ALI HUSSNAIN

APPLICATION OF CLOUD ROBOTICS FOR AUTOMATIC MANIPULATION

Master of Science Thesis

Examiner: Professor Jose L. Martinez Lastra.
Examiner and topic approved on 2nd May 2018.

ABSTRACT

ALI HUSSNAIN: APPLICATION OF CLOUD ROBOTICS FOR AUTOMATIC MANIPULATION
Tampere University of technology
Master of Science Thesis, 73 pages
MAY 2018
Master’s Degree Programme in Automation Engineering
Major: Factory Automation and Industrial Informatics Engineering
Examiner: Professor Jose L. Martinez Lastra
Supervisor: Dr. Borja Ramis Ferrer

Keywords: cloud computing, cloud robotics, Manufacturing systems, Internet of things, knowledge-based systems, machine vision, cyber-physical systems.

Robots are an integral part of industries involved in manufacturing and assembly process. Changing workpiece requires a change in the programming of robot resulting downtime and loss of output in industrial scenarios. With the increase of population necessities and demands have also grown exponentially. Amount of resources available on the shop floor is always challenging due lack of capital investments. Changing market requirement often require new resources, tools and equipment to accommodate new product requirements. With growing populations and consumer demands, there is more need for manufacturing systems which can share resources saving setup cost and maintenance expenses.

Industry 4.0 emphasis on the development of cyber-physical systems to bring new functionalities in existing manufacturing systems. Cloud computing is a one of key paradigm from cyber world to overcome the lack of resources in the physical world by providing additional resources that can be shared by multiple manufacturing systems. These resources range from simple storage to application to application able to execute complex algorithms.

This thesis implements a system which deploys cloud-based vision capability in existing industrial robots. The implemented system uses a knowledge base that contains information about the workpiece. An interactive web interface provides a platform to update the knowledge base with new workpiece data. The system utilizes workpiece specifications in the knowledge base to update robot programming at runtime to achieve automatic manipulation. The designed system also allows different operations to be associated with the workpiece making it possible not only to handle different workpiece but also perform various operations. The thesis work provides a pathway and guidelines to develop a basic robotic flexible manufacturing system which can update its manipulation mechanism accordingly to workpiece data at runtime.

PREFACE

بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ

‘In The Name of Allah, The Most Gracious and The Most Merciful’

اللّٰهُمَّ صَلِّ عَلَى مُحَمَّدٍ، وَعَلَى آلِ مُحَمَّدٍ،

علی علیه السلام امام من است و منم غلام علیعلیه السلام

This thesis work represents a long journey which was not possible without the support and motivation provided by people around. I will take this opportunity to thank those people who made this journey possible.

I am deeply thankful to my supervisor Dr. Borja Ramis for his support, guidance at every step of this thesis work. I am grateful to him for addressing my queries and trying his best to solve whenever possible regardless of his busy schedule. I still remember the day when my journey started and today what I have accomplished that is only possible because of his endless support and encouragement. I would especially like to express my thanks to Professor Jose Lastra for giving me an opportunity to work under his supervision and supporting me with his kind gestures. I would also like to thank Anne Korhonen for her support and kind words during my work in FAST-lab. Her encouragement and help made this work experience a pleasant memory.

I would like to show my gratitude to my friends Shah Hussain, Dilshad Ali and especially Syed Manzar Abbas Kazmi for always motivating me and helping me on this journey. Their motivating talks during tough time always pushed me to achieve best from every situation

I could never repay my parents, brother and sisters for their prayers, support that made possible for me to accomplish this milestone in my life. I have no words to express my love to them for being by my side in every single part of my life.

In the end, I would like to thank Peer Syed Qasim Ali and Peer Syed Wilayat Hussain shah for their blessings and prayers which made every victory in my life possible. I still remember their kind words and faith in me, today that surely have come to reality in the form of my achievements.

Thank you everyone, for your support.

Ali Hussnain

4 May 2018

Tampere, Finland

CONTENTS

LIST OF FIGURES

LIST OF CODES

LIST OF FLOWCHARTS

LIST OF SYMBOLS AND ABBREVIATIONS

3GPP	3rd Generation Partnership Project
ACC	Agent Communication Channel
AMS	Agent Management System
CPS	Cyber-Physical system
DMSs	Dedicated Manufacturing Systems
DACS	Designing Agent-based Control Systems
DF	Directory Facilitator
FMSs	Flexible Manufacturing Systems
FIPA	Foundation for Intelligent Physical Agents
HTML	Hypertext Markup Language
IaaS	Infrastructure as a Service
IRF	International Robotic Federation
IoT	Internet of Things
JSON	JavaScript Object Notation
M2C	Machine to Cloud
M2M	Machine to Machine
MAS	Multi-Agent Systems
NIST	National Institute of Standard and Technology
OCR	Optical character recognition
PaaS	Platform as a Service
QR	Quick Response Code
RMS	Reconfigurable Manufacturing System
REST	REpresentational State Transfer
RDF	Resource Description Framework
SOA	Service Oriented Architecture
SLAM	Simultaneous Localization and Mapping
SaaS	Software as a Service
URL	Uniform Resource Locator
VM	Virtual Machines
W3C	World Wide Web Consortium

.

1. INTRODUCTION

1.1 Background

In 1913, the invention of moving assembly lines resulted in the emergence of the concept of mass production [1]. Mass production in industries was achieved by dedicated production lines and equipment. In the 21st century, companies are experiencing regular, unseen changes in the market led by worldwide competition and changing product requirements [1]. Companies are developing manufacturing systems that are able to accommodate rapid changes in market and customer demand.

Reconfigurable Manufacturing System (RMS) and Flexible Manufacturing Systems (FMSs) are the concepts that can be utilized to overcome the changing market and user requirements [2]. FMSs foresee these changes beforehand and incorporate flexibility in the system to accommodate them. Machine flexibility is one of the key components of FMS which allows execution of different operations without altering machine set-up [3]. In a RMS machine component can be included, eliminated or altered rapidly in reaction to changes in the workspace. RMS hardware and software can be reconfigured within minimum time span to adapt to new changes in a production environment. RMS and FMS play a vital role in reducing lead time for implanted new system and adjusting current systems. They also enable integration of new functionalities into the system without any extensive downtime.

The concept of fourth industrial revolution also known as industry 4.0, describes key technologies which are driving a new generation of manufacturing systems[4]. Autonomous robots, cloud and Internet of things are some of these technologies playing a vital role in the new generation of manufacturing systems [4, 5]. Industries have been using robots form quite a long time to handle industrial tasks. New development has taken place in the field of robotics resulting robots with enhanced functionalities and improved human-robot collaboration. Data from International Robotic Federation (IRF) show that use of the robots is increasing every year [6]. In 2015 there was 15% increase in sales of robots compared to last year, the annual average growth rate of 12% between 2016 and 2018 is observed. IRF estimates that in 2019, 2.5 million industrial robots will be working in industries. Above data is a clear representation of growing importance of robots in manufacturing systems. Industrial robots perform tasks more efficiently than humans with long-term consistency. Regardless of these growing numbers robots are still not working as RMS or FMSs which limit their true potential in the manufacturing sector.

Artificial intelligence and cloud computing are being more widely used in modern industries working on industry 4.0 concepts. Integrating the knowledge of environment and process beforehand in industrial machinery will surely increase there working ability. In the world of Cyber-Physical Systems (CPS), the functionalities of traditional machinery have been improved many folds by integrating cloud computing to provide remote computational access, storage, applications.

Recent development in technologies motivates one to pounder in developing FMSs robotic systems which will be able to perform dynamically hence reducing the number of additional robots in our industries.

1.2 Problem Statement

Traditionally robots are part of Dedicated Manufacturing Systems (DMSs) responsible for working on a specific type of workpiece [7]. This type of arrangement is suitable for bulk productions and in the situations where the product is produced over a large span of time. Robotic DMSs are unable to change their manipulation mechanism at runtime hence resulting in a working environment dedicated to specific type of workpiece.

Most of the existing industrial robots are working in DMSs due to lack of additional tools, available on factory shop floor which will enable a flexible working environment. Each robot dedicated to one process or part which result in unnecessary new purchases of robots for every process. If one robot is used to work on several workpieces, it require change in programming that result in increase of overheads in the process cycle.

The growing use of robots, changing market requirements, lack of funds and lack of available resources on factory shop floor results in some of the following questions:

- How to use cloud robotic concept for developing FMS consisting of industrial robots?
- Which cloud technologies can be integrated with the robots to enhance their functionalities and can be shared by other systems?
- How the entities in cloud robotic system will interact and information will flow between cyber and physical worlds?
- How to distinguish various workpiece with different structures using cloud tools?
- How to update manipulation mechanism in robots at runtime to accommodate workpiece changes?

1.3 Objective

The goal of thesis work is to develop a cloud robotic FMSs by integrating robots with cloud-based tool to handle various workpiece even those that are not defined in the system development stages. This system can update its manipulation mechanism and perform various operations as desired by the system user.

The main objective of the thesis work is

1. Designing of cloud robotics system for industrial robots.
2. Deployment of embedded hardware to include cloud-based vision capability in robots.
3. Development of a tool for defining the workpiece characteristics.
4. Development of communication protocol for this cyber-physical system for the interaction of cloud technologies from cyber world to the physical world
5. Developing automatic manipulation mechanism that can be updated at runtime.

1.4 Limitations, Assumptions and Challenges

The thesis work has following limitations:

- The change of gripper has not been accounted in system design.
- Although the system is designed in such a manner that it can work with most of the robots but testing has been done only with ABB.Minor changes will be required which include reference point adjustment, replacing robot functions with available in new robot controller.
- Assembly operation often requires the use of small force to join parts which is often considered as a collision that results in an error in the robots. Manual adjustments may be required in robot parameters.
- The system doesn't account for errors as result of light variations.
- The design system has fixed height approach for camera, large changes in workpiece height may result in camera to go out of focus.

Following assumptions have been made for achieving results and findings:

- All test workpieces are placed on reference point marked on the delivery pallet.
- Workpiece line of symmetry and gripper symmetry at reference position are aligned.
- The workpiece is always in focus area of the camera.
- The mass of workpiece is evenly distributed

During the thesis work following challenges were encountered

- Detecting and differentiating different workpiece was one of the key challenges but this was solved by using labels attached to the workpiece. These labels were detected by using cloud image analysis tools.

- Adding vision to the existing robotic hardware was another hurdle in the design of the system. This was overcome by developing a portable camera module that can be deployed on existing hardware without any alteration to the robot.
- Transfer of workpiece specification for automatic manipulation was also challenging but this was done by developing a data packet encoded by the main controller for the robot.
- The biggest challenge was testing of the system to avoid collisions and any uncertain motion. This was done by performing step by step execution of programs at slow speeds to stop the motion before any collisions.

1.5 Thesis Outline

The thesis has been divided into seven main chapters. The second chapter discusses the key concepts required to develop the understanding of thesis work. Chapter three discusses the research methodology adopted to achieve system objectives. Chapter four discusses the approach adopted and architectural overview of the system. It also discusses and justifies the technologies and tools selected toward achieving goals of thesis works. Chapter five discusses the actual implementation done to validate the results of the designed system. Chapter six provides the results achieved as the result of implementation and chapter seven concludes the thesis work.

2. THEORETICAL BACKGROUND

The following sections discuss key theoretical concepts which play an important role in the development of the flexible robotic manufacturing system capable of automatic manipulation. The first section discusses the ongoing industrial evolution toward industry 4.0. The second section discusses multi-agent systems that provide guidelines and tools for the development of distributed architecture. The third and fourth section provide an overview of cloud computing and its integration to the robot to enhance their functionalities. The last section discusses knowledge representation in manufacturing systems that is used to overcome anticipated changes in manufacturing process.

2.1 Industry 4.0

Over time industries have gradually evolved to newer efficient versions. The developments over the years led changes in industries to accommodate these new technological developments. Until now there have been four main industrial evolutions. Figure 1 describes these industrial revolutions and developments during those eras. The first Industrial Revolution was the result of mechanization. The invention of steam engine led to mechanized industries. The second industrial revolution was the outcome of electricity. The availability of electricity gave a new shape to industries. The electricity facilitated mechanization, thus enabling mass production. The electromechanical systems driven by electricity accelerated the growth of industries and enhanced the productivity. The development and progress of information technology initiated third industrial revolution based on autonomous manufacturing system. The development of information technologies enabled systems with onboard processing power, processing the environment variables and making basic decisions.

After the three industrial revolutions, the development in industries was incremental compare to breakthroughs in IT, e-commerce and mobile communications [4]. Increased population, changing markets and variable user demands brought serious challenges to the industries. The lack of resources to cope with these challenges paved a path toward another industrial revolution. The next wave of advancement brought the existence of new digital industries named as industry 4.0.

The concept of industry 4.0 was provided by National academy of science and engineering [8].The industry 4.0 aims to develop industries where the physical entities are coupled with the cyber world to develop more efficient systems to run industries. Industry 4.0 establishes smart industries where unique cyber-physical systems are formed in which humans and machines interact with each other as in a social network. In the manufacturing domain, these Cyber-Physical Systems consist of smart machines, warehousing systems and production environment capable of sharing information, initiating actions and controlling each other independently [5]. This integration will help in manufacturing, engineering supply chain and overall management of industries.

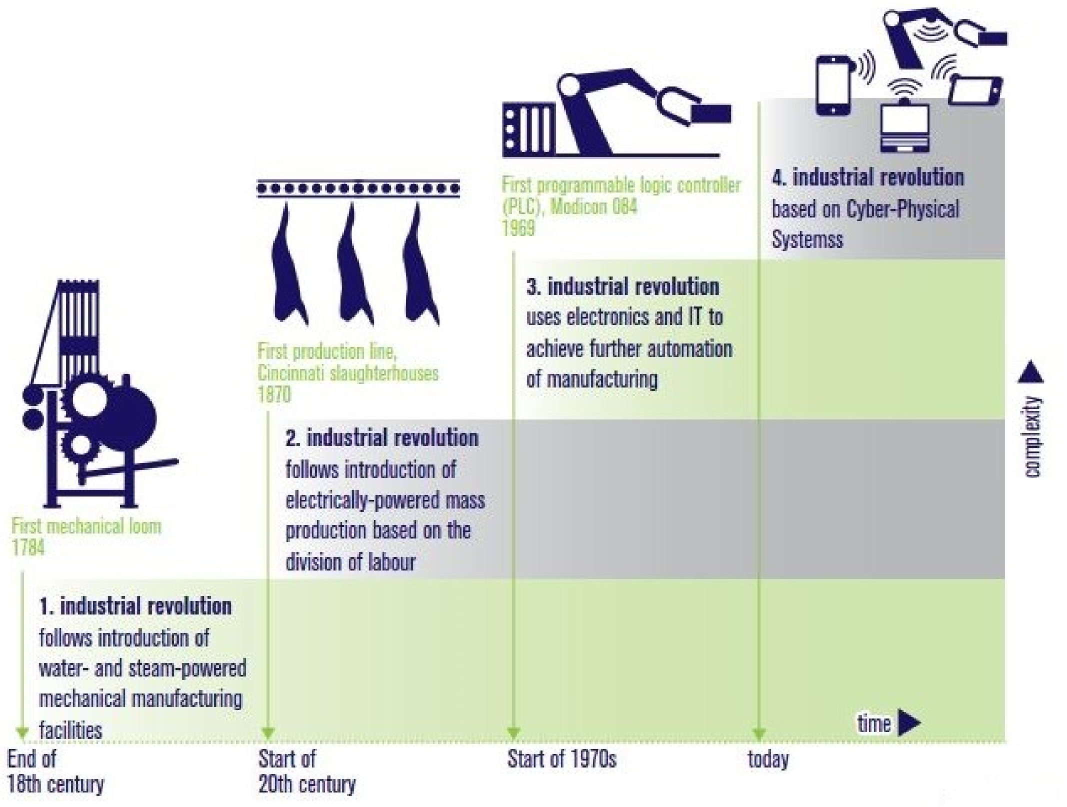

First mechanical loom
1784
1. industrial revolution
follows introduction of water- and steam-powered mechanical manufacturing facilities
First production line, Cincinnati slaughterhouses
1870
2. industrial revolution
follows introduction of electrically-powered mass production based on the division of 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 Systemss
complexity
time
End of 18th century
Start of 20th century
Start of 1970s
today

Figure 1.*Industrial evolution and technological developments [5].*

The transformation of traditional industries to industry 4.0 is powered by development and advances in nine technologies described in Figure 2. These technologies will model production units from isolation to integrated, automated systems with greater efficiency. These key developments will also transform the relationship between machine and human, supplier, producer and costumer.

The benefits of fourth industrial revolutions are numerous and it has a positive impact not only productivity but also on employment opportunities, revenue growth and investments. In Germany Industries 4.0 has resulted in an increase in productivity of manufacturing sector by 90 billion to 250 billion euros [4].

