. Furthermore, in case of expanding the solution over a large-scale enterprise, the SH is added to the proposed HCA. The holon object-oriented models contain all the necessary attributes and describe the interaction mechanism among them, which is the main guide to code these holons during the implementation phase. Accordingly, the implementation chapter introduces three different cooperative workcell cases. Each case focuses on screening a specific aspect of the proposed HCA.

The first cooperative case study considers the cooperation between one worker and one dual-arm cobot. The essential focus of the case study is to proof the feasibility of the proposed concept, by implementing the solution over the automation devices. From the hardware perspective, Baxter rethink robot represents the cobot. Baxter runs by ROS, which is practical and flexible middleware to control the cobot. Furthermore, in order to achieve the physical interaction, a hand gesture sensor is used by the worker during the cooperative scenario. Leap Motion sensor is chosen for this purpose based on studying the different available gesture recognition techniques. The selection of the Leap sensor is also built on showing its advances in comparison with the Microsoft Kinect, which is commonly used as gestures recognition technique. From the software perspective, IEC 61499 FB technology represented the physical interface with the worker I/O, while ROS technology represented the physical interface with the cobot I/O. IEC 61499 FB technology guarantees the compatibility with many industrial controllers such as PACs, while ROS is widely spread among the cobot manufacturers as a hardware control technology. The information exchange between the worker and the cobot is accomplished via JADE agents. The reason to select IEC 61499 FB or ROS as the holon's physical layer, is the feasibility of these technologies to be deployed over the distributed hardware. Furthermore, IEC 61499 FB and ROS have simple data exchange pattern that guarantees the speed of the physical signal exchange. In the same time, the simplicity of IEC 61499 FB and ROS to exchange the data represents a limitation to implement any of them as the holon's communication layer. This is the reason to select JADE agents to implement sophisticated communication patterns.

The second case study focuses on the knowledge representation within the cooperative workcell. The cooperative manufacturing knowledge includes the robot, the worker, and other production components such as the product itself. For this reason, an ontology-based model is developed to represent the case study. The case study composes of two workers in cooperation with one robot. Two customized products can be manufactured in this case study, the first is a centrifugal pump and the second is a screw compressor. The task of the workers is to assemble the customized production order. While the task of the robot is to handle the right amount of the production parts to the workers based on their status. The case study illustrates in details the idea of using a conceptual ontology model to represent and share the common domain knowledge. Furthermore, Drools RE is added as the third layer of the holon, to reason the exchanged knowledge. Drools RE is used within this case study to infer the required manufacturing knowledge from the customer orders, then it utilizes this knowledge to assign the cooperative tasks. Additionally, Drools RE monitors some of the production KPIs. Those KPIs are very important while taking a task assignment decision under ambiguous circumstances, when the two workers are in a free status.

The third case study extends the implementation of Drools RE as the holon's reasoning layer, to dynamically schedule the cooperative workcell. The proposed cooperative workcell composes of one worker in cooperation with one robot. The goal of the cooperative workcell is to reschedule the production orders while operation, to minimize the overall makespan. Drools RE uses Johnson's method to schedule the cooperative workcell, science Johnson's scheduling is been proven to be more efficient than FCFS. The fact that Drools is coded via a set of rules, makes it so easy to transform Johnson's scheduling rules into a decision table that fits the case study. However, the problem is that Johnson's scheduling is mainly built over the condition of knowing the task T_P in advance. For the worker, this condition can be hardly achieved. Due to the human nature of the worker, the task T_P is continuously varying. The solution is that the HCA records the WT_P after every time the worker finishes a task. From these records, the solution is able to estimate the next WT_P, and hence rescheduling the cooperative workcell based on Johnson's method. Using Johnson's scheduling is an example of one efficient scheduling method that can be applied by the proposed HCA. Nevertheless, other scheduling techniques can be applied as well. The idea is to show how the proposed HCA can leverage the capabilities of the cooperative workcell.

Finally, in a large enterprise, the three case studies can be combined together. However, it is easier to explain the implementation by dividing it into three connected parts.

7.2 *Outcomes*

Due to the practical nature of this dissertation, its main outcomes are screened in various conceptual and implemental achievements. Those achievements are summarized as follows:

- **The novelty in the cooperative manufacturing idea:** this novelty has been explained by linking three different subjects, which are: smart manufacturing and industry 4.0 goals, the importance of the worker in manufacturing, and the evolution of industrial robotics. Therefore, those subjects shaped the nature the cooperative manufacturing problem and defined the dimensions of the dissertation.
- **The problem and the challenges:** the manufacturing control system that links between the worker and the cobot, is the main problem of this dissertation. Accordingly, new challenges beyond the worker's physical safety have been addressed. The answers to those challenges provided the guidelines to design the manufacturing control system of the cooperative workcell. Furthermore, considering the proposed challenges maximizes the use of the cobot to its full potential. A cobot is designed to be safe and cooperative, However, the absence of the proper control system would lead to only use the safety features of the cobot, and lose its value as a cooperative machine. The literature review showed the possibility of ensuring the safety of the conventional IR. However, a safe IR is not cooperative yet, until it is integrated from the information and communication point of view to the manufacturing system.
- **The cooperative manufacturing terminologies:** due to the novelty of the dissertation subject, it was crucial to define new terminologies that provide a common ground of understanding of the problem and the solution. It has been noted during the literature review that most of the researches are ignoring the difference between cooperative and collaborative manufacturing concept, by considering them the same. That leads to a misconception and inaccurate definition of the problem and the solution. For example, during the workcell scheduling problem, a huge difference in the nature of the solution would depend either on whether it is cooperative or collaborative. Furthermore, the new terminologies which are related to the cooperative manufacturing, have been created based on adapting and extending the similar terminologies from ISA-88 and ISA-95, which were originally created to support the field of industrial automation.
- **The literature review:** the literature review highlighted the gap between the cooperative workcell control system implementation model and conceptual model. Even that the cooperative workcell idea is novel in manufacturing, it is still an extension and a special case of the RMS or the FMS. The fact that the HCA has proved its ability to solve similar problems to the cooperative workcell, has never been considered by the cooperative manufacturing researchers. Instead, most of the researcher focused on the implementing aspect of the manufacturing control system based on the available technologies without

a conceptual approach. Dismissing the fact that the technologies are limited to specific cases and changes over the time. Sealing the gap between the cooperative manufacturing control system conceptual and implemental approach is one important achievement of this dissertation. Creating the cooperative manufacturing holonic concept provided a generic solution, which can be extended and modified by the other researchers during the future work of this subject.

- **The cooperative manufacturing holonic concept:** as mentioned earlier, the HCA is often used as a solution to the RMS and the FMS problems. Modifying the HCA reference models such as PROSA or ADACOR models, to fit the nature of the cooperative workcell, is a very important outcome of this dissertation. Thus, the dissertation went deep into the details of every holon within the cooperative workcell. This has been achieved by preciously defining the responsibilities of every holon and the interaction models among them. Ultimately, the dissertation defined a generic HCA which can be applied over a small-scale industrial enterprise by implementing the basic holons, or in a large-scale industrial enterprise where an SH will be added. Also, the dissertation has proposed a CH as a cloud application that provides a guidance for the customer while selecting and customizing the production order, then interact with the other holons in the architecture to manufacture the customer order during the real production time.
- **The holon implementation model:** during the implementation of the holon, a modified structural model has been followed. This model differentiates between the manufacturing data, information, and knowledge. Based on this difference, the holon implementation model has been developed to contain three layers, every layer deploys a specific technology that serves its function. The proposed model via the literature review which merges the IEC 61499 FB and the autonomous agent has been upgraded to contain an RE core. The RE is mainly responsible for processing and updating the manufacturing knowledge. Many technical considerations have been taken during developing the holon implementation model such as its modularity, interoperability and scalability. Those considerations guarantee the agility of the implementation model as a software component.
- **The cooperative workcell ontology model:** the dissertation provided an ontology model for the cooperative workcell, which supported the common understanding among the different entities of the workcell. This model can be reused, extended or modified to fit similar case studies. The selection of the ontology implementation method and language has been based on an intensive study of the ontology spectrum, which has been explained in section 4.2.2. This spectrum does not only provide a guidance for this dissertation, but also can be used by other researchers to determine the proper ontology technique to implement, based on each case study requirements.

- **Worker hand gesture recognition:** the dissertation proposed an argument whether the worker uses the hand gestures or the body gestures during the interaction with the manufacturing control system. A comparison between the two methods showed that the hand gesture recognition is more adequate in the context of cooperative manufacturing. The argument also considered that the worker does not have to take off his protection gloves during this interaction, which saves a lot of the manufacturing time. Based on this conclusion, two modes of interaction have been implemented. The first mode is an explicit interaction, where the worker is directly commanding the robot to perform an action, which is outside the context of the production sequence. The second mode is an implicit interaction mode, where the action of the robot is a complement to the previous worker's action, which is within the context of the production sequence.
- **Solution feasibility:** different frameworks have been used during the implementation phase such as FBDK, ROS, JADE, and Drools. These frameworks are very common and well-known in industry, which guarantee the capability of deploying the proposed manufacturing solution over the commercial automation devices and cobots.
- **Cooperative workcell KPIs:** KPIs are generic terms that describe the enterprise performance. KPIs are divided into four types that are production, quality, maintenance, and inventory. Thus, the production KPIs have been studied within the scope of the cooperative workcell, as the dissertation is focused on the cooperative manufacturing. Accordingly, the cooperative workcell KPIs have been implied from the original production KPIs formulas. Those KPIs were not only used to monitor the workcell performance, but also to take control decision during ambiguous situations.
- **Cooperative workcell scheduling:** the dissertation emphasised the effect of the cooperative workcell scheduling on its overall efficiency. Johnson's scheduling method was modified to fit the cooperative workcell case. Then it has been transformed into a set of rules, which have been implemented by Drools RE. As the WT_P is needed to be known in advance, a new technique to predict the WT_P based on the previous history was followed by the proposed HCA.