The coming subsections discuss the key components which are playing important role in the growth of industry 4.0. The first subsection discusses Cyber-Physical systems providing there building blocks and architecture. The second subsection discusses Internet of Things and third provide an overview of Information communication technologies

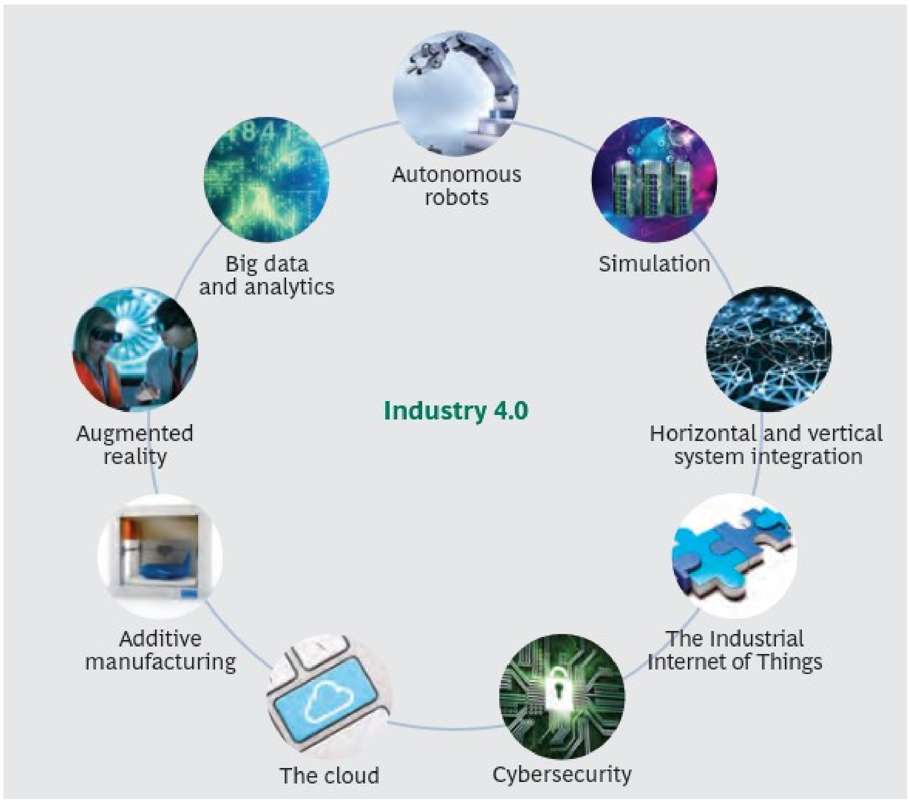


Figure 2.*Nine pillars of industrial transformation [4].*

2.1.1 Cyber-Physical System

Cyber-Physical system (CPS) is a fundamental part of industry 4.0. CPS are systems working as result of integration and collaboration of computational, communication and control technologies [9].Cyber-physical systems are the merger of different fields of science to enhance the output and increase efficiency. Different researchers have viewed and defined cyber-physical systems in a different manner. Lee defines Cyber-physical systems as” the integration of computation with physical process [10]. Coupling the cyber and physical systems by a feedback loop helps to do real-time interactions to control and monitor systems in efficient, safe and reliable manners.

Cyber-Physical systems are composed of several building blocks which integrate to develop the functional system. Figure 3 describes these building blocks. The Physical systems contain two main parts i.e. sensor network and actuator network. The sensor network is responsible for sensing and acquiring the data from target environment. The data must be real-time, secure and reliable. Advance sensor networks include embedded device integrated to them for analog to digital conversion, encryption, and transmission. Actuator network in a CPS is responsible for the execution of control decisions to change the state of the target environment. Actuators must be able to translate the control commands directly or indirectly through the intermediate device for accurate execution of commands.

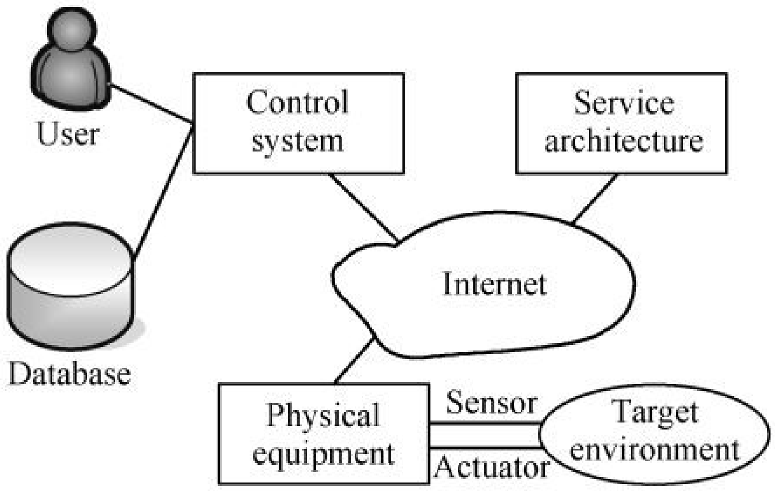


Figure 3.*General building blocks of CPS [9].*

In CPS, communication network between physical and cyber world is critical to data flow and control decisions. This communication interface must be secure, reliable and support high data rates of communication. As Figure 3 describes Control system is an integral part of CPS responsible for harmonizing the system. Control system communicates with each node, acquire the desired information and then compute it accordingly to computation algorithms. It also generates necessary control messages for concerned nodes. Control system provides a user interface and linked to a database to acquire information for operations or update system outputs.

J. Lee provided an architecture for developing and deploying cyber-physical system namely 5C architecture [11]. This architecture contains two main functional components, first state of art communication technology which ensures real-time data flow i.e. sensory data from physical world and control signals from the cyber world. The second component of this architecture is a platform for data analysis, computation. Layer one described in Figure 4 represents sensors network responsible for data acquisition from the targeted environment. Layer two convert data into information using embedded devices and smart analysis algorithms.

From layer three in the 5C model cyber world starts. Data acquired from layer two is sorted on the bases of classification parameters by data mining techniques. Twin models of machine and components are created at layer three which are updated every time when the state of the counterpart is changed in the physical world. Layer four of 5C architecture is responsible for providing a user interface and perform control decisions. Layer five is a high-end layer work performing decision, configuration and analysis to optimize the overall cyber-physical system.

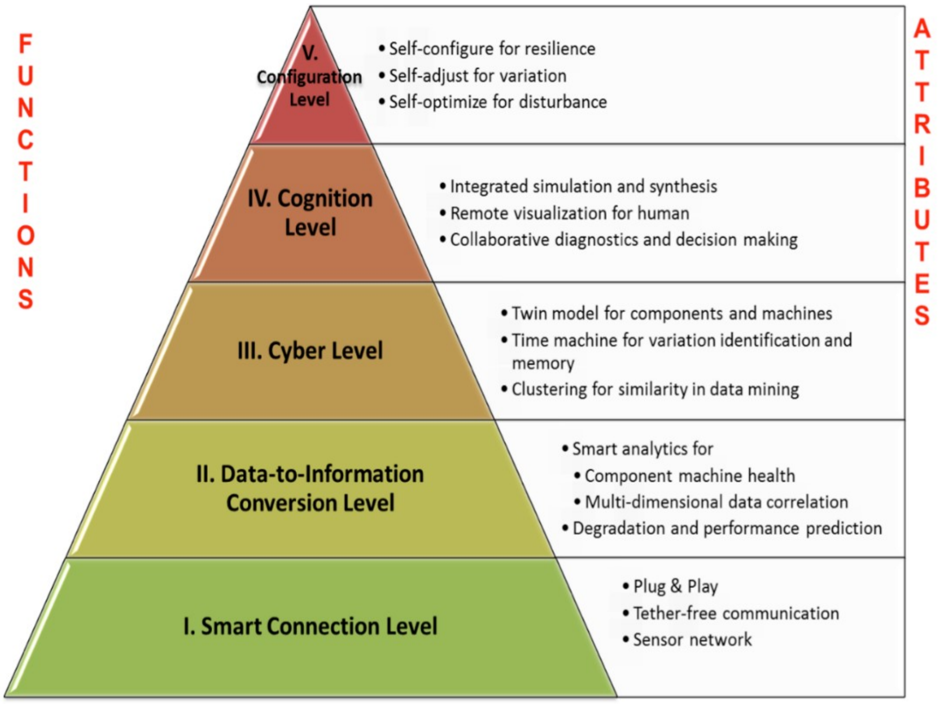

FUNCTIONS
ATTRIBUTES
V. Configuration Level
• Self-configure for resilience
• Self-adjust for variation
• Self-optimize for disturbance
IV. Cognition Level
• Integrated simulation and synthesis
• Remote visualization for human
• Collaborative diagnostics and decision making
III. Cyber Level
• Twin model for components and machines
• Time machine for variation identification and memory
• Clustering for similarity in data mining
II. Data-to-Information Conversion Level
• Smart analytics for
• Component machine health
• Multi-dimensional data correlation
• Degradation and performance prediction
I. Smart Connection Level
• Plug & Play
• Tether-free communication
• Sensor network

Figure 4. *5C architecture of Cyber-Physical system [11].*

2.1.2 Internet of Things

With the growth of industry 4.0, Internet of things (IoT) has become more popular and extensively being used in modern industries. IoT serves as one of the key technologies resulted in transformation to Industry 4.0. IoT devices play important role in the development of CPS.

IoT is an information network consisting of connected devices, such as sensors, actuators [12].The main perception behind the development of IoT was to develop a cyber web where different entities can interact with each other and collaborate to achieve common goals. The concept of IoT equips traditional devices with sensing, networking and processing capabilities enabling them to communicate with one another and service over the internet to accomplish some tasks [13].

Internet of things paradigm enables an environment where different objects are connected through wired or wireless connection to interact with each other to create new applications or services [14].

IoT plays important roles in creating smart industries, smart cities smart homes etc. IoT has enabled devices to be connected anywhere, anytime using suitable network protocols. Development of IoT has totally changed the use of the internet. Objects make them known, achieve intelligence to make decisions by communicating with each other. IoT devices are being used to develop smart houses, autonomous cars etc. These devices can provide notifications, data logging, web interface and even basic control decisions. Use of IoT has integrated security, automation, telecommunication, computers and entertainment into a combined ecosystem with a shared interface controlled with ease and convenience.

The Typical IoT architecture can be distributed into main three components sensing, network and application [15].The sensing domain plays an important role as it enables IoT device to sense the target environment. Sensing domain may process the information before delivering to the network part of the system. The network part of IoT is based on communication infrastructure such as WIFI,2G,4G, satellite, Lora WAN. The main role of the network part of IoT system is to transmit data to remote user e.g. operator, cloud or processing station. The network domain of IoT varies device to device, depending upon the communication hardware. The application domain described in Figure 5 is related to services. It provides a user or allows him to develop an application which processes and visualize the data received from devices

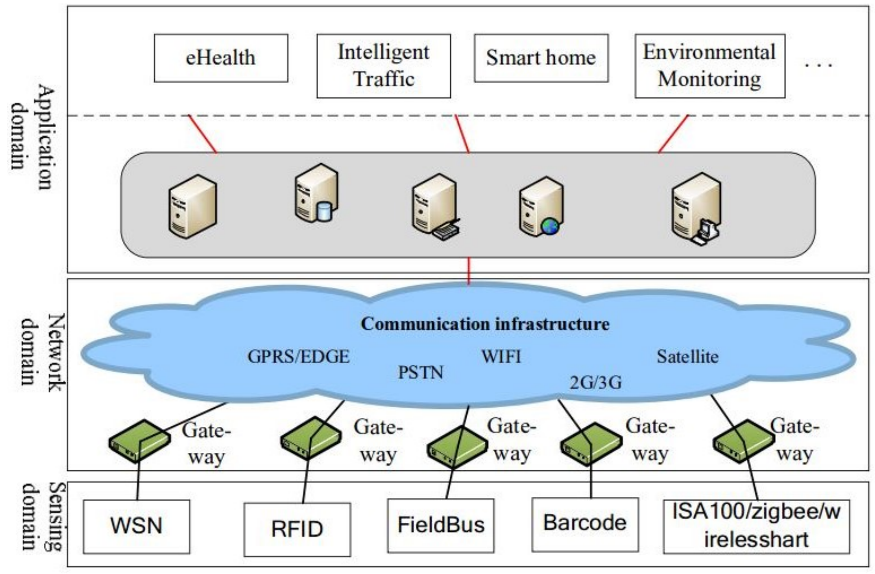


Figure 5.*Three domains of IoT [15].*

2.1.3 Information Communication Technologies

Industry 4.0 emphasis on cyber-physical systems which is not possible without the proper communication medium between the physical and cyber world. The choice of medium of communication should be done on following considerations:

1. *Latency*: Real-time communication is very critical for the working of systems integrated into industrials scenarios. A mode of communication with high rates of data transmission and minimum communications overhead is most appropriate developing the network layer of CPS.

2. *Reliability*: communication technology use for CPS should support data security and tolerance against noise. Data must be delivered in true form to get reliable communication between devices and prevent any hazards.
3. *Mobility*: The CPS may be in the stationary or transient state, the communication networks must hold the stable communication between the components in either state. CPS are integrated into fast trains, aircrafts moving at high speed and continuously communicating with other devices. Communication network must be able to support the devices in the mobile state without any disruption and losses.
4. *Scalability*: The communication network can be upgraded and has the ability to add more devices to support the growth of industries.

As described in Figure 4 and Figure 5 network domain is an integral part of IoT and CPS. Traditional networks are not able to support the rapid growth of IoT devices due to challenges regarding latency, scalability. Researchers have been working on finding new efficient solutions which can support the growth of IoT with real-time working.

The 3rd Generation Partnership Project (3GPP) between groups of telecommunications associations is working on enhancing existing LTE networks to support the increased traffic generated by IoT devices [16]. With the evolution of 5G networks, the IoT devices communication will be much faster and reliable. 5G networks will support real-time updates regarding changing industrial environments.

Below are some means of communication being used in industries for transfer of information from factory shop floor to the cyber world:

1. *Fixed Connection Access*: Traditionally the most common method being used in industries. The IoT devices are connected through wired connections to the controller or processing device. In past, the wire was normally made of copper but now fiber optic cables are most commonly used for transfer of data due to high reliability and low latency.
2. *WIFI*: Wireless devices have been used increasingly due to the evolution of embedded device with built-in wireless circuitry. Wireless technology has been very useful for remote monitoring and control of devices but still faces some challenges regarding power consumption, security and coverage etc.
3. *Lora WAN*: low power wide area network which reflects the new trend in communications and development of IoT devices [17]. Lora WAN has lower data rates as compared to 4G and 5G networks but provides some additional features such as the low power of operation, large transmission range. Lora devices utilize unlicensed frequency bands to communicate with the devices.
4. *5G Networks*: 5G technologies have revolutionized the growth of cyber-physical systems by providing high data rates for information transmission making it possible to achieve a real-time monitoring and controls. 5G has been widely used in autonomous vehicles, cloud robotics, healthcare and virtual reality [18]. The

emergence of 5G networks is a game changer and serves as the backbone for the Industry 4.0 enabling the integration of CPS in an industrial process.

2.2 Multi-Agent Systems

The complexity of the modern manufacturing system is increasing, thus resulting challenges in the development of control mechanisms. With the expansion of industries, the inclusion of new domains of cyber-physical systems there is a need of systems which can work in distributed architectures, perceive their environment and interact with each other to achieve common goals. These problems can be handled by using the agent-based approach to develop distributed, flexible manufacturing systems.

Agent is an entity with defined goals and targets [19].Agent accomplishes its goals independently but interacts with other agents in the system to achieve the targets of Multi-agent system(MAS). Over the years many researchers have defined agents in various ways but the core description is same. An agent is a software that has the capability to integrate into a system, allowing it to exchange information. Agent is autonomous in its working because no other entity has control over its function and internal states but collaborates with other agents toward common goals [20].

Agent is social and reactive in its nature [21]. Social because it collaborates with humans or other agents to accomplish its goals. Agent is reactive because its conscious of its environment and respond to any changes emerging in a timely manner

The following sub-sections describe some general types of agents and agent development methodologies.

2.2.1 Types of Agents

There several criteria to define agents which vary with system designer. Agent can be defined by its state i.e. stationary or mobile agent. Agents can be defined on several other parameters. N R Jennings classifies seven types of agents in his research work [22].Below are these classifications :

1. *Collaborative Agents*: are self-governing, independent agents who interact with other agents to achieve a mutual goal. These agents are independent because no other agent can alter their internal states directly but negotiate with each other to coordinate their task. The main aim of developing a collaborative agent is to form a system in which the functionalities of the overall system are much more than each individual participating agent.
2. *Interface Agents*: have almost same functionalities to collaborating agent except interface agents interact with system designer to attain new information. A user can provide a feedback or some task-specific instructions through an interface.

Interface agent enables the developer to make real-time changes in the system for adapting to new scenarios.

3. *Mobile Agents*: are software with the ability to move through the proper network, performing the task in other systems and delivering information back to its host system. These are collaborative agents who not only can interact with the agents in their system but through the web, they can interact and perform their task in an environment of other systems.
4. *Information Agents:* are responsible for organizing, collecting and processing information from various sources. In this era of world wide web, there is bulk of information flowing, information agent plays a pivotal role in developing a task-relevant database.
5. *Reactive Agent*: are autonomous working agents which do not contain stored information about their environment. These agents interact in a simple manner but new interaction may emerge because of previous interactions. Reactive agents are dedicated to the simple task and are quite commonly implemented in robotics.
6. *Hybrid Agents*: Different type of agents have pros and cons which affect their usability. To achieve optimum performance researchers have introduced hybrid architecture with features from different agent models. Figure 6 describes the general architecture of hybrid agent. It contains three control layers to optimize the working of the agent. These layers are responsible for reacting to the environment changes and plan a new course of action for unknown scenarios. The hybrid agent also has knowledge base containing information about the environment, agents and interaction to enhance its working capabilities.
7. *Smart Agents*: are agents that truly doesn’t exist but reflect an idea for the future development of new class of agents. These agents are perceived to contain artificial intelligence with abilities to optimize, self-configure and learn or time.

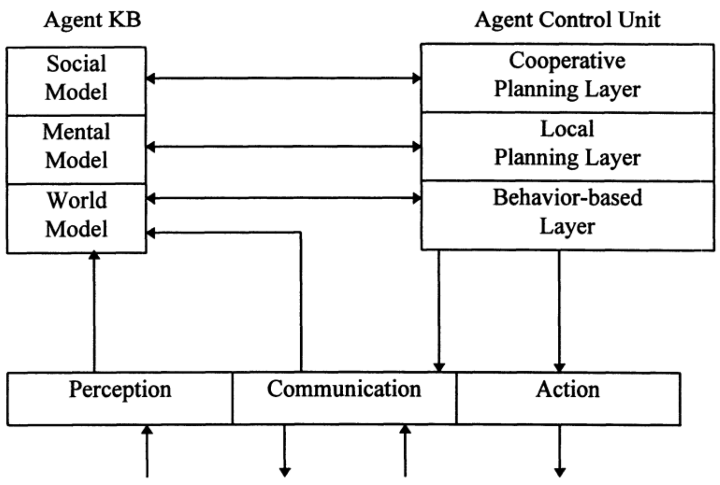


Figure 6.*Hybrid agent architecture [22].*

2.2.2 Foundation for Intelligent Physical Agents

Foundation for Intelligent Physical Agents (FIPA) is an organization which provides a group of standards to develop Multi-Agent Systems(MAS) and achieve interoperability between these systems [23]. FIPA provides list off-specification dealing with agent architecture development, interaction protocols, communication messages etc. to develop MAS with across the board homogeneity [24].

FIPA generated first document called FIPA97 specifications containing rules enabling agents to co-operate, operate and exist in a society of agents [25]. Figure 7 describes reference platform model provided by FIPA. In this model, three key components were defined. In this model, Agent Management System (AMS) is an agent dedicated for controlling access to use the platform. The Agent Communication Channel (ACC) is responsible for reliable communication between the platform and different type of agent promoting interoperability. The third component of the platform is Directory Facilitator (DF), which is an agent containing yellow page functionalities to the platform. The agent may register itself with DF or acquire information about the other agents.

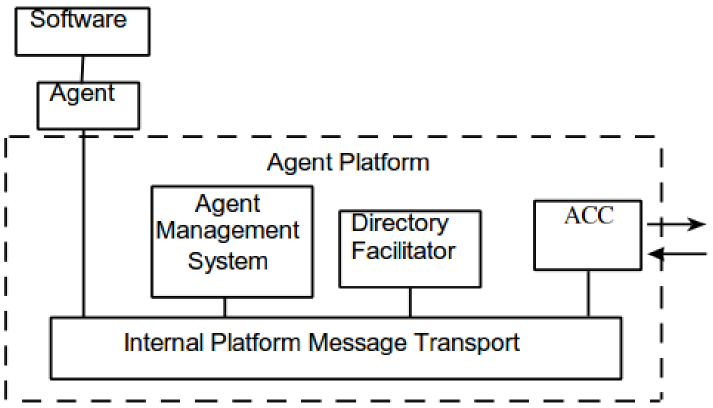


***Figure** 7.FIPA Agent platform reference model [25].*

FIPA specification never restricts developer creativity but provide guidelines and basics to steer developer thoughts that can be integrated to develop a society of agents[26].

2.2.3 Defining a Multi-Agent Systems

Defining agents in a multi-agent system is very important for smooth future working. Several methodologies have been used for designing MAS but often fail to provide desired outcomes. These methodologies were driven by data flow, manufacturing controls or some task-specific needs. In following subsection two main methodologies have been discussed to develop MAS.

2.2.3.1 Designing Agent-Based Control System Methodology

Bussmann, Jennings, and Wooldridge provide a methodology to develop MAS named" Designing Agent-Based Control Systems" (DACS) [27].To start the designing process the detail comprehensive analysis of the system is required. The problem of the existing system and the user requirement should be carefully studied before moving on with further design procedures. Figure 8 describes the steps and the procedures for designing agent-oriented manufacturing system.

To design MAS identification of control decision is very important. Control decision can be identified by an understanding of following key characteristics of each control decision.

- *Cause of control decision*: Any event or change in the environment that result control decision to become active.
- *The result of control decision*: Actions or outcomes that can be seen once control decision is executed.

Once the control decisions have been identified the next step is to identify co-dependent control decisions. In a manufacturing system, several control decisions may be involved in fulfilling a single task. After analyzing the control decisions, the next step is to identify agents. Agents can be identified by grouping control decision on the bases of some similarities. This procedure is repeated until a distinct set of agents is identified.

As Figure 8 describes, once the agent has been defined next step is to define the interaction between these agents. Interaction will depend upon the system requirements. The agent can communicate with each other to share information and coordinate for achieving system goals. DACS methodology is very effective but sometime may result in undesired outcomes, the steps can be repeated to attain suitable well-defined MAS.

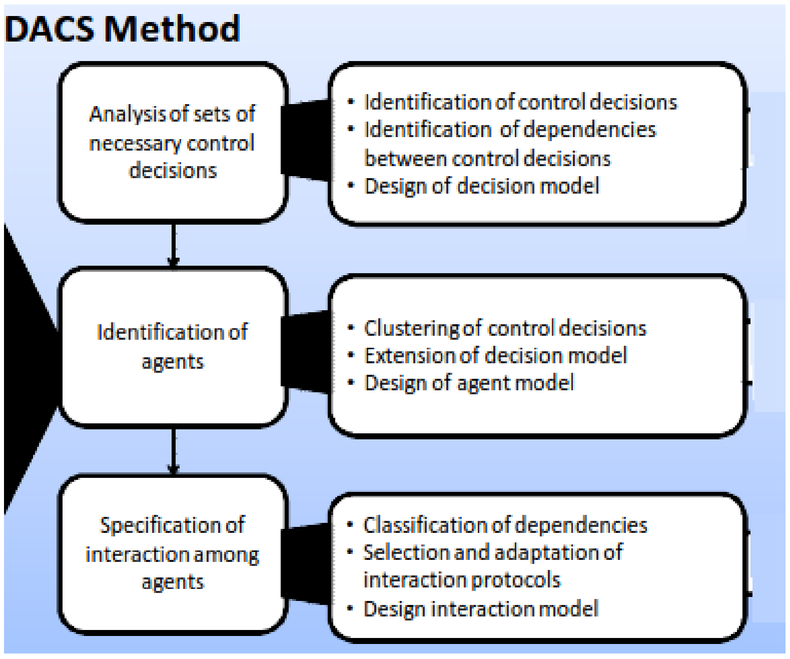

DACS Method
Analysis of sets of necessary control decisions
• Identification of control decisions
• Identification of dependencies between control decisions
• Design of decision model
Identification of agents
• Clustering of control decisions
• Extension of decision model
• Design of agent model
Specification of interaction among agents
• Classification of dependencies
• Selection and adaptation of interaction protocols
• Design interaction model

Figure 8.*Steps of Designing Agent-based Control System(DACS) [19].*

2.2.3.2 The Gaia Methodology

Gaia methodology is one of another popular design methodologies for developing a wide range of agent systems. Gaia methodology is very comprehensive in nature and equally resourceful in developing micro or macro MAS [28]. Figure 9 describes the models developed while developing agent-oriented control systems using Gaia methodology. In first step analysis of the system is performed to understand its requirement, goals etc. During the analysis, the understanding of the system is developed without any reference to practical details. When analysis of the system is performed two categories of models have generated i.e. roles and interaction models.

Roles models indicate roles in the system needed to achieve system goals. When a new role is established in the system it requires identification of its functionalities, resources and computations needed to carry that role. Different roles in a system depend on each other to accomplish system goals. Interaction model consists of a set of rules for interaction between different roles.

Once the analysis of the system is done the next step Gaia methodology defines as shown in Figure 9 is design phase. In the design phase, the models developed in analysis phase are translated into something that can be implemented. In the design phase, three models develop are agent model, service model and acquaintance model. Gaia agent model defines the agents that will be brought into use during practical implementation of a system to carry out roles defined in the design phase. Service models consist of services and functions that are linked to each role. The third and last model of the design phase is acquaintance model which consist information about communication channels between the agents.

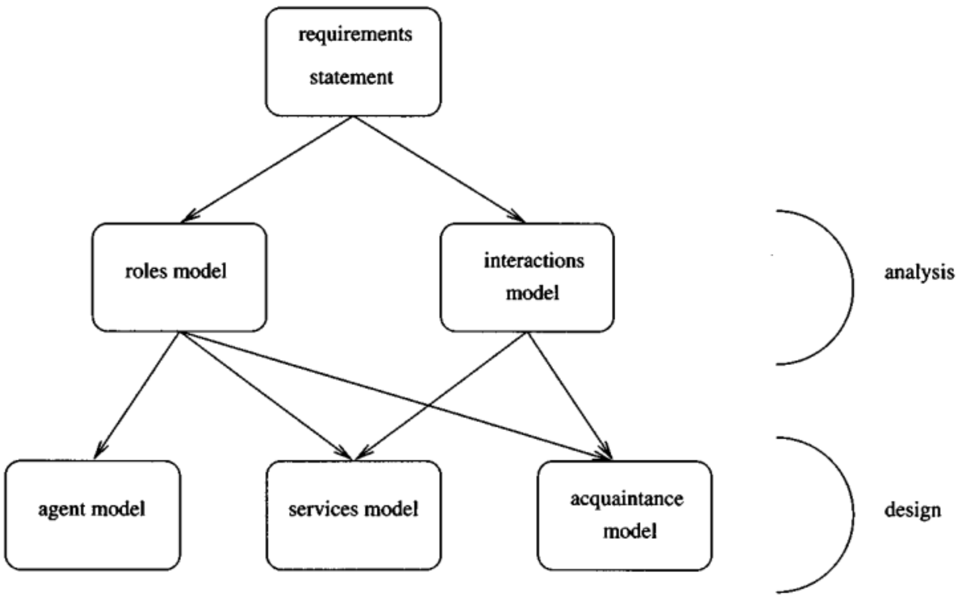


Figure 9.*GAIA methodology models [28].*

2.3 Cloud Computing

Cloud is one of the key driving technologies that resulted in transformation of modern industries. The growth of industry 4.0 and 5G communications has made cloud computing more frequently being used in industries. Researchers have been investing noticeable resources to determine new applications of cloud computing in industrials scenarios. Figure 10 shows relative percentage of usage of cloud in different sectors. The availability of internet and use of cyber-physical systems have elevated the use over the past years.

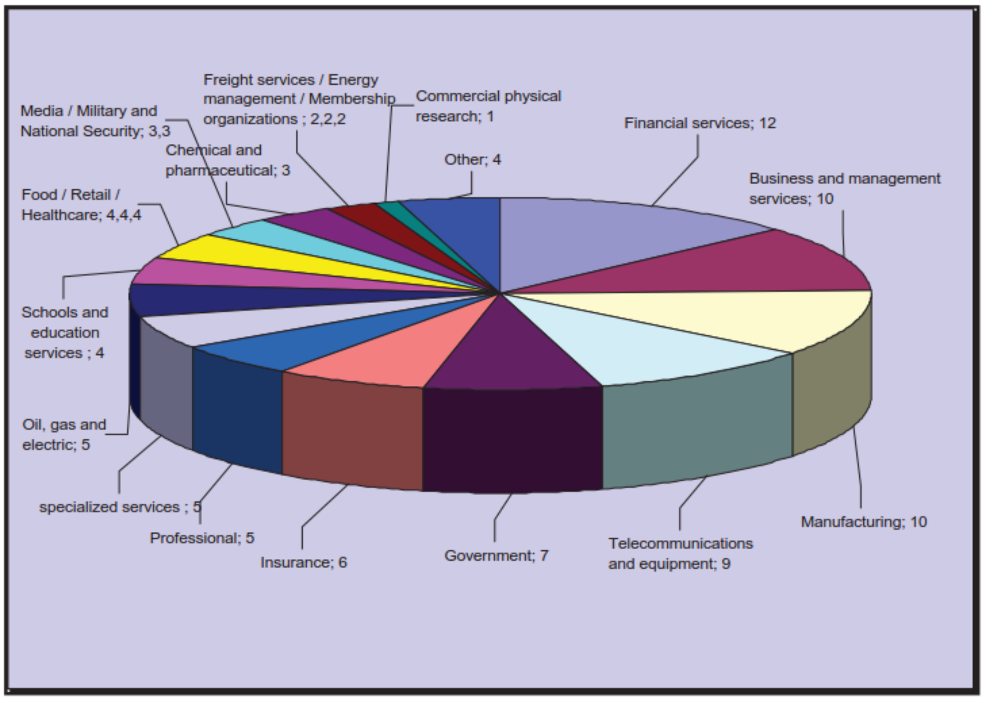


Figure 10. *Cloud computing usage percentage in different sectors in 2009 [29].*

The National Institute of Standard and Technology (NIST) define cloud computing as a paradigm that provides, easy, on request network access to a shared pool of configurable computing resources [30]. These resources may vary depending upon the system but generally may include e.g. storage, applications, and algorithms. These resources can be deployed or used without any additional workload by the consumer.

NIST defines five essential characteristics for designing a cloud base system. These main characteristics are:

1. Cloud should be autonomous in its function without the requirement of human input for servicing, e.g. it can reply to a query regarding storage status without any human input.
2. Cloud services are available across the network through standard protocols for all the users with all devices compatible.
3. Cloud model should be able to handle and distribute resources (physical and virtual) efficiently to multiple customers. Cloud model can assign resources to one or group of customers and then reassign them according to their requirements again and again as the process continues.
4. Cloud model can monitor the services and use it to control and optimize resources deployment toward the consumer.
5. Cloud model should have elastic resources which can be provided and released in appropriate quantity anytime as required by the system

2.3.1 Cloud Service Models.

There are three main service models provided by cloud platform [30, 31]. Figure 11 describes these three main cloud service models which are explained in coming sub-sections.

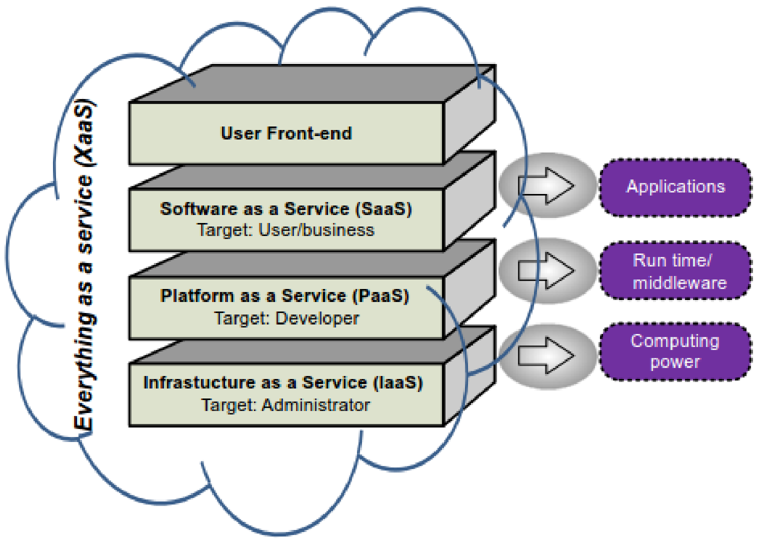


Figure 11.*Service platforms of cloud computing [32].*

2.3.1.1 Software as a Service

Software as a Service (SaaS) service model has received significant attention over the other service models. SaaS provides user facility to utilize various applications deployed by cloud designer. The applications can be accessed by the client on various devices using a simple web interface or a program interface. The user has limited configuration rights for this application. SaaS application usually run over the top of other models i.e. PaaS, IaaS [33]. The user cannot alter the underlying cloud infrastructure, operating systems without administrator rights.

SaaS service model has been widely used to provide engineering application which can be integrated into industrial scenarios. These applications involve advanced tools to perform computation-intensive tasks e.g. image processing, voice reorganization. Google is one of the leading providers of cloud-based application. These software are not needed to be purchased by the user but their usage fee may apply depending upon the individual policies of the provider.

Scalability is one of the important concern in SaaS model. An application may be used by several users who are providing service to more users. To provide reliable services through SaaS model developers have adopted two deployment techniques. Cloud developers may deploy the application on one machine with higher computation power, storage etc. making the user experience more efficient. Some developers may distribute the application on a number of computers with similar configurations to enhance user experience.