7.3 *Outlook*

Based on the outcomes that have been achieved from this dissertation, the following directions can be taken to extend the current status of this research:

- **Large-scale enterprise:** due to the limitations of the dissertation time and capabilities, the implementation phase only focused on the cooperative workcell. However, the solution concept must be applied over a real factory shop floor, where more than one SH are coordinating the cooperative workcells. In addition, the concluded formulas in section 5.4.3 of the KPIs can be applied to measure the cooperative enterprise performance.

- **Empirical results analysis:** the subject of the cooperative manufacturing is very practical. Therefore, it is very important to evaluate the solution by using data from a real factory shop floor. Most of the proposed HCA characteristics such as the robustness, the fault-tolerance, and the reliability can be analysed on the long operation terms. From this analysis, the HCA parameters and behaviours can be adjusted, as this adjustment is a function in the application as well.
- **Hybrid cooperative architecture:** the same proposed HCA can be used to control the workcell in three different modes. Those modes are fully manual, cooperative, and fully automatic. The switching from one mode to another must be depending on the production requirements. The hybrid cooperative HCA is a very suitable solution for SMEs, where the enterprise space is limited and very important. Thus, the hybrid architecture will fuse the privileges of three manufacturing modes in one workcell.
- **Cognitive reasoning:** the RE is the core of the holon. The dissertation has used Drools RE framework to implement the reasoning part of the holon, as Drools affords many privileges such as ReteOO algorithm and hybrid chain of the reasoning. That makes Drools fast and efficient from the software perspective, however recent REs such as Soar are applying the cognitive reasoning architecture. Soar follows a similar chain of reasoning to the human mind, therefore it can be used along with Drools to provide more reasoning capabilities. A very good use of Soar is to represent the PH, as it can provide different possibilities of product recipes based on the prior actions of the worker.
- **Augmented reality assistance:** AR is a booming field in nowadays technologies. AR has a great potential in manufacturing especially in assembly processes. Inasmuch as the possible assembly steps can be provided to the worker via the AR. An AR headset can be connected to the WH in order to virtually guide the worker with the next assigned task. Therefore, the AR headset can replace the GUI outputs that have been often used during the implementation phase of the dissertation.
- **Neural network and machine learning:** the neural network main function is to receive a set of inputs, and perform progressively complex calculation, and it uses the output to solve the problem. Neural network is often used in classification and thereby it can teach the machine its environment. Therefore, the RH can be implemented in from of a neural network that is able to learn from the manufacturing environment. Thus, the robot can recognise one worker from another, to adjust its behaviour based on this specific worker. Additionally, the robot might be able to differentiate between the different production components, and learn a product assembly plan by observing the worker and repeat the steps again.

Bibliography

[4di16] 4diac. *IEC 61499 Implementation for Distributed Devices of the Next Generation*. url: https://www.eclipse.org/4diac/ (Accessed 5 May. 2016).

[ABB16] ABB.com. IRB 14000 YuMi - Industrial Robots from ABB Robotics. url: https://new.abb.com/products/robotics/industrial-robots/yumi (Accessed 15 June. 2015).

[aca14] acatech. Securing the future of German manufacturing industry - Recommendations for implementing the strategic initiative INDUSTRIE 4.0 - Final report of the Industrie 4.0 Working Group. Technical report, 2014, url: https://www.acatech.de/publikationen/.

[ACZ+14] Giovanni Buizza Avanzini, Nicola Maria Ceriani, Andrea Maria Zanchettin, Paolo Rocco, Luca Bascetta. *Safety Control of Industrial Robots Based on a Distributed Distance Sensor*. IEEE Transactions on Control System Technology, 22(6), pages 2127-2140, 2014. doi: 10.1109/TCST.2014.2300696.

[Ajl15] Ajlan Al-Ajlan. *The Comparison between Forward and Backward Chaining*. International Journal of Machine Learning and Computing, 5(2), pages 106-113, 2015. doi: 10.7763/IJMLC.2015.V5.492.

[Alh97] I. M. Alharkan. *On merging sequencing and scheduling theory with genetic algorithms to solve stochastic job shops*. PhD Dissertation, University of Oklahoma, Oklahoma, USA, 1997. url: https://dl.acm.org/citation.cfm?id=269252.

[ass17] Assembly Magazine. *Lean Plant Layout*. url: https://www.assemblymag.com/articles/89823-lean-plant-layout (Accessed 25 Sept. 2017).

[Aut15] Automationworld.com. *ISA-95: Integrating Manufacturing's Future | Automation World*. url: https://www.automationworld.com/article/technologies/mes-mom/isa-95-integrating-manufacturings-future (Accessed 11 Nov. 2015).

[Bal15] S. Balakirsky. Ontology based action planning and verification for agile manufacturing. Robotics and Computer-Integrated Manufacturing, 33, pages 21-28, 2015. doi: https://doi.org/10.1016/j.rcim.2014.08.011.

[Bar15] Daniele Baratta. *Industrial Collaborative Robot Design: a Guideline for Future Design Activity*. In Proceedings of the 1st Workshop on Artificial Intelligence and Design (AIDE), 2015. url: http://ceur-ws.org/Vol-1473/paper1.pdf.

[BB10] Vicent Botti, Adriana Giret Boggino. *ANEMONA: A Multiagent Methodology for Holonic Manufacturing Systems.* Springer Series in Advanced Manufacturing, 2010, ISBN 13: 9781849967785.

[BC06] Radu F. Babiceanu, F. Frank Chen. *Development and Applications of Holonic Manufacturing Systems: A Survey*. Journal of Intelligent Manufacturing, 17(1), pages 111-131, 2006, doi: 10.1007/s10845-005-5516-y.

[BCG07] F. Bellifemine, G. Caire, D. Greenwood. *Developing Intelligent Agent Systems.* John Wiley & Sons Inc, 2007, ISBN 978-0-470-05747-6.

[BDP+16] J. Barbosa, J. Dias, A. Pereira, P. Leitão. Engineering an ADACOR based solution into a small-scale production system. In IEEE 25th International Symposium on Industrial Electronics (ISIE), pages 28-33, 2016. doi: 10.1109/ISIE.2016.7744860.

[Bea12] Ryan A. Beasley. *Medical Robots: Current Systems and Research Directions*. Journal of Robotics, pages 1-14, 2012. doi: http://dx.doi.org/10.1155/2012/401613.

[BHS05] F. Baader, I. Horrocks, U. Sattler. *Description logics as ontology languages for the semantic web*. In Mechanizing Mathematical Reasoning, Springer Berlin Heidelberg, pages 228-248, 2005.

[BHW07] Rafael H. Bordini, Jomi Fred Hübner, Michael Wooldridge. *Programming Multi-Agent Systems in AgentSpeak using Jason.* John Wiley & Sons, 2007, ISBN 978-0-470-02900-8.

[Bit17] Bitzer.de. *Screw Compressors*. url: https://www.bitzer.de/gb/en/screw-compressors/ (Accessed 25 Oct. 2017).

[Bon14] Guillem Solé Bonet. *Robot workspace sensing and control with Leap Motion Sensor*. MSc Thesis, Lund University, Lund, Sweden, 2014. url: https://upcommons.upc.edu/bitstream/handle/2099.1/25190/Report.pdf.

[Bos05] Bosch Rexroth Cooperation. How to Optimize Your Assembly Operations – Designing an assembly system to fit the manufacturing process. Technical report, 2005, url: http://www13.boschrexroth-us.com/pdf/optimize%20your%20assembly.pdf.

[Bos15] Bosch Rexroth Cooperation. *APAS assistant inline*, url: https://www.bosch-apas.com/en/products-and-services/apas-assistant-inline/ (Accessed 17 March 2015).

[BPS17] M. Bdiwi, M. Pfeifer, A. Sterzing. A new strategy for ensuring human safety during various levels of interaction with industrial robots. CIRP Annals, 66(1), pages 453-456, 2017. doi: https://doi.org/10.1016/j.cirp.2017.04.009.

[BR17] J. Berg, G. Reinhart. *An Integrated Planning and Programming System for Human-Robot-Cooperation*. Procedia CIRP, pages 95-100, 2017. doi: https://doi.org/10.1016/j.procir.2017.03.318.

[Bus16] BusinessKorea. *Smart Factories Improving Productivity of SMEs*. url: http://www.businesskorea.co.kr/ (Accessed 25 April 2016).

[Bus98] S. Bussmann. *An Agent-Oriented Architecture For Holonic Manufacturing Control*. First Open Workshop, Proceedings of IMS98 – ESPRIT Workshop in Intelligent Manufacturing Systems, pages 1-12, 1998. url: http://stefan-bussmann.de/downloads/ims98.pdf.

[BV09] G. Black, V. Vyatkin. Intelligent Component-Based Automation of Baggage Handling Systems with IEC 61499. IEEE Transactions on Automation Science and Engineering, 7(2), pages 337-351, 2009. doi: 10.1109/TASE.2008.2007216.

[BWT+12] R. Buchner, D. Wurhofer, A. Weiss, M. Tscheligi. User Experience of a Smart Factory Robot: Assembly Line Workers Demand Adaptive Robots. *7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2012. doi: 10.1145/2157689.2157712.

[BWV+98] H. Brussel, J. Wyns, P. Valckenaers, L. Bongaerts, P. Peeters. Reference architecture for holonic manufacturing systems: PROSA. *Computers in Industry*, 37(3), pages 255-276, 1998. doi: https://doi.org/10.1016/S0166-3615(98)00102-X.

[CB06] J. Chouinard, R. Brennan. *Software for next generation automation and control*. In Industrial Informatics, 2006 IEEE International Conference on, pages 886-891, 2006, doi: 10.1109/INDIN.2006.275694.