2.3.1.2 Platform as a Service

Platform as a Service (PaaS) model provides an environment to the user to deploy an application, developed or acquired from the third party on the cloud. The applications can be built using the tools made available by cloud provider such as libraries, services which comply the cloud architecture. The user can alter the application deployed but cannot make changes in base platform provided by the cloud service provider. PaaS not only provide development tool to the user but also testing, deployment and service tools for cloud user [31].

A very popular example of PaaS is Microsoft Azure. Microsoft Azure provides fast and efficient tools allowing the user to build and deploy applications without user worrying about the hardware resources[34]. Azure is versatile cloud platform supporting both Windows and Linux based operating systems with compatibility to all mobile devices. In PaaS, there are three main stakeholders which include PaaS host provider, and user [35].

PaaS host provides sufficient resources and backup plans for fulfilling needs of consumers efficiently. PaaS provider is responsible to provide an environment to utilize these resources so the targeted users can build their applications. PaaS user can use a web-based interface for developing desired applications. The user is charged only for the usage of resources.

2.3.1.3 Infrastructure as a Service

Infrastructure as a Service (IaaS) is a cloud service platform where cloud provider has made available its potential users the processing, storage and network resources. The users can utilize these resources for deployment of their own software or application on cloud. A user can install operating systems using the resources and provide services to its users. IaaS model is very helpful for the organization with potential expansions. Lack of processing resources can easily be compensated by using cloud base resources. The user has just to pay the resources being utilized.

Figure 12 describes the key components of IaaS model and the security model essential for reliable working. The key component of IaaS model is computer hardware. Computer hardware includes several items such as processing power, storage, visualization, network resources etc. These resources are provided through a communication network with low latency and large coverage. These resources are not owned by the user but only charged for the usage.

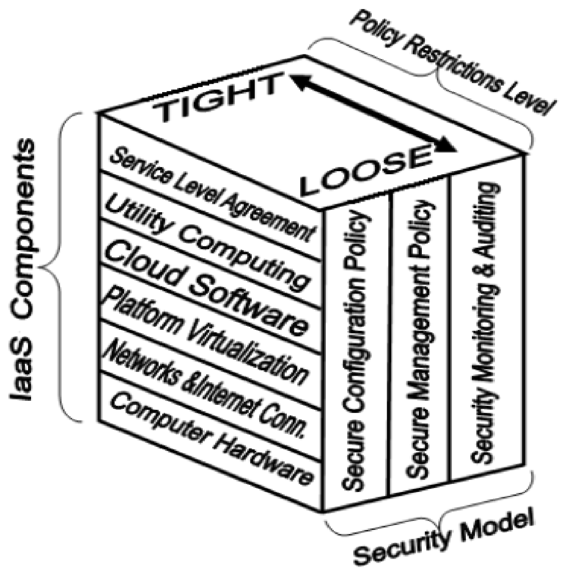


Figure 12.*IaaS components and security model [36].*

2.3.2 Cloud Deployment Model

Deployment of the cloud to right users is very critical to avoid security challenges. Modes of deployment define the intended users which can utilize the cloud services. There are four main deployment modes of cloud which are private cloud, community cloud, public cloud and hybrid cloud [30].

2.3.2.1 Private Cloud

Private cloud is only deployed and used by the single organization. There may be multiple users in the organization utilizing the cloud but no external user is able to use a private cloud. Private cloud is managed by the organization or by the third party with mutual understanding. The main benefit of the private cloud is that the management is much easier than the other modes. Private nature, controlled number of users helps to avoid security threats. Private clouds are normally used by small organizations with limited interaction with other organizations.

2.3.2.2 Community Cloud

When a group of organizations or people have common interests, requirements and share a common cloud this type of cloud is called community cloud. Community cloud can be maintained by one of user company or can be outsourced. Community clouds have a common security policy which is approved by each member organization. Community clouds are formed by the integration of private clouds on the bases of common interests.

2.3.2.3 Public Cloud

In public cloud, the cloud resources are available for public use. Public cloud can be accessed through a simple web browser or an application making it widely accessible. The user can use the cloud and pay a simple fee for the usage. Public cloud size varies depending on the services and resources available. Public clouds are most vulnerable to security threats so service provider often takes strong security measures to provide reliable secure service to the customers.

2.3.2.4 Hybrid Cloud

Hybrid cloud is a combination of two more cloud infrastructures. To achieve more functionalities different clouds are often used in coordination with each other. In developing a hybrid cloud the constituent clouds remain as unique infrastructures but are integrated together by standard protocols that allow the flow of data enabling portability of applications.

2.3.3 Cloud Computing Benefits

Below are some key advantages of cloud computing:

1. *Easy Management*: Cloud infrastructure provides resources which are not owned by users but can be utilized and readily deployed. The user is free from maintaining the hardware or software resources. There is no need for local teams to maintain the resources [37].
2. *Cost Reduction*: Cloud computing provides infrastructure in which user can pay on the bases of usage. The user is not concerned to any expenses regarding purchase and setup of the hardware and software resources.
3. *Scalability*: Cloud computing infrastructure is readily scalable according to demand of individual organization. More resources can be allocated without any additional expense and deployment overhead.

2.4 Cloud Robotics

The term “Cloud Robotics” was coined by J.J. Kuffner in 2010 [38]. Kuffner believed that the robots are restricted to perform in real environments due to the limited amount of processing capability, storage, programming with them. He proposed a concept to utilize the platform of internet. By use of information communication technologies robots can utilize parallel processing powers and share their knowledge with a network of robots. These resources are not physically situated with the robot but can be called upon without any deployment overheads. The concept of Cloud robotics opens new domains of research providing robots with state of advanced features, improved efficiency and rapid scalability.

Many different concepts and paradigms play important role in the development of cloud robotic system. Figure 13 shows the key concepts played an important role in the development of cloud robotic systems. Cloud computing as previously explained playing an integral role in providing remote resources, computational power to the robots. A cloud robotic system mostly follows an agent-oriented approach where robots act as agents interacting with the environment and one another to accomplish tasks. MAS provides guidelines to develop distributed architecture and interaction between agents i.e. robots and cloud. Cloud Robotic system can be developed without MAS approach but using MAS approach in parallel with Cloud robotic systems help to develop the system, with less complexity and more interoperability

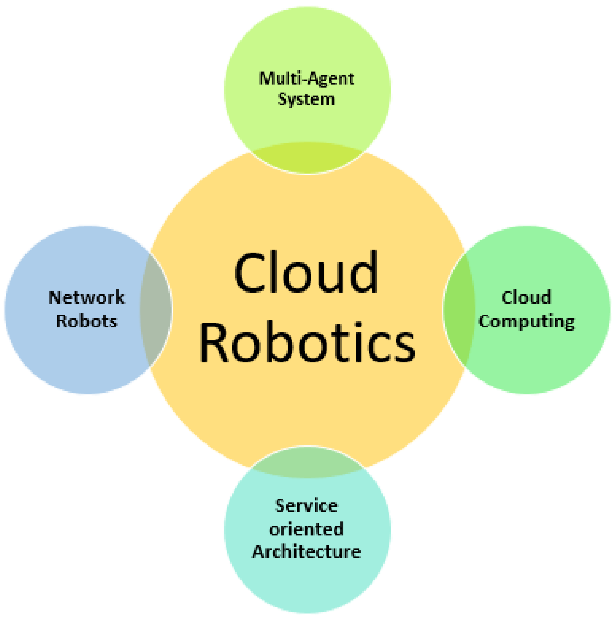


Figure 13.*Cloud robotics and supporting concepts.*

2.4.1 Network Robots

As mentioned in Figure 13 network robots play important role in the development of cloud robotic architecture. The concept of "*Network robots*" existed before the idea of cloud robotics. Network robots concept was proposed by Japanese researchers in 2002[39].A network robot has a capability to communicate through local area network or internet. To define a network robotic system there must be at least one robot which has autonomous hardware and software abilities. The robot, human and environment sensors can communicate with each other through proper network channels in a standard network robotic system [40].

Alberto Sanfeliu characterized two main types of robots that exist in network robotic system [40]. These two robots are:

1. *Teleoperated robots*: A user can send command and receive some notifications from robot through a proper communications network. These robots are very useful for remote operations, medical procedures, and monitoring.

2. *Autonomous Robots*: can share their information and sensory data through a communication network. These robots can extend their information by using data provided by other robots hence can perceive and interact with environments they are not actually part of.

Although network robotics is quite similar to cloud robotics but lacks one integral part i.e. cloud. In a network robotic system, robots are communicating with each other but they lack shared pool of resources. In cloud robotics, the cloud provides hardware and software resources such storage, computational power which can be used by robots remotely.

2.4.2 Service Oriented Architecture

Service Oriented Architecture (SOA) is very important in developing modern cloud-enabled robots. The service-oriented model integrates services to process. Service can be described as an interface that links to a process and able to alter the state of that process or environment [41].Services are modular, can be discovered on the network and made available to be used without any additional overheads

System complying service-oriented architecture contains three main components as described in Figure 14.Service requestor is a user desiring to accomplish a task and make a request to the service broker. Service broker contains information, configurations of all service provider. Once it receives a request from the user it provides the information of relevant service provider. The user can invoke that’s service to accomplish its desired tasks

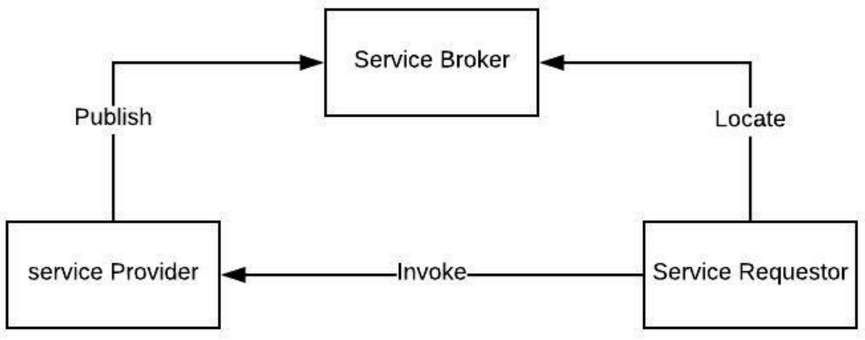


Figure 14. *Service-oriented architecture.*

In cloud robotic system, each robot can perform a certain task and these tasks can be called upon by other robots as services. A supervisory agent containing the information about the services of the individual robot can facilitate other robots to find the appropriate service provider. Using SOA in parallel with cloud robotic systems will help to develop robots that are capable of utilizing each other functionalities in more efficient manner hence improving the efficiency of the overall system.

2.4.3 Cloud Robotics System Architecture.

Guoqiang Hu provides a complete system architecture of cloud robotic system in his research work [42]. The cloud architecture can be divided into two main models and one hybrid model. Figure 15 represents an overview of the cloud architecture comprising of the robot to robot interaction, robot to cloud interaction and cloud resource sharing.

The cloud architecture and resources vary from system to system depending upon requirements, hardware and communication interface. A typical cloud may contain storage, processing tools, knowledge base or various application to facilitate robot carrying tasks.

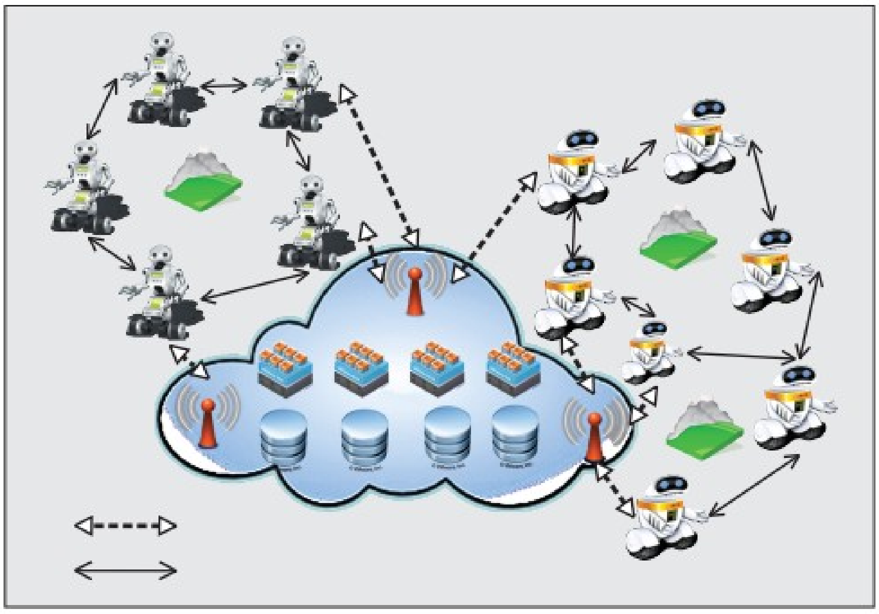


Figure 15.*Cloud robotics general architecture: machine to machine and machine to cloud and resource sharing [42].*

2.4.3.1 Machine to Machine Framework

At machine to machine (M2M) level robots can communicate with each other forming a collaborative network wirelessly. The processing abilities of individual robots are shared to form an Ad-hoc cloud platform.

Ad-hoc cloud utilizes capabilities and resources of participating entities making them available for group computational activities through proper network channel [43]. Ad-Hoc cloud doesn’t fulfill all principle of cloud computing. Ad-Hoc clouds resources are a volunteer so inconsistent and unreliable. A robot may become part of the ad-hoc cloud or leave depending upon individual task and restrictions. In Figure 15 the solid line represents a web of communications between the robots forming a temporary cloud for sharing of information

To establish communications between the robots there are several existing technologies such as ZigBee, WIFI, LoraWan and other radio technologies for short or long-range communications. The selection of medium of communication depend upon system designer and requirement of the system

2.4.3.2 Machine to Cloud (M2C) Framework

At Machine to Cloud (M2C) level cloud provide infrastructure for accessing mutually shared resources. Each robot in the system has its dedicated access to cloud resources which can be used in real time. As previously described in cloud computing chapter the cloud can provide different types of resources some of them are listed below:

- *Storage*: Each robot can store its data or retrieve data of other robots, recipes, instructions from the cloud.
- *Computation resources*: Depending upon the use case robots are often dealing with bulk data that require complex algorithms and computational infrastructure to convert raw data into some valuable information for process controls. Cloud computing can provide just not only hardware computational power but applications working on standard communication protocols. These applications receive data from robots process them return the results to the robot.

2.4.3.3 Hybrid Cloud Robotic Architecture

Guoqiang Hu further proposed three Hybrid working models that can exist in cloud robotics system [42].These models are composed of

1. Ad-hoc cloud infrastructure develops as the result of the M2M interaction.
2. Cloud computing result of M2C interactions.

These three models are:

1. *Peer-Based Model*: In this model robots and virtual machines (VM) existing in the cloud all are fully capable of performing computational tasks. A task can be distributed into subtasks performed by a group of robots or virtual machines.
2. *Proxy-Based Model*: In this model one of robot act as, the supervisory agent responsible for communication with the cloud resources. This leading robot is responsible handling all the data flow from the cloud and to the cloud.
3. *Clone-Based Model*: In this model, every robot has its virtual image in the cloud in terms of computational resources. Any task can be processed either in robot or in its image in the cloud.

2.4.4 Related Works

The benefits cloud robotics has attracted more research focus in the field over the years. Researchers are using cloud robotics platforms to integrate new functionalities to robots to improve their working. This section discusses some research work conducted by the different researchers. Four examples have been chosen to represent the implementation of the concept in various domains.

2.4.4.1 Robot Navigations

Raffaele Limosani team developed a cloud-based navigation system for robot movement in the indoor environment [44]. Robots can't hold every information about the map and topology of environment due to limited storage. The information of maps is stored in the cloud which can be accessed by Quick Response Code (QR) codes placed in the indoor environment. Figure 16 describes the working of the system. Whenever robot moves to a new area it can use QR codes to access information from cloud i.e. map, parameters sequence of motions from one point to another etc. This use case is clear reflection how cloud robotics has overcome the limitations of onboard storage problems in robots.

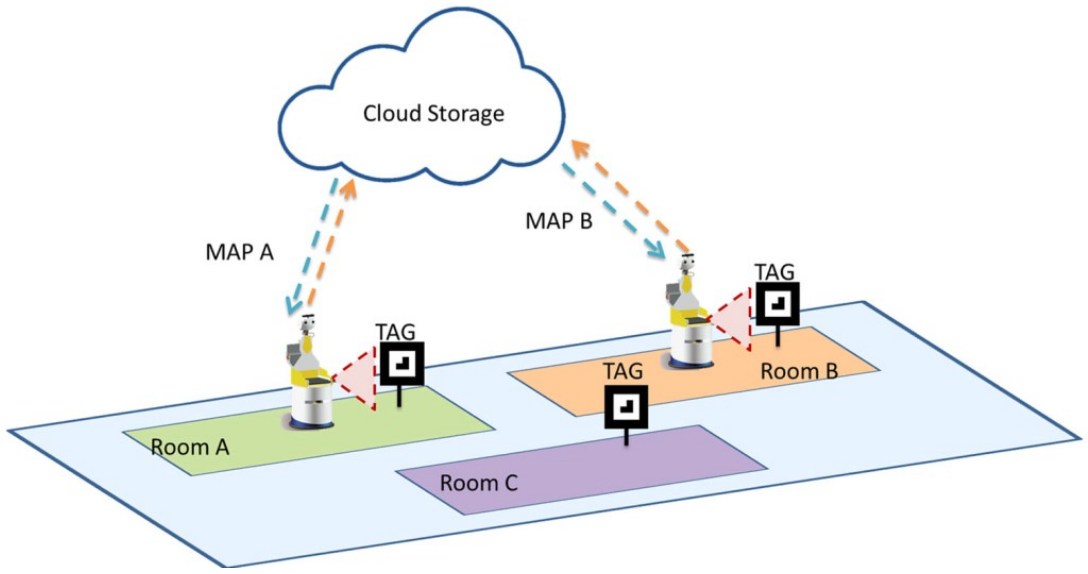


Figure 16.*Navigation of robots using QR codes [44].*

2.4.4.2 Robot Grasping

Object recognition is considered important for robots to perform tasks in an autonomous manner. Robots working in industrial scenarios often need to know the context of the object to perform certain operations. Due availability of cloud image processing techniques context awareness complexity has decreased manifolds. Researchers from the University of California has integrated Google Cloud Vision to robots for grasping a specific object from the group [45].

Robot use camera to attain picture of items in its environment and send to Google Cloud Vision for processing. The result is used by the robot to grab a specific object. This type of systems can be very useful while performing assembly, item-specific tasks.

2.4.4.3 Material Handling

Jiafu Wan and his team developed a context-aware cloud-enabled material handling system [46]. The information of equipment, environment, and orders are extracted from the factory shop floor and stored in the cloud. The main computational intensive tasks are performed on cloud i.e. analyzing data related to material, equipment, orders etc. through process specific algorithms. The information relevant to cloud-enabled robots is shared with them for updating their knowledge about the environment. The cloud platform also contains algorithms such as Simultaneous localization and mapping (SLAM) responsible for the navigation of robots to the targeted device. SLAM is highly computational intensive algorithms requiring mathematical computations [47]. The cloud also contains grasping algorithms for robots for material handling.

Figure 17 shows the working layout of the material handling system. The information of material is continuously updated in the cloud through RFID tracking devices. Whenever any material is required the robots are connected to cloud through M2C interactions from where they receive navigation information by SLAM and material grasping instructions Each robot on the floor is also connected through M2M interaction for sharing information of the context of material with neighboring robot for resource optimizations

This system utilizes various platforms of the cloud to offload computational intensive task, sharing of data in order to reduce the cost, improve the efficiency of the overall system. By achieving resource optimization, the energy losses were also reduced

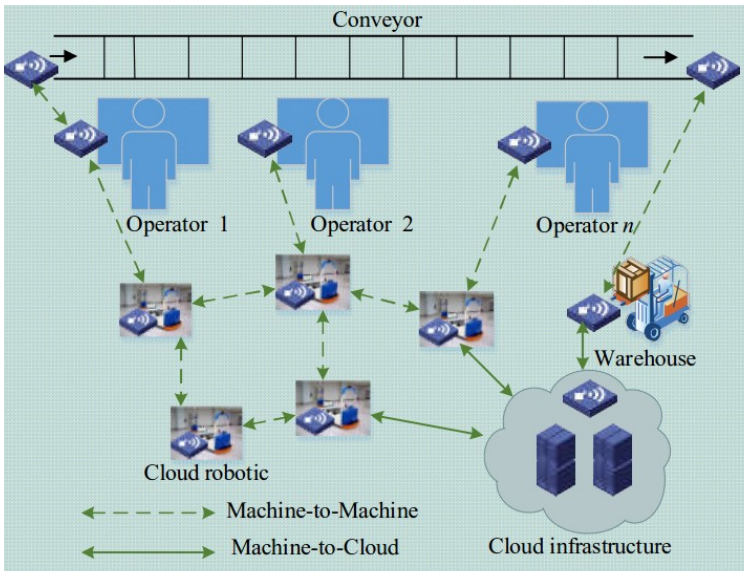


Figure 17.*Physical layout and interactions of cloud-enabled material handling system[46].*

2.4.4.4 RoboEarth

RoboEarth is a freely available cloud platform, facilitates sharing and reuse of data within robots through proper network standards [48]. RoboEarth increase working efficiency of the robot by enhancing robot ability to acquire new knowledge and familiarize itself with complex tasks. RoboEarth enables robots to perform a task which was not incorporated in a robot during development stages.

Figure 18 represents the general three-layered architecture of RoboEarth systems. Server layer is responsible for storing data sets. RoboEarth database may contain information about the environment, objects, images. It also contains set of tasks that can be performed, capabilities etc. Server layer also provides knowledge about the service Uniform Resource Locator (URL)s. Generic component layer in the architecture contains support functions which enable the flow of information between robot specific layer and server layer. The generic component layer contains tools which allow the robot to translate the information, recipes contained by server layer for robot use. Robot specific layer is responsible for the execution of recipes and establishing an interface for hardware relevant tasks.

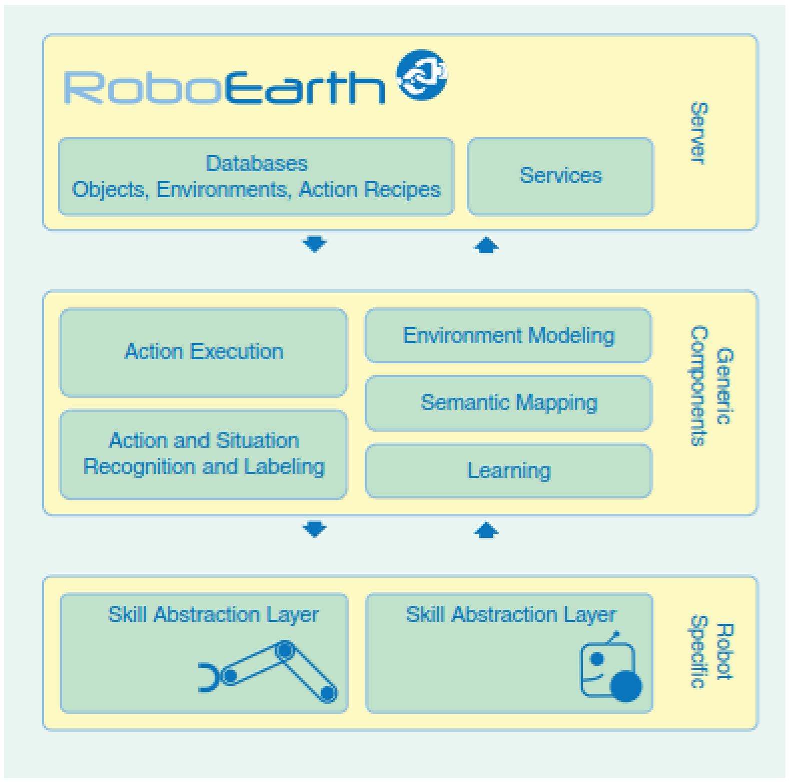


Figure 18.*RoboEarth three-layered architecture [48].*

RoboEarth platform has several components which can also be utilized individually to enhance functionalities of the robot.

Rapyuta is PaaS infrastructure developed to provide a computational platform for robots which can be configured according to the individual need of robots [48, 49].

RoboEarth provides a cloud-supported knowledgebase supported by web services. This knowledge base contains data represented in form of ontologies to describe the maps and objects in an environment [50]. It also contains SLAM map to describe environment parameter in relation to robot location.

RoboEarth platform also contains a newly developed language to facilitate sharing of knowledge among robots [51]. This language not only enabled sharing of information but contained features which help robots to discover information and analyze its usage or importance to the robot.

2.4.5 Challenges

This section discusses some problems and challenges faced in cloud robotics. Most of these challenges affect the overall efficiency of the system. JIAFU WAN conducted a vast research on cloud robotics and shared some challenges in his work [52].

2.4.5.1 Resource Distribution and Organization

As discussed in previous sections, the cloud provides access to resources to offload computationally intensive task. Offloading a certain task to cloud sometimes become complex due to a different type of network interface and tasks. Additional algorithms are required to accommodate multiple robot's interaction to the cloud. Resource distribution often results in additional overheads in the interaction between the robots and cloud. Robotic systems are getting complex with the development of new technologies. The offloading computational intensive task may result in a trade-off with communication latency. The real-time requirement must be kept in consideration while deploying a cloud robotic system to avoid problems. Transferring large amount of data for processing require communication resources with low latency that is still a challenge which cloud robotics systems are facing.

2.4.5.2 Information flow between robots and clouds

Robots are being manufactured following a variety of design protocols. These robots are equipped with hardware and sensors generating data in different formats. Compatibility issues often emerge during M2M and M2C interactions in a cloud robotic system. In recent developments, some of the cloud platforms are providing an interface to a different form of data structures. The available support for different data formats is still limited so often the robots are integrated with some of the converters to process data for the cloud. In real time scenarios processing data for transfer to the cloud often become challenging due to limited onboard processing power and communication infrastructure. The problem of interface compatibility is being faced not only while sending data for processing but also while receiving processed data from the cloud.

2.4.5.3 Security

Integration of cloud and web technologies where eased operations of robots but also made these systems vulnerable to external threats. Data privacy in cloud robotic systems is often challenging, loss of data to unauthorized users may result in serious problems. The stakes become high when these robots are being used in critical research organization such military units. Photos, audio, video files are often extracted by robots and forwarded to the cloud which can be leaked resulting catastrophic effect. To handle these challenges researchers are working on integrating different encryption and identity management software to achieve more reliable secure communications.

2.5 Knowledge Representation in Manufacturing Systems

Over the years the complexity of manufacturing system has increased, attracting research toward developing systems capable of rapid configurability and responsive to changing environment dynamics [53]. These new systems are able to adjust to changes in a timely manner improving process efficiency.

To achieve a system which can adapt to the environmental changes researchers have introduced Artificial Intelligence in manufacturing systems. These intelligent machines have knowledge of their surrounding needed for efficient working [54]. Knowledge can be represented in numerous ways; each method has its own pros and cons. New knowledge representation techniques are not only interpreted by machine but are also quite similar to human natural language.

AI researchers use different types of ways to conceptualize the environment in terms of objects, functions and their links with each other [54]. System designer develops knowledgebase on his view of the system and its environment. Conceptualization is abstract, simplified picture of the environment that one wants to represent in some sort of knowledge. Every knowledge-oriented system or agent is dedicated to some sort of con conceptualization directly or indirectly [55].

2.5.1 Ontologies

Ontologies are one of the most significant methods of representing and sharing knowledge. The use of ontologies for knowledge representation in different fields has proven very effective by reducing ambiguity and enabling sharing of knowledge [56].

Ontology is “an explicit specification of a conceptualization” [55]. The term “ontology” hails from the domain of philosophy which focuses on the study of being or existence. In computer sciences, ontology is a technical term representing an item developed for a specific purpose which allows shaping of knowledge about some domain [57].

Over the years several researchers in the field of AI defined ontologies in various ways. In general, ontology is clear detail description of “concepts in a domain of discourse” [58]. Ontology of a system can be represented by defining “a set of representational terms” [55]. Concepts are represented by classes in an ontology model. Each class has some properties which describe the characteristics and traits of the concepts. An ontology together with an individual instance in a class form a knowledge base. In developing, ontologies classes are the prime focus. To explain this concept of ontologies further, a class of dogs represents all dogs and specific dogs are the instances of this class. Each class can also have subclass in the above example the class of dog can contain a subclass of hunting dogs, pet dogs, guard dogs etc.

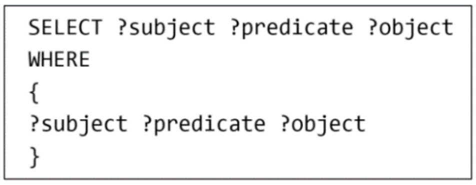
Ontologies are part of World Wide Web Consortium (W3C) standard for the semantic web, in which they are used to represent conceptual dictionary [57]. Ontologies help in data flow among the systems, provide services to respond queries, publish knowledge base, enable integration of systems and databases.

Using ontologies for modeling knowledge has become quite popular due to various benefits and features accompanying them. Ontologies develop a bridge for sharing information between human and software agents. This help to update new information form human directly without underlying conversion and similar manner a person can comprehend the data from the machine. Ontologies facilitate reuse of domain information making it effective in environments with collaborative working. Use of ontologies enables dynamic configuration in manufacturing systems. In previous systems, the conditions, the restriction was hardcoded in software programming. To understand and alter these systems require special expertise. Ontologies enable representation of these conditions and restriction in more formals simpler ways, making understanding and reconfiguration much easier. It's much easy to analyze data represented in ontologies due to more similarities to human-readable natural language.

2.5.2 Querying Ontologies

SPARQL is W3C recommended query language for Resource Description Framework (RDF). It a graph matching query language [59]. SPARQL helps to clarify cases where current semantics fail to provide information and interpretability. SPARQL enables value testing and outlining queries by RDF graphs [60]. Figure 19 shows general SPARQL query. A general SPARQL query contains three main components 'subject', predicate and 'object'.

SPARQL queries enable accumulation, negations and generating values by expressions[61]. SPARQL not only allow the user to access knowledgebase through queries but also make changes in the existing knowledge through SPARQL update [62].



```
SELECT ?subject ?predicate ?object
WHERE
{
?subject ?predicate ?object
}
```

Figure 19.*Generic SPARQL query [62].*

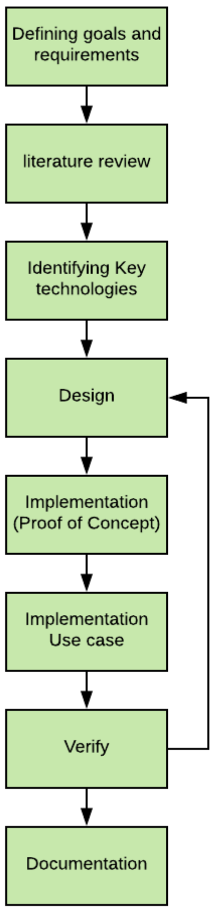
2.5.3 Ontology Editor

Protégé is an ontology editor which is freely available and opensource. Protégé provides user advance tools with ease of working for knowledge management. Protégé provides a user-friendly interface which can be adjusted according to user requirements.

Protégé is being used by users from a wide range of fields such biomedicine, manufacturing sector. It provides a strong user support in the form of tutorials, comprehensive documentation enabling efficient use of the software. Protégé also provides a web-based application enabling users to develop, modify and share models for collaborative working [63].

3. RESEARCH METHODOLOGY

The thesis aimed to develop a Cyber-Physical system which can be implemented in existing robots by employing cloud technologies to achieve automatic manipulation. The thesis involves set of steps which were adopted to achieve the desired targets of the thesis. Flowchart 1 represents steps which were adopted during the thesis work.


Defining goals and requirements
literature review
Identifying Key technologies
Design
Implementation (Proof of Concept)
Implementation Use case
Verify
Documentation

Flowchart 1.*Research methodology.*

Defining goals and requirements: The analysis of the problem and understanding the root cause is an integral part of this step. Existing systems were studied to understand their working environments and limitations. Detail problem statement and objectives were defined which can be found in the introduction section of the thesis. The goals and requirements defined in this step were the first preliminary direction towards the next phases of research and implementation of thesis work.

Literature Review: The goals and requirement defined in previous steps provide the base and direction for the literature review. The literature review includes journal articles, conference proceedings, books and other authentic resources available. The literature review helped in understanding relevant work done by other researchers, key technologies they used and the results they achieved. A literature review is a fundamental step of methodology in refining and converging objectives to more specific targets. The initial literature review provided the foundation for the development of thesis work.

Identifying key technologies: Literature review done in the previous step provided the bigger picture of the overall working of systems and group of technologies which can be utilized to accomplish thesis work goals. The key technologies were selected at this stage played an important role in the development of the system. These technologies were selected on the bases of the literature review keeping in consideration objectives and problem statement. During the selection of technologies, it was kept in consideration that they can be deployed on existing systems without any significant alterations.

Design: Once the key technologies were identified, the next important steps is the designing of the system. In designing phase there are two main components that needed to be addressed i.e. hardware and software. The interaction protocol between the technologies is the focus of design procedure. The operational principle of different technologies is usually different that require an intermediate interfacing protocol and hardware. Communication protocols developed will be used later to integrate different modules and technologies for working of the overall system.

Implementation (Proof of Concept): In the design phase, the preliminary communication protocols and software modules were developed but they require several practical iterations to become in more refined form for final implementation. Before going toward final implementation, a prototype is needed to be developed which will act as proof of concept. This proof of concept is small scale but utilize most of the key technologies. Proof of concept is very important in refining and eliminating preliminary errors in modules.

Implementation (use case): Once the preliminary protocols and technologies are tested the actual implementation proceeds. The actual implementation is the continuation of the proof of concept but at much wider scale. The final implementation is done by updating or changing existing protocols developed in proof of concept. Some new technologies were also introduced during final implementation, their interaction protocols are developed and integrated into final system.

Verify: After the final implementation is completed the verification of the system is next phase. Several test scenarios were created to analyze system performance and capabilities. The outcome of the system is analyzed in context to the objectives defined in the initial stage. If there are gaps in the expected performance, redesigning of the system is done to achieve the optimum results.