[CC07] Camilo Christo, Carlos Cardeira. *Trends in Intelligent Manufacturing Systems*. In 2007 IEEE International Symposium on Industrial Electronics, pages 3209-3214, 2007. doi: 10.1109/ISIE.2007.4375129.

[CGA+14] Kim On Chin, Kim Soon Gan, Rayner Alfred, Patricia Anthony, and Dickson Lukose. *Agent Architecture: An Overview*. In Transactions On Science and Technology, 1(1), pages 18-35, 2014. url: https://www.researchgate.net/publication/275643980_Agent_Architecture_An_Overview.

[Chr94] J. Christensen. Holonic Manufacturing Systems: Initial Architecture and Standards Directions. *In Proceedings of the 1st European Conference on Holonic Manufacturing Systems*, pages 1-20, 1994. url: http://holobloc.com/papers/hannover.pdf.

[CLP03] O. Corcho, M. López, A.Pérez. *Methodologies, tools and languages for building ontologies: where is their meeting point?* In Data & Knowledge Engineering, 46 (1), pages 41 - 64, 2003. doi: 10.1016/S0169-023X(02)00195-7.

[DBW91] D. Dilts, N. Boyd, H. Whorms. The evolution of control architectures for automated manufacturing systems. Technical Journal of Manufacturing Systems, 10(1), pages 79–93, 1991. doi: 10.1016/0278-6125(91)90049-8.

[Des17] Designs CAD. *Centrifugal Pump DWG Block for AutoCAD*. url: https://designscad.com/downloads/centrifugal-pump-dwg-block-for-autocad-2/ (Accessed 22 Oct. 2017).

[Doc17] Docs.jboss.org. Drools Expert User Guide. url: https://docs.jboss.org/drools/release/5.2.0.CR1/drools-expert-docs/html_single/ (Accessed 14 May 2017).

[Dro16] Drools. url: https://www.drools.org/ (Accessed 25 May 2016).

[DS12] E. Diaconescu, C. Spirleanu. *Communication solution for industrial control applications with multi-agents using OPC servers*. In International Conference on Applied and Theoretical Electricity (ICATE), pages 1-6, 2006, doi: 10.1109/ICATE.2012.6403431.

[Ec16] Ec.europa.eu. Statistics on small and medium-sized enterprises - Statistics Explained. url: http://ec.europa.eu/eurostat/statistics-explained/index.php/Statistics_on_small_and_medium-sized_enterprises (Accessed 27 Dec. 2016).

[Elm05] Hoda A. ElMaraghy. Flexible and reconfigurable manufacturing systems paradigms. *International Journal of Flexible Manufacturing Systems*, 17(4), pages 261-276, 2005. url: https://link.springer.com/article/10.1007/s10696-006-9028-7.

[Fan16] Fanuc.eu. Collaborative industrial robot FANUC CR-35iA. url: http://www.fanuc.eu/de/en/robots/robot-filter-page/collaborative-robots/collaborative-cr35ia (Accessed 22 Dec. 2016).

[FBS15] M. Faber, J. Bützler, C.M. Schlick. Human-robot cooperation in future production systems: analysis of requirements for designing an ergonomic work system. *In Procedia Manufacturing*, 3, pages 510-517, 2015. doi https://doi.org/10.1016/j.promfg.2015.07.215.

[FCG+15] S.R. Fiorini, J.L. Carbonera, P. Gonçalves, V.A. Jorge, V.F. Rey, T. Haidegger, , M. Abel, S.A. Redfield, S. Balakirsky, V. Ragavan, H. Li. Extensions to the core ontology for robotics and automation. Robotics and Computer-Integrated Manufacturing, 33, pages 3-11, 2015. doi: https://doi.org/10.1016/j.rcim.2014.08.004.

[Fen02] D. Fensel. Ontology-based knowledge management. *IEEE Computer*, 35(11), pages 56-59, 2002. doi: 10.1109/MC.2002.1046975.

[Fip15] Fipa.org. Welcome to the Foundation for Intelligent Physical Agents. url: http://www.fipa.org/ (Accessed 15 Oct. 2015).

[Fra16] Fraunhofer IAO. Lightweight Robots in Manual Assembly – Best to Start Simply. Technical report, 2016, url: https://www.produktionsmanagement.iao.fraunhofer.de/content/dam/produktionsmanagement/de/documents/LBR/Studie-Leichtbauroboter-Fraunhofer-IAO-2016-EN.pdf.

[Fra17] Franka.de. FRANKA EMIKA. url: https://www.franka.de/ (Accessed 22 Jan. 2017).

[FVS+14] Marco Fontana, Rocco Vertechy, Simone Marcheschi, Fabio Salsedo, and Massimo Bergamasco. The Body Extender: A Full-Body Exoskeleton for the Transport and Handling of Heavy Loads. *IEEE Robotics & Automation Magazine*, 21(4), pages 34-44, 2014. doi: 10.1109/MRA.2014.2360287.

[GB04] A. Giret, V. Botti. *Holons and Agents*. Journal of Intelligent Manufacturing, 15(5), pages 645–659, 2004, doi: https://doi.org/10.1023/B:JIMS.0000.

[GGG+06] J.F. Guo, X.J. Gu, L.P. Cui, J. Ma, S.F. Wang, H.F. Zhan. An Implementation for Collaborative Manufacturing Platform Oriented to Distributed Environment. In Computer Supported Cooperative Work in Design (CSCWD'06), pages 1-6, 2006. doi: 10.1109/CSCWD.2006.253128.

[Gha10] Raji Ghawi. *Ontology-based Cooperation of Information Systems*. PhD dissertation, Université de Bourgogne, 2010. url: www.theses.fr/2010DIJOS003.pdf.

[GMM+13] A. García-Domínguez, M. Marcos-Bárcena, I. Medina-Bulo, and L. Prades-Martell. Towards an Integrated SOA-based Architecture for Interoperable and Responsive Manufacturing Systems. Procedia Engineering, 63(C), pages 123 - 132, 2013. doi: https://doi.org/10.1016/j.proeng.2013.08.268.

[GP15] Rajesri Govindaraju, Krisna Putra. A methodology for Manufacturing Execution Systems (MES) implementation. *In Proceedings of the IOP Conference Series: Materials Science and Engineering,* 114, pages 12–14, 2015. doi :10.1088/1757-899X/114/1/012094.

[Gre16] Tom Green. Universal Robots Strikes Again: Sells to BMW. Robotics Business Review.url: https://www.roboticsbusinessreview.com/manufacturing/universal_robots_strikes_again_sells_to_bmw/.

[Gru95] Thomas R.Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43 (5), pages 907-928, 1995. doi: https://doi.org/10.1006/ijhc.1995.1081.

[Grz16] W. Grzechca. *Manufacturing in Flow Shop and Assembly Line Structure*. International Journal of Materials, Mechanics, and Manufacturing, *4 (1), pages 25-30, 2016*. doi: *10.7763/IJMMM.2016.V4.219.*

[GS07a] Michael A. Goodrich, Alan C. Schultz. Human–robot interaction: a survey. *Foundations and Trends® in Human–Computer Interaction*, 1, pages 203-275, 2007. doi: 10.1561/1100000005.

[GS07b] E. German, L. Sheremetov. Specifying Interaction Space Components in a FIPA-ACL Interaction Framework. *In International Workshop on Languages, Methodologies and Development Tools for Multi-Agent Systems*, Springer, Berlin, Heidelberg. pages 191-208, 2007. doi:10.1007/978-3-540-85058-8_12.

[GS17] M.T. Gonçalves, K. Salonitis. Lean assessment tool for workstation design of assembly lines. *Procedia CIRP*, 60, pages 386-391, 2017. url: https://www.sciencedirect.com/science/article/pii/S2212827117300665.

[GTA14] GTAI. INDUSTRIE 4.0 - Smart Manufacturing for the Future. Technical report, William McDougall Marketing & communication - Germany Trade & Invest, 2014, url: https://www.manufacturing-policy.eng.cam.ac.uk/documents-folder/policies/germany-industrie-4-0-smart-manufacturing-for-the-future-gtai/view

[GW06] D. Goldin, P. Wegner. Principles of interactive computation. *In Interactive computation: The new paradigm,* Springer Science & Business Media. pages 25-37, 2006. doi: 10.1007/3-540-34874-3.

[HC01] H. Heikki, S. Campadello. Providing Messaging Interoperability in FIPA Communication Architecture. *In IFIP International Conference on Distributed Applications and Interoperable Systems*, Springer, Pages 121-126, 2001. url: https://link.springer.com/content/pdf/10.1007/0-306-47005-5_10.pdf.

[HDW+17] W. Huang, W. Dai, P. Wang, V. Vyatkin. *Real-time data acquisition support for IEC 61499 based industrial cyber-physical systems*. In Industrial Electronics Society, IECON 2017-43rd Annual Conference of, pages 6689-6694, 2017. doi: 10.1109/IECON.2017.8217168.

[HHZ+08] I. Hegny, O. Hummer, A. Zoitl, G. Koppensteiner, M. Merdan. *Integrating software agents and IEC 61499 real time control for reconfigurable distributed manufacturing systems*. In 2008 International Symposium on Industrial Embedded Systems, pages 249-252, 2008. doi: 10.1109/SIES.2008.4577710.

[HKW+11] S. Hu, J. Ko, L. Weyand, H. ElMaraghy, T. Lien, Y. Koren, H. Bley, G. Chryssolouris, N. Nasr, M. Shpitalni. Assembly system design and operations for product variety. *CIRP Annals-Manufacturing Technology,* 6 (20), pages 715-733, 2011. url: https://pdfs.semanticscholar.org/a00a/5d891db36469914309cdfa3fce60f42bf774.pdf.

[HLB+10] J. Honkola, H. Laine, R. Brown, O. Tyrkkö. *Smart-M3 information sharing platform*. In Computers and Communications (ISCC) - 2010 IEEE Symposium, pages 1041-1046, 2010. doi: 10.1109/ISCC.2010.5546642.

[Hol16] Holobloc Inc. *FBDK - The Function Block Development Kit.* url: http://www.holobloc.com/doc/fbdk/ (Accessed 5 May. 2016).

[HPF+16] E.G. Hernández-Martínez and Erika S. Puga-Velazquez and Sergio A. Foyo-Valdés and J.A. Meda Campaña. *Task-based Coordination of Flexible Manufacturing Cells using Petri Nets and ISA standards*. 8th IFAC Conference on Manufacturing Modelling, Management and Control MIM 2016, 49(12), pages 1008-1013, 2016.

[Hu13] S. Hu. Evolving Paradigms of Manufacturing: From Mass Production to Mass Customization and Personalization. *Procedia CIRP,* 7, pages 3-8, 2013. doi: https://doi.org/10.1016/j.procir.2013.05.002.

[IBa17] IBaset Conrad Leiva - VP Product Strategy and Alliances. On the Journey to a Smart Manufacturing Revolution. IndustryWeek. url: http://www.industryweek.com/systems-integration/journey-smart-manufacturing-revolution?page=2 (Accessed 12 June 2017).

[IEC16] IEC61499.*The new Standard In Automation*. url: http://www.iec61499.de/ (Accessed 5 May. 2016).

[Inf17] Infogrid.org. InfoGrid Web Graph Database. url: http://infogrid.org/trac/wiki/Reference/PidcockArticle (Accessed 10 May. 2017).

[ISA88] ISA-The Instrumentation, System, and Automation Society. ANSI/ISA-88.01 - Batch Control – Part 1: Models and Terminology, 1995, url: http://www.gmpua.com/GAMP/ISA-88.pdf.

[ISA95] ISA-The Instrumentation, System, and Automation Society. ANSI/ISA-95.00.01- Enterprise-Control System Integration – Part 1: Models and Terminology, 2008,

[ISO14] ISO - The International Organization for Standardization. ISO 22400-2 - Automation systems and integration - Key performance indicators (KPIs) for manufacturing operations management - Part 2: Definitions and descriptions, 2014, url: https://www.iso.org/obp/ui/#iso:std:iso:22400:-2:ed-1:v1:en.

[JAD15] JADE.tilab.com. JADE Site | Java Agent DEvelopment Framework. url: http://JADE.tilab.com/ (Accessed 10 Oct. 2015).

[JBU06] Charlotta Johnsson, Dennis Brandl, Keith Unger. ISA 95 for Beginners. White Paper, 2015. url: http://www.control.lth.se/media/Education/EngineeringProgram/FRTN20/2015/ISA95%20White%20Paper%20-%20beginners.pdf.

[Joh14] C. Johnsson. Key Performance Indicators Used as Measurement Parameter for Plant-Wide Feedback Loops. *IFIP International Conference on Advances in Production Management Systems (APMS),* pages 91-99, 2014.

[JW98] Nicholas R. Jennings, Michael Wooldridge. *Agent technology: foundations, applications, and markets.* Springer, 1998, ISBN 3-540-63591-2.

[Kan16] Jason O'Kane. A Gentle Introduction to ROS. University of South Carolina, 2016, ISBN 978-14-92143-23-9.

[KB13] K. Kruger, A. Basson. Multi-agent Systems vs IEC 61499 for Holonic Resource Control in Reconfigurable Systems. Forty Sixth CIRP Conference on Manufacturing Systems, 7, pages 503-508, 2013. doi: https://doi.org/10.1016/j.procir.2013.06.023.

[KB18] K. Kruger, A. Basson. JADE Multi-Agent System Holonic Control Implementation for a Manufacturing Cell, 2018. url: http://academic.sun.ac.za/mad/Papers/KrugerBasson_HolonicControlImplementationUsingJadeMAS_20180208.pdf.

[KJM+99] Y. Koren, U. Heisel, F. Jovane, T. Moriwaki, G. Pritschow, G. Ulsoy, H. Van Brussel. Reconfigurable Manufacturing Systems. *Reconfigurable Manufacturing Systems and Transformable Factories,* 48(2), pages 527–540, 1999. doi: https://dx.doi.org/10.1007/978-3-642-55776-7_19.

[Koe67] Arthur Koestler. *The Ghost in the Machine.* Hutchinson & Co. Ltd., London, 1967, ISBN 0-14-019192-5.

[Kor06] Y. Koren. *General RMS Characteristics. Comparison with Dedicated and Flexible Systems*. In Reconfigurable manufacturing systems and transformable factories, Springer Berlin Heidelberg, pages 27-45, 2006. url: https://link.springer.com/chapter/10.1007/3-540-29397-3_3.

[Küc04] G. Kück. *Tim Berners-Lee's Semantic Web*. In South African Journal of Information Management, 6 (1), 2004. url: https://www.sajim.co.za/index.php/SAJIM/article/download/297/288.

[KUK16] KUKA AG. LBR Iiwa | KUKA AG, 2017. url: https://www.kuka.com/en-de/products/robot-systems/industrial-robots/lbr-iiwa (Accessed 10 June 2016).

[Kum12] S. Kumar. *Agent-based semantic web service composition.* Springer Science, 2012, ISBN: 978-1-4614-4663-7.

[Kut07] Appu K. K. Kutta. *Robotics.* New Delhi: I.K. International Publishing House (Penguin Group), 2007, ISBN: 978-81-89866-38-9.

[KZL+16] N. Kang, C. Zhao, J. Li, J. A. Horst. A Hierarchical structure of key performance indicators for operation management and continuous improvement in production systems. *International Journal of Production Research, 54(21),* pages 6333-6350, 2016. url: https://doi.org/10.1080/00207543.2015.1136082.

[Lai07] Lien F. Lai. *A knowledge engineering approach to knowledge management*. Information Sciences, 19 (1), pages 4072-4094, 2007. url: https://doi.org/10.1016/j.ins.2007.02.028.

[Lea16] Leapmotion.com. Leap Motion. url: https://www.leapmotion.com/ (Accessed 23 May 2016).

[Lee97] Q. Lee. *Facilities and workplace design*. Engineering & Management Press, 1997. ISBN 0-89806-166-0.

[Lei04] P. Leitão. *An Agile and Adaptive Holonic Architecture for Manufacturing Control*. PhD dissertation, University of Porto, 2004. url: https://repositorio-aberto.up.pt/bitstream/10216/10960/2/%20Texto%20integral.pdf.

[LFS17] A. Lasota, T. Fong, A. Shah. A Survey of Methods for Safe Human-Robot Interaction. In Foundations and Trends® in Robotics, 5(4), pages 261-349, 2017. doi: 10.1561/2300000052.

[Log17] Logicdesign.com. url: https://www.logicdesign.com/industrial_robotics.php (Accessed 28 April 2017).

[LR08] Paulo Jorge Pinto Leitão, Francisco J. Restivo. Implementation of a Holonic Control System in a Flexible Manufacturing System. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* 38(5), pages 699 - 709, 2008. doi: 10.1109/TSMCC.2008.923881.

[LRS14] A. Lasota, F. Rossano, A. Shah. Safe Close-Proximity Human-Robot Interaction with Standard Industrial Robots. In proceedings of the IEEE International Conference on Automation Science and Engineering (CASE), pages 339-344, 2014. doi: 10.1109/CoASE.2014.6899348.

[LRT12] P. Leitão, N. Rodrigues, C. Turrin, A. Pereira, P. Petrali. GRACE ontology integrating process and quality control. In IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society, pages 4348-4353, 2012. doi: 10.1109/IECON.2012.6389189.

[LTL+94] S. Lozano, J. Teba, J. Larrañeta, L. Onieva, P. Álvarez. Dynamic part-routing in a Flexible Manufacturing. *Journal of Operations Reaserch, Statistics and Computer Science,* 34(4), pages 16-28, 1994. url: https://personal.us.es/jteba/FMS_Jorbel_PDF_Word.pdf.

[LW17] H. Liu, L. Wang. Human motion prediction for human-robot collaboration. Journal of Manufacturing Systems, 44(2), pages 287-294, 2017. doi: https://doi.org/10.1016/j.jmsy.2017.04.009.

[Mat14] Björn Matthias. *Industrial Safety Requirements for Collaborative Robots and Applications*. Workspace Safety in Industrial Robotics: trends, integration and standards - ERF 2014, 2014. url: https://www.roboticsbusinessreview.com/wp-content/uploads/2016/05/Industrial_HRC_-_ERF2014.pdf.

[McK16] McKinsey&Company. Where machines could replace humans and where they can't (yet). url:https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/where-machines-could-replace-humans-and-where-they-cant-yet (Accessed 10 Nov. 2016).

[McK17a] McKinsey&Company. A future the works: AI, Automation, Employment, and Productivity. Research Report, 2017. url: https://www.jbs.cam.ac.uk/fileadmin/user_upload/research/centres/risk/downloads/170622-slides-manyika.pdf.

[McK17b] Ellen McKewen. What is Smart Manufacturing? (Part 1A). Cmtc.com. url: https://www.cmtc.com/blog/what-is-smart-manufacturing-part-1a-of-6 (Accessed 10 June 2017).

[MDA+17] S.A. Matthaiakis, K. Dimoulas, A. Athanasatos, K. Mparis, G. Dimitrakopoulos, C. Gkournelos, A. Papavasileiou, N. Fousekis, S. Papanastasiou, G. Michalos. Flexible programming tool enabling synergy between human and robot. Procedia Manufacturing, 11, pages 431-440, 2017. doi: https://doi.org/10.1016/j.promfg.2017.07.131.