Documentation: When a working system is attained that can solve the problem defined initially, the next important step is to document the research and findings of thesis work for future reference. The documentation includes development stages, interaction protocols, integration of technologies and other essential information that would be helpful for the future readers. It also contains theoretical background that is essential for the reader to comprehend different modules of research work

4. APPROACH

This section describes the approach adopted for the implementation of the system. The section also provides architectural view and justification of technologies used to develop the system components to attain thesis goals of automatic manipulation.

4.1 System architectural view

This section discusses the architectural view and general functionality of the designed system. The system is designed in such a manner that it is able to achieve its objectives of automatic manipulation to accommodate changing workpiece without any manual alteration in the programming of the system. Figure 20 shows the building blocks of this designed system.

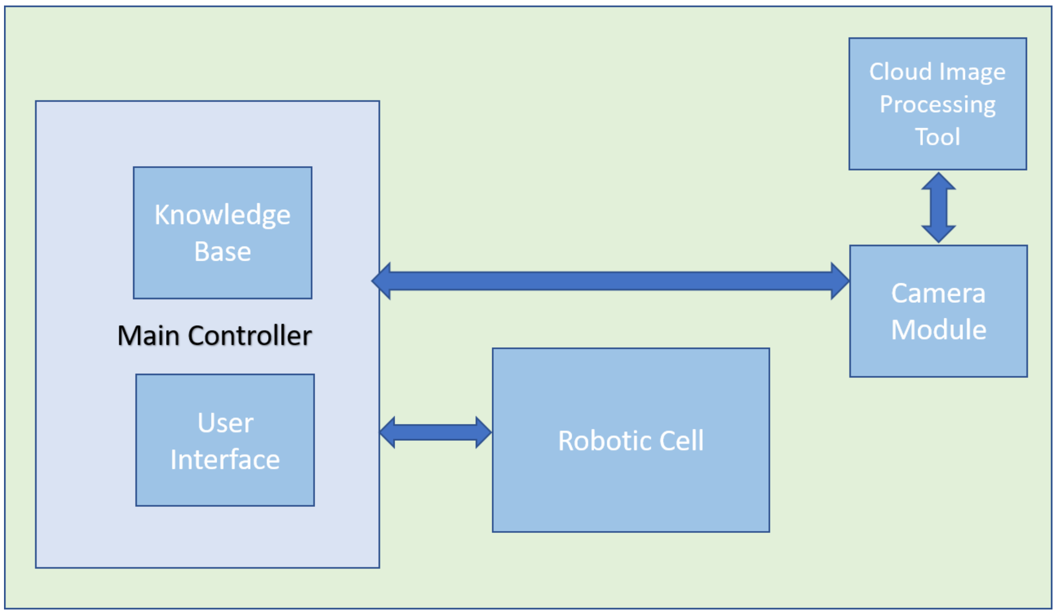
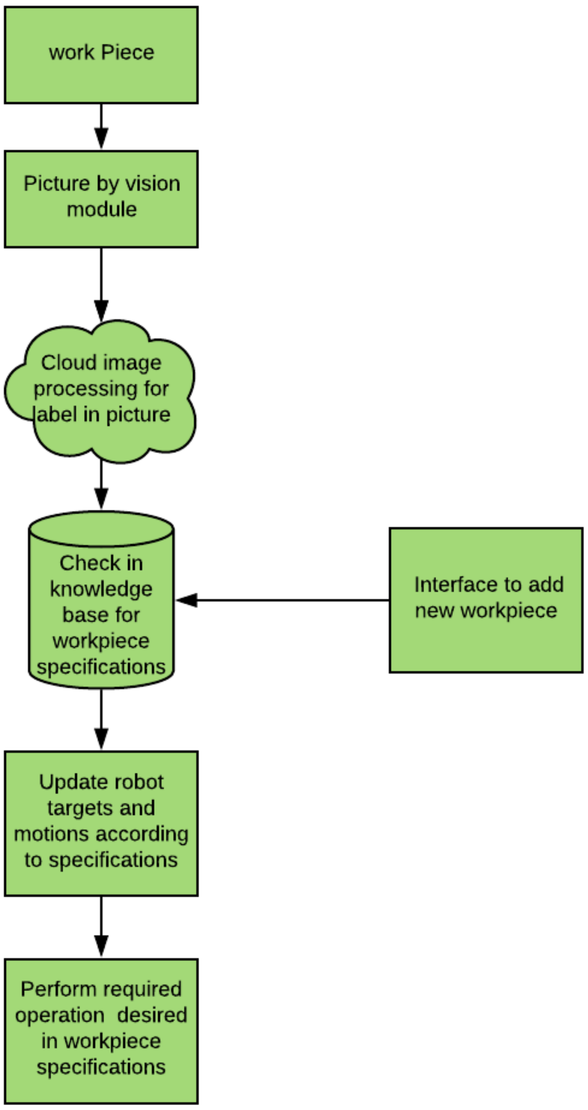


Figure 20. *Architectural view of the overall system.*

The above overall architecture of system works as described in the Flowchart 2. The system uses a vision hardware to acquire a picture of the workpiece. This picture is sent to cloud image analysis tools for detection of the label. Once the label is detected, specifications corresponding to the label are fetched from the knowledgebase and used to update programming to achieve updated manipulation mechanism. The robot performs the update in its manipulation mechanism and performs the operation associated with the workpiece in the knowledgebase. An interface accompanies the system to add new work piece specification into the knowledgebase.


work Piece
Picture by vision module
Cloud image processing for label in picture
Check in knowledge base for workpiece specifications
Interface to add new workpiece
Update robot targets and motions according to specifications
Perform required operation desired in workpiece specifications

Flowchart 2.*Overall working of the system.*

4.2 Key Technologies and Tools

The following sub-sections describe the key tools and technologies used to design the system. It also provides an overview and justification behind the choice of these specific tools for the implemented system.

4.2.1 ABB IRB140 Robotic Cell

The main aim of the thesis is to attain automatic manipulation in industrial robots for which a test environment was required to deploy the developed system and study the results. The final deployment was done on the robotic cell containing ABB IRB 140 robots. The cell contains a set of conveyors for delivery of workpiece. Figure 21 shows ABB robots used for the final implementation.

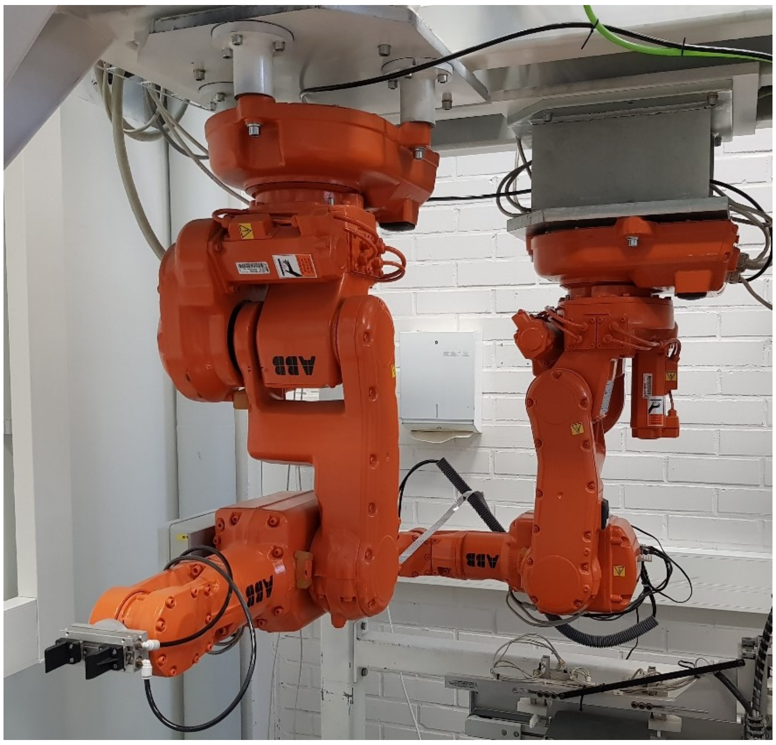


Figure 21.*ABB IRB 140 robots.*

ABB IRB 140 is powerful six axes robot. These robots are inverted and can handle 6 kg of payload [64]. IRB 140 robots support a vast range of communication platforms and programming environment. These robots can be programmed through FLEX pendent that comes with robot or with ABB Robot Studio software. IRB 140 robots are integrated with advanced safety features such as collision detection. The robots are operated by electrical power and the grippers are operated through the pneumatic mechanism.

The robots have a service port which is used for communication with the external environment through TCP/IP protocol and LAN port for interfacing with an external device for programming and monitoring.

The robotic cell contains ABB IRC5 controller that is responsible for all computations and communications. The controller contains digital input and outputs for developing control mechanisms for conveyors, actuators and other devices attached to the robotic cell.

ABB 140 robots are used as a test case, for development of system no specific feature found in ABB robots have been used. The module can easily be deployed on other robots by changing the syntax of programming. ABB 140 robots provide a test environment with socket communication, motion sequence and basic programming tools that are available in other industrial robots.

4.2.2 Main Controller

For the final implementation of the system, a processing hardware is used for hosting web interface and Fuseki server. This module is also responsible for communication with camera module and robot. For this purpose, two hardware were used in parallel one was embedded device Raspberry Pi 3 as shown in Figure 22 and other was a Laptop.

The developed system work in both environment. Intermediary devices like Raspberry Pi and laptop has been used to provide communication interface, support for wireless communication and processing of data between system modules which the current robot controller is unable to provide.

4.2.3 Raspberry Pi Camera Module

To detect different workpiece introduced into the system a camera module is required for taking the pictures of the label on the workpiece. For this purpose, Raspberry Pi 3 B+ module and Raspberry Pi Camera Module v2 were used. Figure 22 shows the camera module used for image acquisition.

Raspberry Pi 3 is 1.4GHz 64-bit quad-core processor, with wireless LAN, Ethernet and support for various programming platform [65]. Raspberry Pi 3 is Linux based architecture with a wide range of support for various computational algorithms. Raspberry Pi Camera Module v2 contains an 8-megapixel sensor capable of still photography and video recording [66]. The camera modules can be connected through a ribbon cable to the CSI port on the Raspberry Pi. These cables come in various length for this module 1-meter length of cable was used.

This module has been used in the system as it is potable, cheap and easily available. Raspberry Pi provides reasonable infrastructure for processing and transmission of data through suitable wired and wireless mediums. The module is low powered and miniature in size which makes its deployment to an existing system without any major alterations. There is significant open-source support available for integration and development of software infrastructure of the camera with Raspberry Pi.

Figure 22*.Raspberry Pi camera module.*

4.2.4 Knowledgebase

The knowledgebase is the key component that contains the specifications and other relevant information regarding workpiece. There are two main components of the knowledge base, the first tool to develop ontology model and second, a platform for deployment of the ontology model. For developing, knowledgebase ontologies were used and the model was developed in Protégé. The detail of protégé has been already discussed in the theoretical background section.

For deployment of ontology model Apache Jena Fuseki server[1] was selected. Fuseki is a SPARQL server which can operate on both window and Linux based operating systems. It provides REST-style SPARQL HTTP Update and SPARQL Query features [67]. Fuseki also provides a user interface for monitoring and performing basic tasks. Both development and deployment tools are opensource with supporting documentation.

Traditionally, there are two main platforms being used for data handling i.e knowledgebase and database. Michal Sir and his team have discussed both in detail two provide an understanding of both [68]. Databases like MySQL store information in form of tables but don't provide details what concept means and how they relate to each other.Ontologies provide a richer way of storing information that can be readily interpreted by humans and machine.SPARQL query language for ontologies enables advance and intelligent queries as compare to it counterparts. Ontology eases the handling and building of complex and large schema. It also allows the schema to be used in query time where its counterpart may result in complexity.

[1] https://jena.apache.org/

4.2.5 Google Cloud Vision API

Most of the existing old robot controller are not suitable for performing image analysis. To detect the label from the picture acquired Google cloud vision API has been selected to perform text recognition of the label attached to the workpiece.

Google Cloud Vision API uses complex machine learning algorithms to perform analysis of the image. It can detect various features in the picture such as text, face, landmark etc. [69].Cloud Vision API utilizes optical character recognition (OCR) to detect text in the images. The application works on REpresentational State Transfer (REST) which makes use of application much easier using standard computational resources and communication infrastructure.

Google Cloud Vision API has been used for the image analysis as it has vast support for integration to various programming infrastructures and hardware. The API can be used free for one year and nominal price after that. The processing speed of the API is significant and can handle multiple analysis queries. Each user has its dedicated credentials which help to maintain secure communication without loss of image to the unwanted user.

4.2.6 User Interface

User Interface provides all required infrastructure for the user to enter new data regarding workpiece. The user interface contains a user-friendly webpage where the user can select and enter different specification for the new workpiece. The user interface is integrated into the knowledge base, every time user enters a new entry in the interface it is automatically updated into the knowledgebase.

For developing user interface Hypertext Markup Language (HTML) is used which support various form features such as radio button, text field. The data entered by the user will be delivered to an application responsible for processing and updating knowledge base. HTML form is used to acquire the data from a web page and deliver to the processing application. Web page developed using HTML is simple, light in execution but provide sufficient features to be deployed for remote operations.

4.2.7 Programming Languages

Three main programming languages were used in the development of complete system i.e. Python, Node.js and RAPID.

Python[2] language has been used to develop the infrastructure of camera module i.e. communication to google cloud vision and to the main controller of the system. Python has

[2] https://www.python.org/

one if the largest support for embedded device with opensource libraries that ease the process of integration of various platform to devices.

All the computation and communication infrastructure is developed in *Node.js*[3] that includes knowledgebase update, acquisition of data from the knowledge base, processing of data and TCP/IP communication with robot.Node.js is used because of its support for developing systems working on web services and communication protocols. The express framework[4] is used to develop applications for communications between the desired modules. Node.js has opensource modules for various communication protocols which make the development of system much easier. This framework allows parallel processing of various web requests which make multiple communication possible without any additional effort.

Programming of ABB robots has been done in *RAPID* programming, dedicated language for ABB robots. The syntax is quite like structured text with abundant support provided by ABB. ABB Robot studio has been used to develop robot codes.

[3] https://nodejs.org/en/
[4] https://expressjs.com/

5. SYSTEM IMPLEMENTATION

This section discusses the actual implementation of the system. It discusses how the key technologies which were discussed in the approach used to develop a flexible manufacturing system that is capable of rapidly configuring itself whenever a new workpiece is introduced into the system by updating its manipulation mechanism. The section provides a detailed description of different system modules their software and hardware operations.

5.1 Proof of Concept

Before proceeding with implementation of the actual system, the small-scale prototype was developed to test following

1. Vision module hardware
2. Interaction of Google cloud Vision API with vision module
3. Preliminary communication protocols and data handling for cloud robotic system

The implemented system was documented as research publication[70].The implanted system contains two Lego robots, one with the camera module. The robot with camera module would scan its area by taking pictures of object it encounters in its way. The robot with the camera is called mapping agent uses Google cloud vision API to perform analysis of label on the objects and store in a supervisory agent. A web interface has been developed for a user to request any item that mapping agent has scanned. The lifter robot will pick the object and bring back to the user.

Figure 23 represents working and interaction of porotype designed to test some of the technologies. As shown in sequence diagram user select scan area option from the web interface, an HTTP post request is generated toward supervisory agent URL with a message to initiate the scan of the area. The supervisory agent is a laptop in this case but also can be an embedded device. This message is relayed by the supervisory agent to mapping agent through HTTP post request. The mapping agent situated in LEGO robot which starts to scan area by taking pictures of every object it encounters. The image is sent to google cloud vision for text detection, the response received by mapping agent is stored in supervisory agent through HTTP post request.

Once the mapping of the area is achieved Figure 23 also shows the interaction of lifter agent working in LEGO robot responsible for lifting objects. The user enters the product required in the web interface, an HTTP request is generated to the supervisory agent which look for the product in its storage. Once the product is found, as shown in sequence diagram the supervisory agent sends an HTTP request containing pallet number of the product. Lifter robot picks the product and returns to the user.