[MDJ+16] P. Muller, S. Devnani, J. Julius, D. Gagliardi and C. Marzocchi. Annual Report on European SMEs 2015/ 2016. Technical Report, 2016. url: https://ec.europa.eu/jrc/sites/jrcsh/files/annual_report_-_eu_smes_2015-16.pdf.

[Mec15] Mech.kuleuven.be. An overview of PROSA reference architecture. url: https://www.mech.kuleuven.be/en /pma/research/MACC/research/prosaresearch/Overview (Accessed 23 August 2015).

[MF13] Aaron Martinez, Enrique Fernández. Learning ROS for Robotics Programming. PACKT Publishing Ltd, 2013, ISBN 978-1-78216-144-8.

[Mic16] Microsoft.com. *Kinect Sensor*. url: https://msdn.microsoft.com/en-us/library/hh438998.aspx (Accessed 21 Oct. 2016).

[MIT14] MIT. Breakthrough Factories. Research Report, MIT Technology Review, 117(6), 2014. url: http://ilp.mit.edu/media/webpublications/pub/literature/tr-breports/14-12-manufacturing.pdf.

[MK16] António Moniz, Bettina-Johanna Krings. Robots Working with Humans or Humans Working with Robots? Searching for Social Dimensions in New Human-Robot Interaction in Industry. *Societies*, 6(3), 2016. doi: 10.3390/soc6030023.

[MMS+14] George Michalos, Sotiris Makris, Jason Spiliotopoulos, Ioannis Misios, Panagiota Tsarouchi, George Chryssolouris. ROBO-PARTNER: Seamless Human-Robot Cooperation for Intelligent, Flexible and Safe Operations in the Assembly Factories of the Future. *Procedia CIRP - 5th CATS 2014 - CIRP Conference on Assembly Technologies and System*, 23(C), pages 71-76, 2014. doi: 10.1016/j.procir.2014.10.079.

[MMT+15] G. Michalos, S. Makris, P. Tsarouchi, T. Guasch, D. Kontovrakis, G. Chryssolouris. *Design Considerations for Safe Human-robot Collaborative Workplaces*. Procedia CIRP, 37, pages 248-253, 2015. doi: https://doi.org/10.1016/j.procir.2015.08.014.

[MOE17] R. Meziane, M. Otis, H. Ezzaidi. Human-robot collaboration while sharing production activities in dynamic environment: SPADER system. Robotics and Computer-Integrated Manufacturing, 48(1), pages 243-253, 2017. doi: https://doi.org/10.1016/j.rcim.2017.04.010.

[Mon13] António Moniz. Robots and humans as co-workers? The human-centred perspective of work with autonomous systems. *IET Working Papers Series*, pages 1-21, 2013. url: http://hdl.handle.net/10362/10980.

[Mot02] B. Motik A. Maedche R. Volz. A Conceptual Modeling Approach for Building Semantics-Driven Enterprise Applications. *1st international Conference on Ontologies Databases and Application of Semantics*, pages 1082-1099, 2002. url: http://www.cs.ox.ac.uk/people/boris.motik/pubs/mmv02conceptual.pdf.

[MP07] M. Metzger, G. Polaków. Holonic multiagent-based system for distributed control of semi-industrial pilot plants. In Holonic and Multi-Agent Systems for Manufacturing, pages 338-347, Springer, Berlin, Heidelberg, 2007. doi: https://doi.org/10.1007/978-3-540-74481-8_32.

[MPM17] T. Meudt, M. Pohl, J. Metternich. Die Automatisierungspyramide - Ein Literaturüberblick. Technische Universität Darmstadt, 2017, url: http://tuprints.ulb.tu-darmstadt.de/6298/

[MS10] M. Morioka, S. Sakakibara. A new cell production assembly system with human–robot cooperation. *CIRP Annals - Manufacturing Technology*, 59(1), pages 9-12, 2010. url: https://doi.org/10.1016/j.cirp.2010.03.044.

[MTM+17] S. Makris, P. Tsarouchi, A.S. Matthaiakis, A. Athanasatos, X. Chatzigeorgiou, M. Stefos, K. Giavridis, S. Aivaliotis.. Dual arm robot in cooperation with humans for flexible assembly. CIRP Annals, 66 (1), pages 13-16, 2017. doi: https://doi.org/10.1016/j.cirp.2017.04.097.

[Nat16] National Instruments. PACs. url: http://www.ni.com/pac/ (Accessed 5 May. 2016).

[NFF+91] R. Neches, R. E. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, W. R. Swartout. Enabling Technology for Knowledge Sharing. *Al Mag.*, 12 (3), pages 36–56, 1991. url: https://doi.org/10.1609/aimag.v12i3.902.

[NM01] N. Noy, D. McGuinness. Ontology Development 101: A Guide to Creat Your First Ontology. *tech. rep.*, 2001. url: https://protege.stanford.edu/publications/ontology_development/ontology101.pdf.

[NMP+00] A. Neely, J. Mills, K. Platts., H. Richards, M. Gregory, M. Bourne, and M. Kennerley. Performance measurement system design: developing and testing a process-based approach. *International journal of operations & production management*, 20(10), pages 1119-1145, 2000. doi: https://doi.org/10.1108/01443570010343708.

[NNK10] H.T. Nejad, S. Nobuhiro, I. Koji. *Multi agent and holonic manufacturing control*, in Future Manufacturing Systems, pages 95-122, intechopen, 2010, doi: 10.5772/10240.

[Nof99] Shimon Y. Nof. *Handbook of industrial robotics.* New York: Wiley, 1999, ISBN: 0-471-17783-0.

[Nok17] Nokia.com. url: https://www.nokia.com (Accessed 6 Sept. 2017).

[NPW+14] M. Nowicki, O. Pilarczyk, J. Wąsikowski, K. Zjawin and W. Jaśkowski. Gesture recognition library for leap motion controller. BSc Thesis, Poznan University of Technology - Faculty of Computing - Institute of Computing Science, Poland, 2014. url: http://www.cs.put.poznan.pl/wjaskowski/pub/theses/LeapGesture_BScThesis.pdf.

[NSR11] P. T. Nguyen, V. Schau, W. Rossak. Performance Comparison of some Message Transport Protocol Implementations for Agent Community Communication. *In Proceedings of the Innovative Internet Computing System - Second International Workshop, IICS*, pages 193-204, 2001. url: http://swt.informatik.uni-jena.de/swt_multimedia/SWT/PDFs/PerformanceMTP-p-16200.pdf.

[nxt16] nxtControl. *nxtStudio.* url: http://www.nxtcontrol.com/en/ (Accessed 5 May. 2016).

[Obr13] Leo Obrst. Ontologies for semantically interoperable systems. *CIKM '03 Proceedings of the twelfth international conference on Information and knowledge management*, pages 03 - 08, 2013. doi: 10.1145/956863.956932.

[OEO+16] Armando Ordóñez, Luis Eraso, Hugo Ordóñez, Luis Merchan. Comparing Drools and Ontology Reasoning Approaches for Automated Monitoring in Telecommunication Processes. *Procedia Computer Science,* 95(C), pages 353 - 360, 2016. doi: https://doi.org/10.1016/j.procs.2016.09.345.

[Ora16] Oracle.com. What is ERP | Oracle. url: https://www.oracle.com/applications/erp/what-is-erp.html (Accessed 22 September 2016).

[ORW+11] Marianna Obrist, Wolfgang Reitberger, Daniela Wurhofer, Florian Förster, Manfred Tscheligi. User Experience Research in the Semiconductor Factory: A Contradiction. *Human-Computer Interaction 2011*, pages 144-151, 2011. doi: https://doi.org/10.1007/978-3-642-23768-3_12.

[Păt15] Aurelia Pătraşcu. *Comparative Analysis between OWL Modelling and UML Modelling*. Petroleum-Gas University of Ploiesti Bulletin, Technical Series, 67(2), 2015. url: http://www.upg-bulletin-se.ro/archive/2015-2/9.Patrascu.pdf.

[PD98] Lynne E. Parker, John V. Draper. *Robotics Applications in Maintenance and Repair Handbook of Industrial Robotics*. 2nd Edition, J. Wiley, 1998. url: http://web.eecs.utk.edu/~leparker/ publications/Handbook99.pdf.

[PD98] Lynne E. Parker, John V. Draper. *Robotics Applications in Maintenance and Repair Handbook of Industrial Robotics*. 2nd Edition, J. Wiley, 1998. url: http://web.eecs.utk.edu/~leparker/ publications/Handbook99.pdf.

[Pes14] Michael Peschl. *An architecture for flexible manufacturing systems based on task-driven agents*. PhD dissertation, Faculty of Information Technology and Electrical Engineering, Oulu University, Finland, 2014. url: http://jultika.oulu.fi/files/isbn9789526203669.pdf.

[Pho16] Phoenix Contact. RAMI 4.0 and IIRA reference architecture models - A question of perspective and focus. Technical report, 2016. url: https://www.mynewsdesk.com/material/document/56241/download?resource_type=resource_document.

[Pil17] Pilz.com. *Safe Camera System – Safety Eye*. url: https://www.pilz.com/en-INT/eshop/00106002207042/SafetyEYE-Safe-camera-system (Accessed 15 Dec. 2017).

[Pin99] Steven Pinker. *How the Mind Works.* Annals of the New York Academy of Sciences. J. Wiley, pages 119-127, 1998, ISBN: 0393045358.

[PMP+16] S. Pellegrinelli, F. Moro, N. Pedrocchi, L. Tosatti, T. Tolio. A probabilistic approach to workspace sharing for human–robot cooperation in assembly tasks. *In CIRP Annals,* 65(1), pages 57-60, 2016. doi: https://doi.org/10.1016/j.cirp.2016.04.035.

[PTL+15] J. Pauchot, L. D. Tommaso, A. Lounis, M. Benassarou, P. Mathieu, D. Bernot and S. Aubry. Leap Motion Gesture Control With Carestream Software in the Operating Room to Control Imaging. *In Surgical Innovation,* 22(6), pages 615-620, 2015. doi: 10.1177/1553350615587992.

[Ret16] Rethink Robotics. Baxter Collaborative Robots for Industrial Automation | Rethink Robotics. url: http://www.rethinkrobotics.com/baxter/ (Accessed 15 June 2016).

[ria17] Robotic Industry Association. Robotics Industry Insights. url: https://www.robotics.org/content-detail.cfm/Industrial-Robotics-Industry-Insights/The-Business-of-Automation-Betting-on-Robots/content_id/6076 (Accessed 15 Jan. 2017).

[RN16] Stuart J, Russell., Peter, Norvig. *Artificial intelligence: a modern approach.* Malaysia - Pearson Education Limited, 2016, ISBN 978-1-292-15396-4.

[Rod12] Nelson Rodrigues. *Development of an ontology for a multi-agent system controlling a production line*. MSc Thesis, Instituto Politécnico de Bragança, Bragança, Portugal, 2012.

[Ros16] Ros.org. ROS.org | Powering the world's robots. url: http://www.ros.org/ (Accessed 28 Feb. 2016).

[RPL13] N. Rodrigues, A. Pereira, P. Leitão. Adaptive multi-agent system for a washing machine production line. In International Conference on Industrial Applications of Holonic and Multi-Agent Systems, pages 212-223, Springer, Berlin, Heidelberg, 2013. doi: https://doi.org/10.1007/978-3-642-40090-2_19.

[RS11] A. Ricci, A. Santi. A Programming Paradigm based on Agent-Oriented Abstractions. *International Journal on Advances in Software,* 5(2), pages 36-52, 2011.

[Sal01] G. Salvendy. *Handbook of Industrial Engineering: Technology and Operations Management, Third Edition.* John Wiley & Sons, Inc., 2001, ISBN 0-471-33057-4.

[SAP+17] Z. Salcic, U.D. Atmojo, H.J. Park, A.T. Chen, I. Kevin, K. Wang. Designing Dynamic and Collaborative Automation and Robotics Software Systems. IEEE Transactions on Industrial Informatics, pages 1-10, 2017. doi: 10.1109/TII.2017.2786280.

[SCG09] Ja-Young Sung, Henrik I. Christensen, Rebecca E. Grinter. Robots in the Wild: Understanding Long-term Use. *HRI '09 Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 45-52, 2009. doi: 10.1145/1514095.1514106.

[sch16] schunk.com. url: https://schunk.com/de_en/gripping-systems/series/lwa-4p/ (Accessed 21 Sept. 2016).

[Sew16] Sew-eurodrive.de. Industry 4.0 - Lean Smart Factory | SEW-EURODRIVE. url: https://www.sew-eurodrive.de/hannovermesse/the_promise_of_possible/trends_and_innovations/industry_40__lean_smart_factory/industry_40_-_lean_smart_factory.html (Accessed 19 August 2016).

[Shi06] H. K. Shivanand. Teahan. *Flexible manufacturing system*. New Age International Limited, 2006. ISBN 978-81-224-2559-8.

[SHY+06] Weiming Shen, Qi Hao ,Hyun Joong Yoona, Douglas H. Norrieb. Applications of agent-based systems in intelligent manufacturing: An updated review. *Advanced Engineering Informatics,* 20(4), pages 415-431, 2006. doi: https://doi.org/10.1016/j.aei.2006.05.004.

[SIK06] G.D. Serugendo, M.P. Irit, A. Karageorgos. Self-organisation and emergence in MAS: An overview. Informatica, 30(1), 2006. url: http://www.informatica.si/ojs-2.4.3/index.php/informatica/article/view/72.

[SKM+16] A. Smirnov, A. Kashevnik, S. Mikhailov, M. Mironov, M. Petrov. Ontology-based collaboration in multi-robot system: Approach and case study. In System of Systems Engineering Conference (SoSE), 2016. doi: 10.1109/SYSOSE.2016.7542945.

[SN99] W. Stevenson, P. Ness. *Study Guide for Use with Production/Operations Management.* McGraw-Hill: Boston, MA, USA, 1999, ISBN 978-0073377841.

[SSK13] Balkeshwar Singh, N. Sellappan, P. Kumaradhas. Evolution of Industrial Robots and their Applications. *International Journal of Emerging Technology and Advanced Engineering,* 3(5), pages 763-768, 2013. url: http://www.ijetae.com/files/Volume3Issue5/ IJETAE_0513_126.pdf.

[Su07] J. Su. *Component-based Intelligent Control Architecture for Reconfigurable Manufacturing System*. PhD thesis, Faculty of the Virginia Polytechnic Institute and State University, 2007. url: http://hdl.handle.net/10919/29980.

[Tea10] William J. Teahan. *Artificial Intelligence – Agent Behaviour*. 1st ed.; BookBoon: London, UK, 2010. ISBN 978-87-7681-559-2.

[TLP+03] D. Trindade, P. Leal, P. Peças, E. Henriques. *Lean manufacturing application to an automotive assembly line*. Proceedings of the Business Excellence I: Performance Measures, Bechmarking and Best Practices in New Economy.; pages 590-595, 2003.

[TMC16] P. Tsarouchi, S. Makris, G. Chryssolouris. *On a Human and Dual-arm Robot Task Planning Method*. Procedia CIRP, 57, pages 551-555, 2016. doi: https://doi.org/10.1016/j.procir.2016.11.095.

[TMM+15] P. Tsarouchi, S. Makris, G. Michalos, A.S. Matthaiakis, X. Chatzigeorgiou, A. Athanasatos, M. Stefos, P. Aivaliotis, G. Chryssolouris.. *ROS based coordination of human robot cooperative assembly tasks-An industrial case study*. Procedia CIRP, 37, pages 254-259, 2015. doi: https://doi.org/10.1016/j.procir.2015.08.045.

[TMM+17a] P. Tsarouchi, A.S. Matthaiakis, S. Makris, G. Chryssolouris. D. Kontovrakis, G. Chryssolouris. *On a human-robot collaboration in an assembly cell*. International Journal of Computer Integrated Manufacturing, 30 (6), pages 580-589, 2017. doi: https://doi.org/10.1080/0951192X.2016.1187297.

[TMM+17b] P. Tsarouchi, G. Michalos, S. Makris, T. Athanasatos, K. Dimoulas, G. Chryssolouris. *On a human–robot workplace design and task allocation system*. International Journal of Computer Integrated Manufacturing, 30 (12), pages 1272-1279, 2017. doi: https://doi.org/10.1080/0951192X.2017.1307524.

[Tmr16] Tm-robot.com. Techman Robot. url: http://tm-robot.com/ (Accessed 08 June 2016).

[TT98] A. De Toni, S. Tonchia. Manufacturing flexibility: A literature review. International Journal of Production Research, 36(6), pages 587-1617, 1998. doi: https://doi.org/10.1080/002075498193183.

[tut16] tutorialspoint.com. Drools. url: https://www.tutorialspoint.com/drools/drools_quick_guide.htm (Accessed 18 Dec 2016).

[TWZ+17] T. Terzimehic, M. Wenger, A. Zoitl, A. Bayha, K. Becker, T. Müller, H. Schauerte. Towards an industry 4.0 compliant control software architecture using IEC 61499 & OPC UA. *In the 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA),* pages 1–4, 2017. doi: 10.1109/ETFA.2017.8247718.

[ULT+18] V.V. Unhelkar, P.A. Lasota, Q. Tyroller, R.D. Buhai, L. Marceau, B. Deml, J.A. Shah. Human-Aware Robotic Assistant for Collaborative Assembly: Integrating Human Motion Prediction with Planning in Time. IEEE Robotics and Automation Letters, 3(3), pages 2394-2401, 2018. doi: https://doi.org/10.1016/j.cirp.2017.04.009.

[Uni16] Universal-Robots.Com. Collaborative Industrial Robotic Robot Arms | 6 Axis UR. url: https://www.universal-robots.com/ (Accessed 08 June 2016).

[VWE17] C. Vogel, C. Walter, N. Elkmann. Safeguarding and Supporting Future Human-robot Cooperative Manufacturing Processes by a Projection- and Camera-based Technology. *Procedia Manufacturing,* 11, pages 39 - 46, 2017. doi: https://doi.org/10.1016/j.promfg.2017.07.127.

[Vya10] V. Vyatkin. *Towards Using the Reconfiguration Capability of IEC 61499 Specifications for Modeling and Implementing Dynamic Holonic Interactions.* The 12th WSEAS International Conference on Automatic Control, Modelling & Simulation, pages 213-218, 2010, ISBN: 978-954-92600-1-4.

[Vya11] V. Vyatkin. *IEC 61499 as enabler of distributed and intelligent automation: State-of-the-art review*. IEEE Transactions on Industrial Informatics, 7(4), pages 768-781, 2011, doi: 10.1109/TII.2011.2166785.

[WAG+05] P. Wegner, F. Arbab, D. Goldin, P. McBurney, M. Luck, and D. Robertson. The role of agent interaction in models of computing: Panelist reviews. *Electronic Notes in Theoretical Computer Science*, 141(5), pages 181-198, 2005. doi: https://doi.org/10.1016/j.entcs.2005.05.022.

[WBR+13] F. Weichert, D. Bachmann, B. Rudak, and D. Fisseler. Analysis of the Accuracy and Robustness of the Leap Motion Controller. *In Sensors*, 13(5), pages 6380-6393, 2013. doi: 10.3390/s130506380.

[WGP+08] Hai H. Wang, Nick Gibbins, Terry Payne, Ahmed Saleh, Jun Sun. A Formal Model of Semantic Web Service Ontology (WSMO) Execution. *13th IEEE International Conference on Engineering of Complex Computer Systems (iceccs 2008)*, 2008. doi: 10.1109/ICECCS.2008.25.