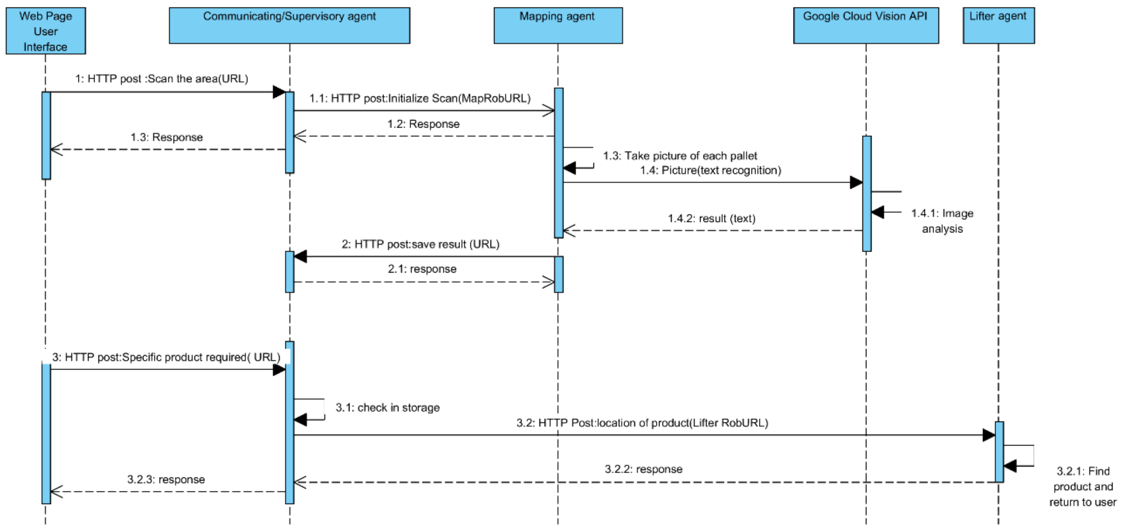

Web Page User Interface
Communicating/Supervisory agent
Mapping agent
Google Cloud Vision API
Lifter agent
1: HTTP post :Scan the area(URL)
1.1: HTTP post:Initialize Scan(MapRobURL)
1.2: Response
1.3: Response
1.3: Take picture of each pallet
1.4: Picture(text recognition)
1.4.1: Image analysis
1.4.2: result (text)
2: HTTP post:save result (URL)
2.1: response
3: HTTP post:Specific product required(URL)
3.1: check in storage
3.2: HTTP Post:location of product(Lifter RobURL)
3.2.1: Find product and return to user
3.2.2: response
3.2.3: response

Figure 23.*Sequence diagram for the developed prototype.*

5.2 System Modules

5.2.1 Knowledgebase Modeling and Deployment

Knowledgebase plays important role in the implemented system for storing the information regarding the workpiece. As discussed in approach section protégé has been used for development of the ontology model. Figure 24 represents the key properties defined in ontology model for the description of the workpiece. In the designed model there is only one class i.e. workpiece. Each workpiece can have properties which may include geometry, orientation, shape, length, width, height and radius.

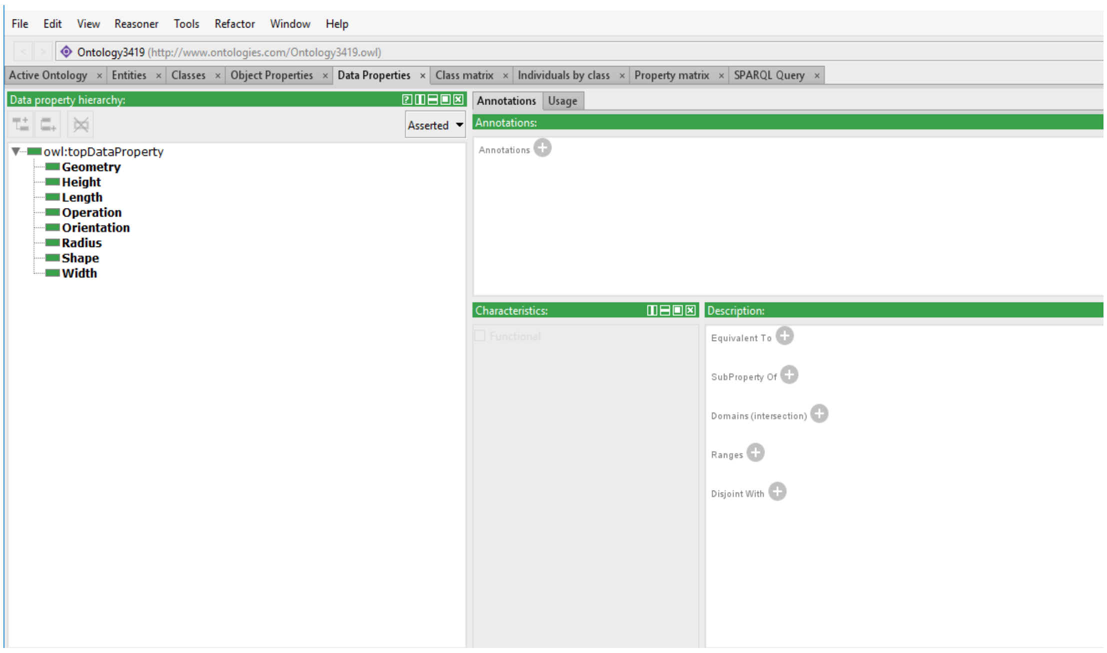


Figure 24.*Workpiece properties defined in Protégé.*

Geometry: contains information whether the workpiece is ‘regular’ or ‘irregular’. For implemented system shapes square, rectangle and circle are considered regular and all other geometries are considered irregular. The geometry property plays important role in performing the calculation for manipulation of robots.

Orientation: describes whether the workpiece is placed vertically or horizontally in accordance with the reference point of the delivery pallet. Figure 25 represents the vertical orientation and horizontal orientation of the workpiece in accordance to reference.

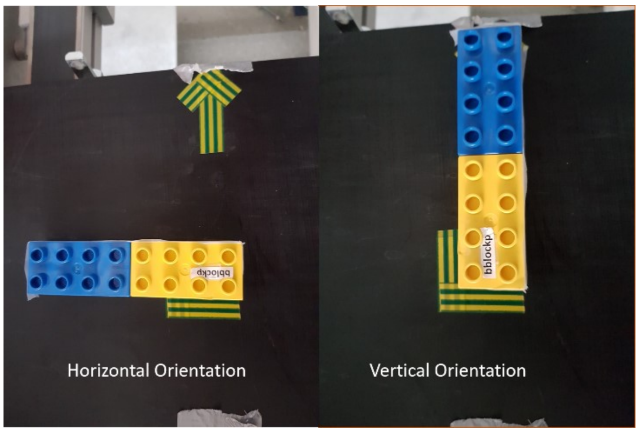


Figure 25. *Orientation of Workpiece.*

Shape: contains the information about the polygonal type of workpiece. The workpiece may be in circular, square or rectangular shape. This property is dedicated for a workpiece with regular geometry.

Operation: In an industrial environment different workpiece require different operations needed to be performed on them. In the current implementation, three main procedures have been included in implementation i.e. ‘discard’, ’place’ and ‘assemble’. In discard operation, the workpiece is picked and released at discarding position, wherein ‘place’ operation the workpiece is placed at a different position allocated for the task. Where in ‘assemble’ operation an object is assembled with a workpiece that has been delivered with the labeled workpiece.

Length, width, height and Radius: are properties defined to specify dimensions of relevant shapes of the workpiece. All these dimensions are in *mm*. Figure 26. Shows the different dimensions defined in the model to describe the workpiece.

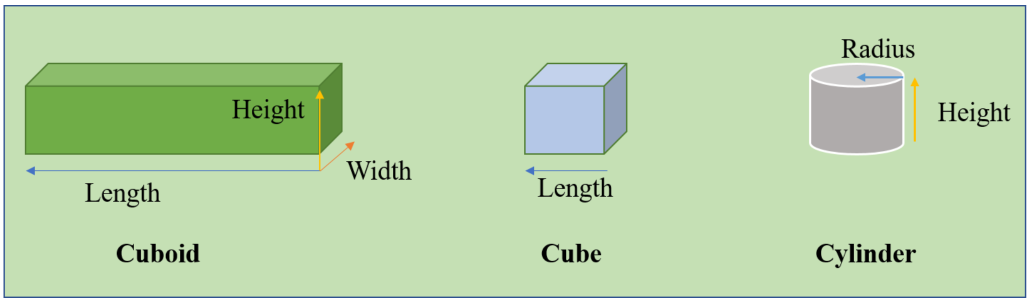


Figure 26.*Dimensions of different workpiece.*

It is important that workpiece is placed in accordance to reference marked on the delivery system for accurate calculations in accordance with dimensions stored in the knowledgebase. Figure 27. shows how the workpiece should be placed for dimension data in the knowledge base to be effective for automatic manipulation.

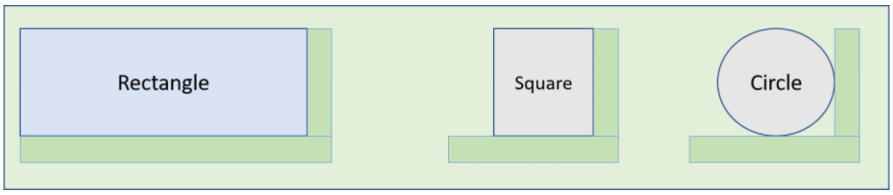


Figure 27.*Top view of a workpiece placed in accordance to reference.*

In case of an irregular object, the dimensions specified represent the pickup point for the robot. Figure 28 shows an example of an irregular object. The dot represents the pickup point for the robot. These dimensions help the robot to adjust itself accordingly to the dimensions specified in order to grip the object. In case of the irregular object, only basic functionalities are available. The length, width and height specified in knowledgebase are from reference point toward pickup point.

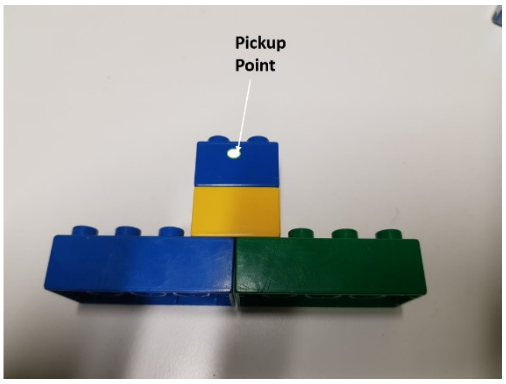


Figure 28.*An example scenario of the irregular workpiece.*

The knowledgebase uploaded into FUEKI server. In Fuseki server information can be retrieved and updated into knowledgebase through SPARQL queries. Once the application file of FUSEKI server is downloaded, a server can be initialized. Fuseki server initialized can listen to a specific port mentioned by the user for SPARQL queries and updates.

5.2.2 Web Interface and Update Module

A web interface has been developed for the system. This can be accessed through the internet for remote entry of data in the knowledgebase. Figure 29 shows the web interface of the system. The web interface is designed in such a manner that it can be fed with all data required for knowledgebase. HTML is the key technology used in the designing of the web page. This web page is hosted by update application designed in Node.js being executed by the main controller in the system.

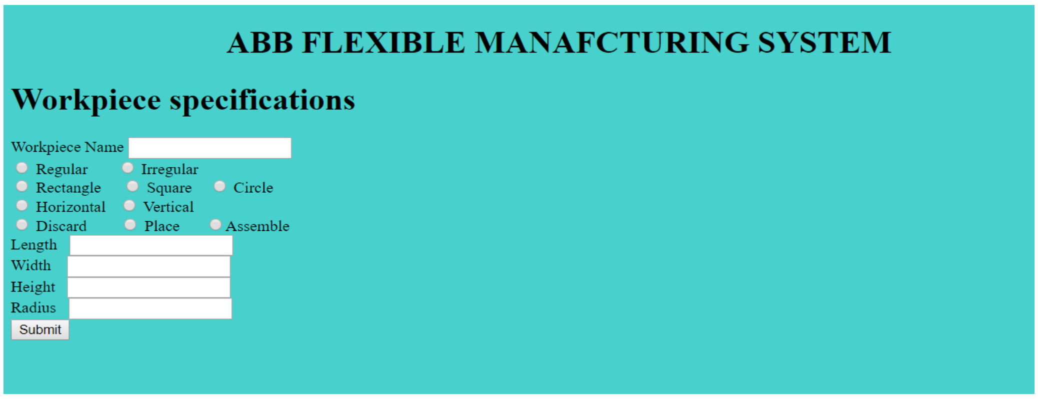


ABB FLEXIBLE MANAFCTURING SYSTEM

Workpiece specifications

Workpiece Name

○ Regular ○ Irregular
○ Rectangle ○ Square ○ Circle
○ Horizontal ○ Vertical
○ Discard ○ Place ○ Assemble

Length
Width
Height
Radius

Submit

Figure 29.*Web Interface.*

All the data entered by the user is acquired through HTML form. This form contains 'Radio Buttons' and 'Text Fields' for easing the process of data entry. Whenever a user enters data and press 'submit' button on interface all the data is sent to update application server application through HTTP post protocol. When form in initialized it contains three main fields as described in Code 1. The first URL of the application responsible for processing the form data, the second method of communication, in this case HTTP post and third is data type i.e. multipart form data.

```
<form   action="http://localhost:3001/api/form" method="POST" enctype="multipart/form-data">
```

Code 1.*HTML form Initialization.*

Once the data is sent to application responsible for handling the HTTP post request, its processing is started. NPM[5] 'Multer' package is used to extract information from form data send by the web page and store into variables for further procedures. Whenever HTTP request is received it contains form data encoded in request body which is extracted

[5] https://www.npmjs.com/

using NPM ‘Multer’ and ‘body-parser’[6] packages. The received data is decoded into JavaScript Object Notation (JSON) data as shown in Code 2.

```
{ wkname: 'block1',
  geometry: 'regular',
  shape: 'rectangle',
  orientation: 'horizontal',
  operation: 'place',
  length: '33',
  width: '22',
  height: '21',
  radius: '' }
```

Code 2.*Workpiece specification Object.*

The data send by the user is extracted by the update application, SPARQL update queries for FUSEKI server are generated autonomously in accordance with data received. There are four main generalized update queries depending upon four types of workpieces i.e. square, rectangle, circle and irregular. Each update query is responsible for creating new instant in knowledgebase according to the name entered by the user that will also be placed as a label on the workpiece. This query also contains all the specifications associated with that workpiece. Code 3 shows an update query generated for data received in Code 2 . These queries are generated automatically by the update application every time user enters new data in the web interface. The values are replaced by the new values decoded by the update application.

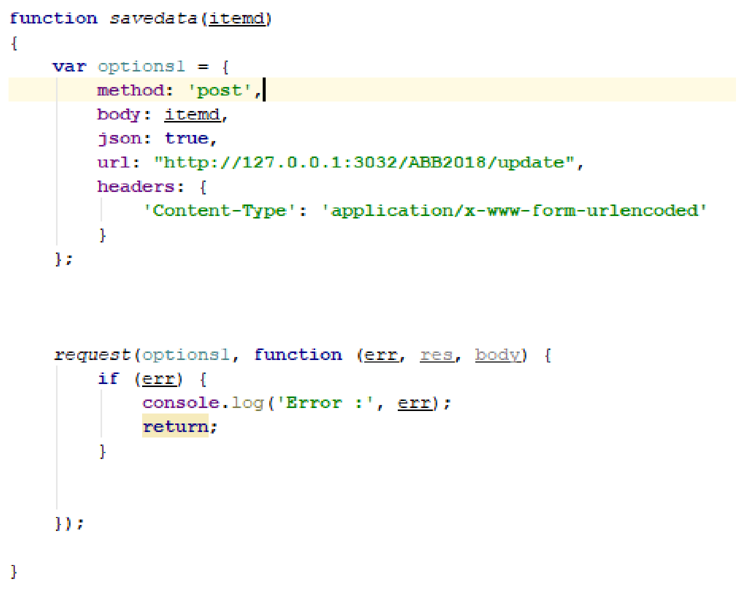
```
PREFIX aa:<http://www.ontologies.com/Ontology3419.owl#>
INSERT DATA {
aa: block1 a aa: Workpiece.
aa: block1 aa: Length “33”.
aa: block1 aa: Width “22”.
aa: block1 aa: Height “21”.
aa: block1 aa: Orientation “horizontal”.
aa: block1 aa: Geometry “regular”.
aa: block1 aa:Operation “place”.
aa: block1 aa: Shape "rectangle”.
}
```

Code 3.*Example update query.*

After formulating the queries, the next step is to create HTTP request for the update of data in knowledgebase in FUSEKI server. A ‘*savedata*’ function was developed which is

[6] https://www.npmjs.com/package/body-parser

called upon to perform the update through an HTTP request. Code 4 describe this function. The query generated is the input of this function. Other parameters i.e. Fuseki server URL, method and data type are same for every update query.



```
function savedata(itemd)
{
    var options1 = {
        method: 'post',
        body: itemd,
        json: true,
        url: "http://127.0.0.1:3032/ABB2018/update",
        headers: {
            'Content-Type': 'application/x-www-form-urlencoded'
        }
    };


    request(options1, function (err, res, body) {
        if (err) {
            console.log('Error :', err);
            return;
        }


    });
}
```

Code 4.*Savedata function for update of the knowledgebase.*

Figure 30 shows the sequence diagram of the interaction between the web interface and an update module. As shown in the diagram user can access the web interface through a web browser by using URL of the web page. When the user enters the URL in the browser an HTTP get request is generated to update application hosting the web interface. The user receives the interface where the data for the new workpiece can be added. Once the user submits the data, the data is received by update application through HTTP post request. This data is decoded and its update query is generated. This update request is sent to Fuseki server through HTTP post for the update of the knowledge base with new workpiece data. The data is updated in knowledgebase hosted by Fuseki server and user is sent back with acknowledgment about the success of the operation.