[WH16] Astrid Weiss, Andreas Huber. User Experience of a Smart Factory Robot: Assembly Line Workers Demand Adaptive Robots. *5th International Symposium on New Frontiers in Human-Robot Interaction*, 2016. url: https://arxiv.org/abs/1606.03846.

[WHM16] Astrid Weiss, Andreas Huber, Jürgen Minichberger, Markus Ikeda. First Application of Robot Teaching in an Existing Industry 4.0 Environment: Does It Really Work?. *In Societies*, 6(3), 2016. url: http://www.mdpi.com/2075-4698/6/3/20.

[WHP13] Brian A. Weiss, John A. Horst, Frederick M. Proctor. Assessment of Real-Time Factory Performance through the Application of Multi-Relationship Evaluation Design. *In NIST Interagency/Internal Report (NISTIR) - 7911,* 2013. URL: https://nvlpubs.nist.gov/nistpubs/ir/2013/NIST.IR.7911.pdf.

[WJ95] Michael Wooldridge, Nicholas R. Jennings. *Intelligent Agents: Theory and Practice.* Knowledge Engineering Review, 10 (2), pages 115-152, 1995, url: http://www.cs.ox.ac.uk/people/michael.wooldridge/pubs/ker95.pdf.

[WJR90] J. Womack, D. Jones, D. Roos. *The machine that changed the world*. Rawson Associates, New York, 1990. ISBN 0-892.56-350-8.

[WKV+17] X.V. Wang, Z. Kemény, J. Váncza, L. Wang. Human–robot collaborative assembly in cyber-physical production: Classification framework and implementation. CIRP annals, 66 (1), pages 5-8, 2017. doi: https://doi.org/10.1016/j.cirp.2017.04.101.

[Wu13] Xichuan Wu. *Enhancing Ontology Matching using Logic-based Reasoning*. MSc Thesis, Technische Universität Dresden, Germany, 2013. url: https://lat.inf.tu-dresden.de/research/mas/Wu-Mas-13.pdf.

[WW14] R. Weidner, J. P. Wulfsberg. Concept and Exemplary Realization of Human Hybrid Robot for Supporting Manual Assembly Tasks. *In Procedia CIRP*, 23, pages 53-58, 2014. doi: https://doi.org/10.1016/j.procir.2014.10.096.

[You16a] YouTube. Light L16 Camera (the light camera story). 2016. url: https://www.youtube.com/watch?v=cW3rx7jiX8E (Accessed 18 May 2016).

[You16b] YouTube. Das Industrie 4.0 Modell auf der HANNOVER MESSE 2016. 2016. url: https://www.youtube.com/watch?v=A5ZL_PaY29Y (Accessed 18 May 2016).

[You16c] YouTube. Sensitives Fügen von Kegelrädern im Mensch-Roboter-Kollaboration-Betrieb (MRK). url: https://www.youtube.com/watch?v=OxNC8yvsZ6s (Accessed 18 May 2016).

[You17a] YouTube. How Boeing Builds a 737 Plane in Just 9 Days | WIRED. url: https://www.youtube.com/watch?v=liZ0WEEsuz4 (Accessed 10 Feb. 2017).

[You17b] YouTube. Assembling Android Tablet Factory Tour in China (This is how your tablet is made). url: https://www.youtube.com/watch?v=tbyu1tbGE38 (Accessed 12 Feb. 2017).

[You17c] YouTube. ABB Robotics - YuMi-ABB-ElektroPraga. url: https://www.youtube.com/watch?v=i9Vbh2mPG6M (Accessed 25 June 2017).

[ZLL+17] N. Zhou, D. Li, S. Li, S. Wang, C. Liu. *Model-Based Development of Knowledge-Driven Self-Reconfigurable Machine Control Systems*. IEEE Access, 5, pages 19909-19919, 2017, doi: 10.1109/ACCESS.2017.2754507.

[ZS08] X. Zhao, Y.J. Son. *Extended BDI framework for modelling human decision-making in complex automated manufacturing systems*. International Journal of Modelling and Simulation, 28 (3), pages 347-356, 2008. doi: https://doi.org/10.1080/02286203.2008.11442487.

List of Own Publications

[SB16a] Ahmed R. Sadik, Bodo Urban. A Holonic Control System Design for a Human & Industrial Robot Cooperative Workcell. *In Proceedings of the 2016 International Conference on Autonomous Robot Systems and Competitions (ICARSC-2016),* pages 118-123, 2016. doi: 10.1109/ICARSC.2016.16.

[SB16b] Ahmed R. Sadik, Bodo Urban. A Novel Implementation Approach for Resource Holons in Reconfigurable Product Manufacturing Cell. *In Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO-2016),* pages 130-139, 2016. doi: 10.5220/0005956801300139.

[SBA17] Ahmed R. Sadik, Bodo Urban, Omar Adel. Using Hand Gestures to Interact with an Industrial Robot in a Cooperative Flexible Manufacturing Scenario. *In Proceedings of the 3rd International Conference on Mechatronics and Robotics Engineering (ICMRE 2017),* pages 11-16, 2017. doi: 10.1145/3068796.3068801.

[SB17a] Ahmed R. Sadik, Bodo Urban. Applying the PROSA Reference Architecture to Enable the Interaction between the Worker and the Industrial Robot - Case Study: One Worker Interaction with a Dual-Arm Industrial Robot. *In Proceedings of the 9th International Conference on Agents and Artificial Intelligence (ICAART-2017),* pages 190-199, 2017. doi: 10.5220/0006191801900199.

[SB17b] Ahmed R. Sadik, Bodo Urban. Combining Adaptive Holonic Control and ISA-95 Architectures to Self-Organize the Interaction in a Worker-Industrial Robot Cooperative Workcell. *MDPI - Future Internet,* 9(3), 35, 2017. doi:10.3390/fi9030035.

[SB17c] Ahmed R. Sadik, Bodo Urban. Flow Shop Scheduling Problem and Solution in Cooperative Robotics—Case-Study: One Cobot in Cooperation with One Worker. *MDPI - Future Internet,* 9(3), 48, 2017. doi:10.3390/fi9030048.

[SB17d] Ahmed R. Sadik, Bodo Urban. An Ontology-Based Approach to Enable Knowledge Representation and Reasoning in Worker-Cobot Agile Manufacturing. *MDPI - Future Internet,* 9(4), 90, 2017. doi:10.3390/fi9040090.

[SB17e] Ahmed R. Sadik, Bodo Urban. Towards a Complex Interaction Scenario in Worker-Cobot Reconfigurable Collaborative Manufacturing via Reactive Agent Ontology - Case-Study: Two Workers in Cooperation with One Cobot. *In Proceedings of the 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KEOD-2017),* 2, pages 27-38, 2017. doi: 10.5220/0006487200270038.

[STB17] Ahmed R. Sadik, Andrei Taramov, Bodo Urban. Optimization of Tasks Scheduling in Cooperative Robotics Manufacturing via Johnson's Algorithm. *In Proceedings of 2017 IEEE Conference on Systems, Process and Control (ICSPC-2017),* pages 36-41, 2017. doi: 10.1109/SPC.2017.8313018.

[SB18a] Ahmed R. Sadik, Bodo Urban. CPROSA-Holarchy – An Enhanced PROSA Model to Enable the Worker-Cobot Agile Manufacturing. *In Proceedings of the 4th International Conference on Artificial Intelligence (ICOAI-2017) and the International Journal of Mechanical Engineering and Robotics Research,* 7(3), pages 296-304, 2018. doi: 10.18178/ijmerr.7.3.296-304.

[SB18b] Ahmed R. Sadik, Bodo Urban. Ontology in Holonic Cooperative Manufacturing: A Solution to Share and Exchange the Knowledge. *Lectures Notes in Computer Science – Springer,* 2018. (accepted)

[SBA18] Ahmed R. Sadik, Bodo Urban, Omar Adel. Holonic control architecture implementation in worker-robot cooperation: a case study, *Production Planning and Control,* 2018. (submitted)

Thesis Statement

- The rapid development in robotics technology led to a new generation of the industrial robot, which is commonly known as a **co**llaborative ro**bot** (cobot). Cobot is a social industrial robot that can cooperate safely with a human co-worker. This in contrast with the conventional industrial robot, which is dangerous to operate in a direct contact with the human being, therefore it is often isolated from the worker.
- A variety of cobots are available in the commercial industrial market, which guarantee the worker physical safety during the cooperation. However, the physical safety is not enough to achieve this cooperation. The manufacturing control system is the main tool to link the worker, the cobot, and the production components in one workcell.
- The cooperative workcell is distributed by nature, which requires an autonomous control solution to fulfil this nature. Therefore, the holonic control architecture has been proposed to control the cooperative manufacturing workcell.
- Many challenges in cooperative manufacturing can be solved via the holonic control architecture, those challenges are summarized as follows:
  - ➢ Worker data interpretation: the control data and events are generated from the worker during his cooperation with the cobot. The holonic control architecture is responsible for interpreting these inputs to useful information, and interpreting its outputs into the right format, which can be understood by the worker.
  - ➢ Workcell knowledge representation: the abstract representation of the objects within the workcell is an essential necessity to establish a successful cooperation. A proper approach is required to describe the shared environment between the worker and the cobot including themselves.
  - ➢ Workcell knowledge exchange: the holonic control architecture is the nervous system of the workcell, which provides a communication framework that connects the cooperative workcell components together. In the absence of this communication framework, the maximum usability of the cobot can never be achieved, inasmuch as it loses its real value as a smart tool.
  - ➢ Workcell self-organising: the main purpose of the cooperative workcell is to adapt with the variations in the production demands. This means that the holonic control architecture must comprehend these variations, and hence self-organize the workcell during the real-time of the production.
  - ➢ Workcell planning and scheduling: the holonic control architecture is responsible for planning the cooperative task between the worker and the cobot. This must be done in consideration of many variations such as the number and the status of the workers and the cobots. The holonic control architecture must also consider scheduling the cooperative tasks to obtain the maximum production efficiency.
  - ➢ Workcell integration into the industrial enterprise: the workcell is the smallest unit of the industrial enterprise. Thus, it is important to define an information model that horizontally connects all the cooperative workcells over the shop floor, and vertically connects the different layers of the industrial enterprise.

- There is a gap in the literature review between the cooperative manufacturing control concept and implementation model. This gap has been sealed during this thesis work.
- A control software component called a holon represented the building block of the holonic control architecture. A holon is composed of three layers which are physical data, information communication, and knowledge reasoning. Each layer implements a different technology that serves its purpose.
- The holonic control architecture composes of four holon categories which are, the operational holon that represent a physical entity within the workcell such as a worker or a cobot. The product holon represents the production plan required to manufacture a product. The order holon represents the production tasks. The supervisor holon provides coordination services to the other holons if it is required.
- The responsibilities and the interaction model among the previous four holons have been defined in the context of the cooperative workcell.
- The implementation of the holonic manufacturing architecture has been conducted over three different case studies, which are as follows:
  - The first case study supports the feasibility of the manufacturing control concept. The case study involves the cooperation between a dual-arm robot (Baxter from rethink robotics) and a worker. During this case study, the worker uses a variety of hand gestures to cooperate with the cobot to achieve the highest production flexibility. The worker hand gestures are recognized by the Leap Motion sensor.
  - The second case study addresses the shared environment representation in the cooperative workcell. The case study involves two workers in cooperation with one cobot. An ontology model of the cooperative workcell has been developed to provide a common understanding among the workcell entities.
  - The third case study addresses the cooperative workcell scheduling. The case study involves one worker in cooperation with one cobot. The HCA applies Johnson's rules to minimize the production makespan.

Appendix

Appendix A - Worker FB

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE FBType SYSTEM "http://www.holobloc.com/xml/LibraryElement.dtd" >
<FBType Name="WorkerFB" Comment="Basic Function Block Type">
        <Identification Standard="61499-2" />
        <VersionInfo Organization="Rockwell Automation" Version="0.2"
                Author="JHC" Date="2003-12-04" Remarks="Renamed for better indexing." />
        <VersionInfo Organization="Rockwell Automation" Version="0.1"
                Author="JHC" Date="2002-11-05" Remarks="Corrected missing ST element in Algorithm REQ." />
        <VersionInfo Organization="Rockwell Automation" Version="0.0"
                Author="JHC" Date="2000-05-30" />
        <CompilerInfo header="package fb.rt;">
        </CompilerInfo>
        <InterfaceList>
                <EventInputs>
                        <Event Name="Worker_Registration_Input_Event">
                                <With Var="Worker_ID" />
                                <With Var="Worker_Location" />
                                <With Var="Worker_Capabilities" />
                                <With Var="Worker_Status_Input" />
                                <With Var="Worker_Availability_Input" />
                        </Event>
                        <Event Name="Worker_Deregistration_Input_Event">
                                <With Var="Worker_ID" />
                                <With Var="Worker_Location" />
                                <With Var="Worker_Capabilities" />
                                <With Var="Worker_Availability_Input" />
                                <With Var="Worker_Status_Input" />
                        </Event>
                        <Event Name="Worker_Availablity_Input_Event">
                                <With Var="Worker_Status_Input" />
                                <With Var="Worker_Availability_Input" />
                        </Event>
                        <Event Name="Task_Status_Input_Event">
                                <With Var="Task_Status_Input" />
                        </Event>
                        <Event Name="Task_Assignment_Event">
                                <With Var="Task_ID" />
                                <With Var="Task_Description" />
                                <With Var="Input_Time_Stamp" />
                                <With Var="Task_Status_Input" />
                        </Event>
                        <Event Name="Constrain_Input_Event">
                                <With Var="Constrain_Input" />
                        </Event>
                </EventInputs>
                <EventOutputs>
                        <Event Name="Worker_Registration_Onput_Event" Comment="Initialization Confirm">
                                <With Var="Worker_Local_Information" />
                                <With Var="Output_Time_Stamp" />
                                <With Var="Worker_Status_Output" />
                                <With Var="Worker_Availability_Output" />
                        </Event>
                        <Event Name="Worker_Deregistration_Output_Event">
                                <With Var="Worker_Local_Information" />
                                <With Var="Output_Time_Stamp" />
                                <With Var="Worker_Status_Output" />
                                <With Var="Worker_Availability_Output" />
                        </Event>
                        <Event Name="Worker_Availablity_Output_Event">
                                <With Var="Output_Time_Stamp" />
                                <With Var="Worker_Availability_Output" />
                        </Event>
                        <Event Name="Worker_Status_Output_Event">
                                <With Var="Worker_Status_Output" />
                                <With Var="Output_Time_Stamp" />
                        </Event>
                        <Event Name="Task_Information_Output_Event">
                                <With Var="Current_Task_Information" />
                                <With Var="Output_Time_Stamp" />
                        </Event>
                        <Event Name="Constrain_Output_Event">
                                <With Var="Constrain_Output" />
                                <With Var="Output_Time_Stamp" />
                        </Event>
                </EventOutputs>
                <InputVars>
                        <VarDeclaration Name="Worker_ID" Type="WSTRING" />
                        <VarDeclaration Name="Worker_Location" Type="WSTRING" />
                        <VarDeclaration Name="Worker_Capabilities" Type="ARRAY" />
                        <VarDeclaration Name="Worker_Status_Input" Type="WSTRING" />
                        <VarDeclaration Name="Worker_Availability_Input" Type="WSTRING" />
                        <VarDeclaration Name="Task_ID" Type="WSTRING" />
                        <VarDeclaration Name="Task_Description" Type="WSTRING" />
                        <VarDeclaration Name="Task_Status_Input" Type="WSTRING" />
                        <VarDeclaration Name="Constrain_Input" Type="WSTRING" />
                        <VarDeclaration Name="Input_Time_Stamp" Type="DATE_AND_TIME" />
                </InputVars>
                <OutputVars>
                        <VarDeclaration Name="Worker_Local_Information" Type="ARRAY" />
                        <VarDeclaration Name="Worker_Status_Output" Type="ARRAY" />
                        <VarDeclaration Name="Worker_Availability_Output"
                                Type="ARRAY" />
                        <VarDeclaration Name="Current_Task_Information" Type="ARRAY" />
                        <VarDeclaration Name="Constrain_Output" Type="ARRAY" />
                        <VarDeclaration Name="Output_Time_Stamp" Type="DATE_AND_TIME" />
                </OutputVars>
        </InterfaceList>
    </FBType>
```

Appendix B - Worker Agent

```
public class WorkerAgent extends Agent {
	private static final long serialVersionUID = 1L;
	private AID OrderAgentAID = new AID("Order", AID.ISLOCALNAME);
	private WorkerUI wui;
	private Boolean WorkerStatus = false; // the worker status
	private Pump Pump;
	private WorkerAgent(WorkerUI wui) {
		this.wui = wui;
	}
	private WorkerUI getWorkerUI() {
		return this.wui;
	}
	public void setWorkerStatus(Boolean WorkerStatus) {
		this.WorkerStatus = WorkerStatus;
	}
	public Boolean getWorkerStatus() {
		return this.WorkerStatus;
	}
	public void setPump(Pump Pump) {
		this.Pump = Pump;
	}
	public Pump getPump() {
		return this.Pump;
	}
	public static WorkerAgent init(WorkerUI wui) {
		ContainerController cc = Resource_Holon_Container_Generator
					.getCCInstance();
		WorkerAgent W = new WorkerAgent(wui);
		try {
			cc.acceptNewAgent("Worker", W).start();
		} catch (Exception ex) {
			ex.printStackTrace();
		}
		return W;
	}
	protected void setup() {
		addBehaviour(new ReceiveOrderToBeExecuted());
	}
	public void SendWorkerDoneSignal() {

		ACLMessage msg = new ACLMessage(ACLMessage.INFORM_IF);
		setWorkerStatus(true);
		msg.setConversationId("WorkerDone");
		msg.addReceiver(OrderAgentAID);
		msg.setContent("Free");
		send(msg);
	}
	private class ReceiveOrderToBeExecuted extends CyclicBehaviour {
		private static final long serialVersionUID = 1L;
		MessageTemplate mt = MessageTemplate
					.MatchConversationId("PumpAssemblyOperation");
```

```
				public void action() {
					ACLMessage msg = myAgent.receive(mt);
					// ACLMessage reply = new ACLMessage(ACLMessage.CONFIRM);
					if (msg != null) {
						try {
							setPump((Pump) msg.getContentObject());

						} catch (UnreadableException e) {
							// TODO Auto-generated catch block
							e.printStackTrace();
						}
						WorkerEventHandler
								.fireTaskEvent("Order ID: " + getPump().getID() + "\n"
										+ "Order Amount: " + getPump().getAmount(),
										new WorkerEvent(getWorkerUI()));
						WorkerEventHandler.fireStatusEvent("Busy", Color.RED,
								new WorkerEvent(getWorkerUI()));
						setWorkerStatus(true);
						// reply.addReceiver(OrderAgentAID);
						// send(reply);
					}
					else {
						block();
					}
				}
		}
}
```

Selbständigkeitserklärung

Ich erkläre, dass ich die eingereichte Dissertation Selbständig und ohne fremde Hilfe verfasst, andere als die von mir angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzen Werken wörklich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Ich versichere weiterhin, dass ich bisher weder die vorliegende Dissertation noch Teile von ihr als Prüfungsarbeit oder zum Zweck der Promotion eingereicht bzw. verwendet habe.

Ahmed Rabee Ahmed Sadik