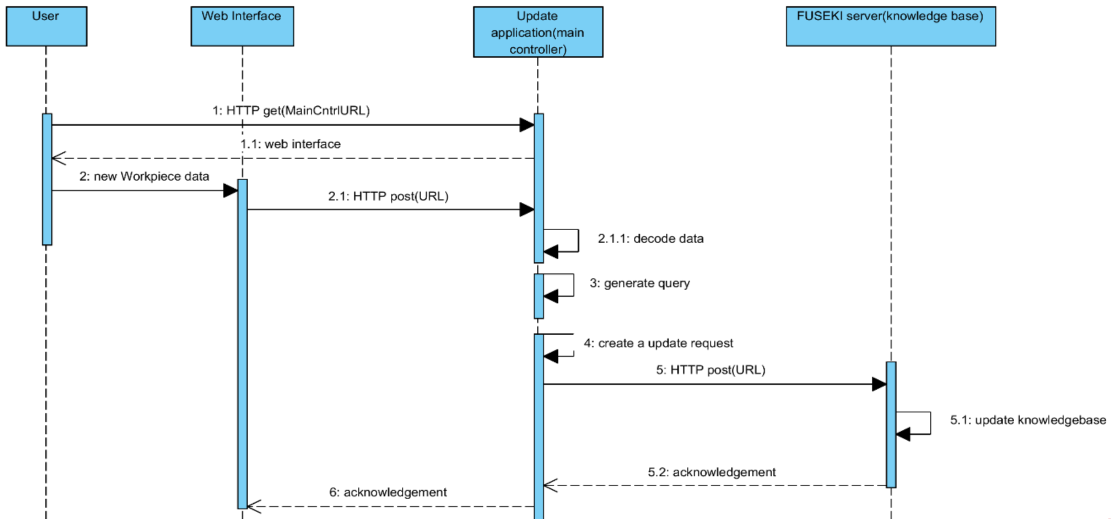


Figure 30.*Sequence Diagram for the web interface and update application.*

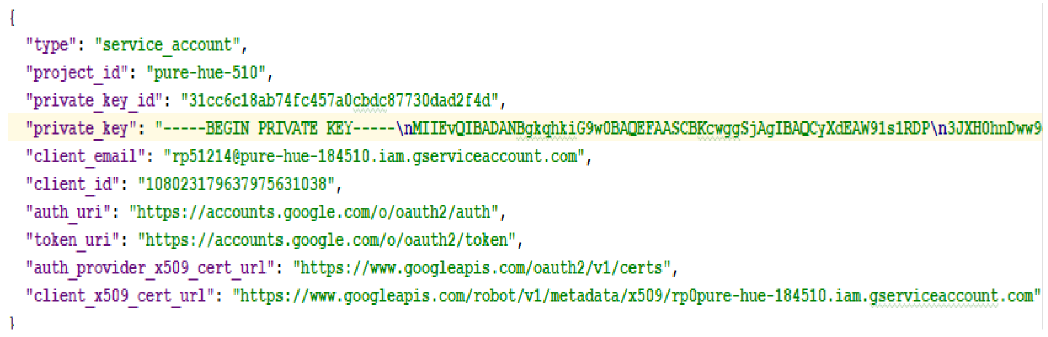
5.2.3 Camera Module

The camera module is dedicated hardware containing camera agent responsible for acquiring images of the workpiece and processing it through Google cloud Vision API for detection of label text. This module has three main functions

1. Acquiring pictures through the camera
2. Processing the images through Google cloud vision
3. Communication with the main central controller.

For acquiring the image through Raspberry Pi camera is used. For developing software infrastructure dedicated python libraries have been used which make the acquisition of an image with ease. A customized function is created, whenever this function is called upon the image is acquired through the camera and saved in 'home' directory of the controller. The acquired image is always saved with name 'image.jpg' in current implementation which can be changed as per requirement of the developer. Whenever a new image is attained through the camera the previous image is erased to avoid lack of storage.

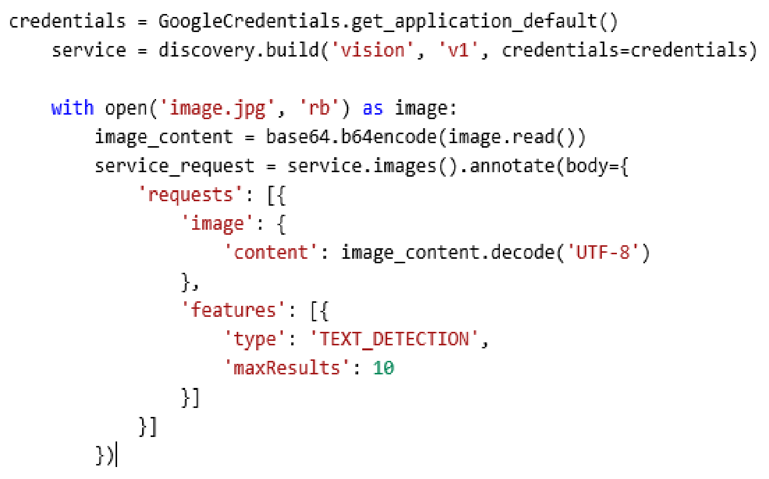
Next step of this agent is the use of Google Cloud Vision API. The user needs to create an account in Google cloud API for use of image analysis tools. When the account is created a user is provided with credentials which are needed to be used for data security purpose. These credentials must be loaded in Raspberry Pi as root user by executing the following in terminal export '*GOOGLE_APPLICATION_CREDENTIALS=filename.json'* where the 'filename' is the name of the file of credential stored in JSON format. Code 5 shows the credential these will varies depending upon the user individual data and type of computation required.



```
{
  "type": "service_account",
  "project_id": "pure-hue-510",
  "private_key_id": "31cc6c18ab74fc457a0cbdc87730dad2f4d",
  "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgIBAQCyXdEAW91s1RDP\n3JXH0hnDww9
  "client_email": "rp51214@pure-hue-184510.iam.gserviceaccount.com",
  "client_id": "108023179637975631038",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://accounts.google.com/o/oauth2/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/rp0pure-hue-184510.iam.gserviceaccount.com"
}
```

Code 5.*Sample Google Cloud Vision credentials.*

After loading credentials in the controller, the next step is to create a request for image analysis to the Google Cloud Vision API. This request in general consists of three main components i.e. image, credentials and type of analysis required. Code 6 shows the main part of request used by camera agent for image analysis. Whenever this request is generated, it uses python libraries in the background and opens the image saved in Raspberry Pi. This image is sent for text detection to API servers. For this request to work internet connectivity is must for the controller. In this request, the user can also adjust the depth of analysis required by changing '*maxResults*' parameter. More the value of the parameter more features of the image are returned.

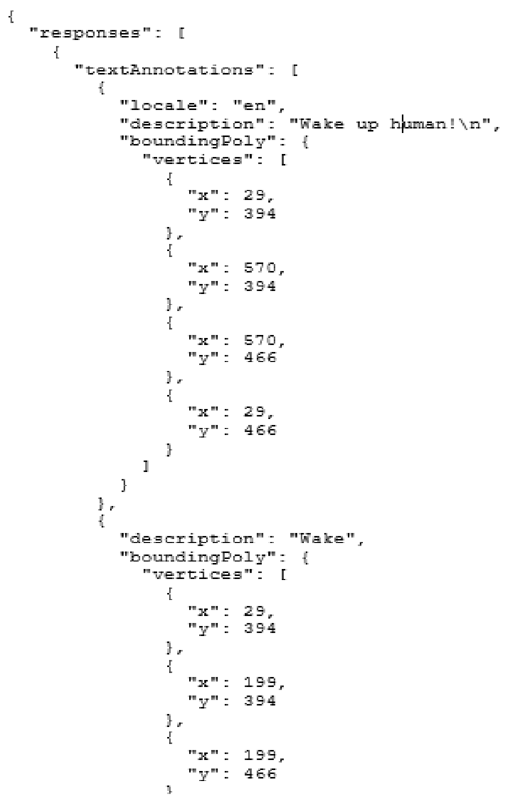


```
credentials = GoogleCredentials.get_application_default()
    service = discovery.build('vision', 'v1', credentials=credentials)

    with open('image.jpg', 'rb') as image:
        image_content = base64.b64encode(image.read())
        service_request = service.images().annotate(body={
            'requests': [{
                'image': {
                    'content': image_content.decode('UTF-8')
                },
                'features': [{
                    'type': 'TEXT_DETECTION',
                    'maxResults': 10
                }]
            }]
        })
```

Code 6.*Google cloud vision API request.*

When this request is executed through Raspberry Pi, the response received is a JSON object containing numerous data fields depending upon the depth of analysis desired in the request. Code 7 represents the part of the response received once the image analysis request is initiated. This response contains various information including the text detected, its position and individual word properties. The text label is extracted using JSON data extraction method.



```
{
  "responses": [
    {
      "textAnnotations": [
        {
          "locale": "en",
          "description": "Wake up human!\n",
          "boundingPoly": {
            "vertices": [
              {
                "x": 29,
                "y": 394
              },
              {
                "x": 570,
                "y": 394
              },
              {
                "x": 570,
                "y": 466
              },
              {
                "x": 29,
                "y": 466
              }
            ]
          }
        },
        {
          "description": "Wake",
          "boundingPoly": {
            "vertices": [
              {
                "x": 29,
                "y": 394
              },
              {
                "x": 199,
                "y": 394
              },
              {
                "x": 199,
                "y": 466
              }
```

***Code** 7.Google cloud vision partial response to the request.*

In camera module, there is an HTTP server which is continuously listening for incoming requests. Whenever HTTP post request with message '*pic*' is received the picture is acquired through the camera and processed through Google cloud vision API. The result received is sent back to HTTP post to the main controller of the designed system. Camera agent is independent of any physical bonding to the robot it can be interfaced to any robot and works on wireless medium for communication to the main controller.

Figure 31 shows the detail interaction of camera agent to system modules. The main controller initiates the image processing process by sending HTTP post request to camera agent URL with 'pic' message. As shown in sequence diagram the camera agent acquires a picture of the workpiece through the camera and send a text detection request to Google cloud vision API. The API sends the response with text detected to the camera agent. Camera agent decodes the message and sends the result through HTTP post request to the main controller. The main controller acknowledges on attaining results successfully and start the further process.

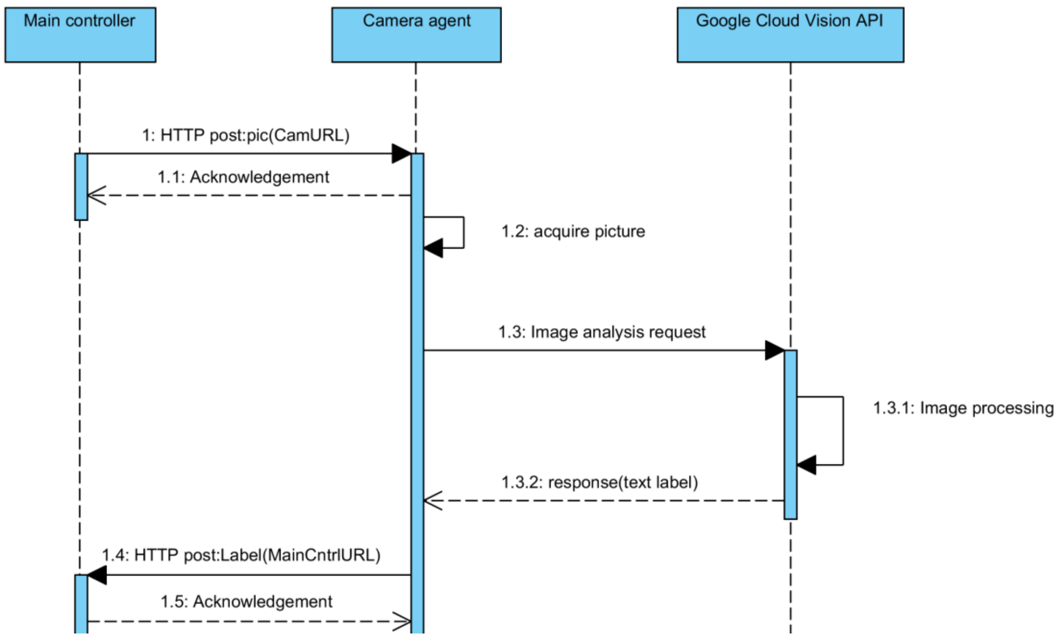


Figure 31.*Sequence Diagram for camera Agent.*

5.2.4 Main Controller

The main controller is responsible carrying following tasks

1. Establishing communication with the robot
2. Communication with camera agent.
3. Accessing data from the knowledgebase.
4. Encoding and decoding data for the robot and from knowledgebase respectively.
5. Hosting update and controller applications, knowledgebase and web interface.

The infrastructure of the main controller has been developed in such a manner that it can be deployed on both embedded device such as Raspberry Pi and on a windows laptop. There are two separate applications running in the main controller first one is update application that has been already discussed in the previous section, the other one is controller application responsible for completing first four tasks mentioned above.

In the first step of the application, a communication channel to the robot is established by connecting to TCP/IP socket server initiated by the robot controller. This channel established allows bidirectional communication between the robot and main controller. The connection is established between the service port of robot and the ethernet port of Raspberry Pi or laptop. For creating the socket client node.js 'Net[7]' platform has been used create a virtual device able to communicate with the robot through TCP/IP protocol. The

[7] https://nodejs.org/api/net.html

data can be sent and received in the form of the data stream. TCP/IP has been used because of most of the industrial robots this mode of communication.

Once the communication channel is functional the main controller waits for message '*camera*' from robot controller. This message is an indication that workpiece has arrived in working area of robot and robot need camera module for detection. Once the message is received the main controller initiate an HTTP request to camera agent for the acquisition of the image and its analysis. The detailed function of camera agent is explained in camera modules. The camera agent returns the main controller with results i.e. label by HTTP post request.

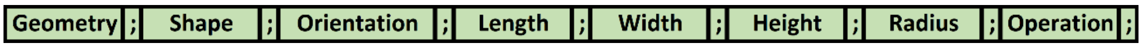
When the label of the workpiece is identified the main controller generates a query to access data of workpiece from knowledgebase in Fuseki server. There are four main type of workpieces with a different type of specification stored in the knowledgebase, to avoid latency by numerous queries one generalized query has been used. Code 8 shows the sample query generated to access workpiece data. The '*optional*' field has been used to avoid the error if that specific data is not found in the system. Instead of making four separate cases one general query fetch all possible data associated to any workpiece.

```
PREFIX aa:http://www.ontologies.com/Ontology3419.owl#
SELECT ?a ?b ?c ?d ?e ?f ?g ?h
WHERE{
{aa:block2 aa:Geometry ?a.
OPTIONAL { aa:block2 aa:Shape ?b.}
OPTIONAL{ aa:block2 aa:Orientation ?c.}
OPTIONAL{ aa:block2 aa:Length ?d.}
OPTIONAL{aa:block2 aa:Width ?e.}
OPTIONAL{aa:block2 aa:Height ?f.}
OPTIONAL{aa:block2 aa:Radius ?g.}
OPTIONAL{aa:block2 aa:Operation ?h.} }|
```

Code 8.*Example query to access workpiece specifications.*

All the information received is in binded form. Binding is the result of data handling between two applications developed in different environments. The binding function act as a bridge between two programming environments. In order to unbind a function '*extractor*' is developed with aim of unbinding all data which has been received and store it into respective variables depending upon the shape of workpiece.

All data relevant to the workpiece received by the main controller is packed in the special format shown in Figure 32.Each data specification is separated with a semi-colon; all the eight specifications are packed into a single data string. Fields not available are filled with zero value by the controller application. This data string is transmitted to robot controller through TCP/IP socket channel established previously by the robot controller. The data is sent in form of the continuous stream with string format.



Geometry	;	Shape	;	Orientation	;	Length	;	Width	;	Height	;	Radius	;	Operation	;

Figure 32.*Data Packet for workpiece specifications.*

5.2.5 Robot Controller

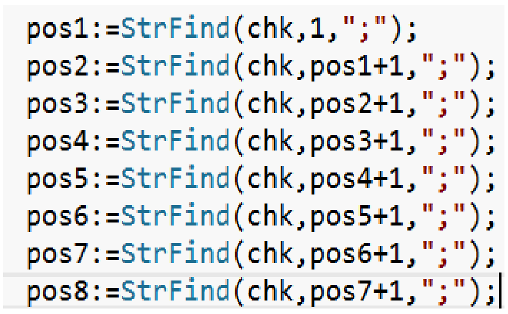
Robot controller has following main tasks:

1. Establish communication with the main controller
2. Parsing of the data string
3. Generating motion sequence
4. Executing motion sequence

When robot controller is activated it creates TCP/IP socket server associated with its service port. The TCP/IP server waits for incoming connections. Once a client connects to the socket created, a bidirectional communication channel is established.

Robot controller waits for digital input used for detection of the pallet. Once the pallet arrives at desired position robot controller activates set of motion where camera attached to robot reaches a position where the workpiece is in focus. Robot controller sends '*camera*' message to the main controller through socket communication. Robot controller starts to wait for the results from the main controller about the detected workpiece.

A data packet is received from the main controller which is already defined in Figure 32. A dedicated function is responsible for decoding the data sent by the main controller and retrieving the workpiece specification. In the first step as shown in Code 9, the function finds all the positions of the semicolon in the data packet. These semicolons were intentionally embedded in the data string to identify different fields. By using '*strFind*' function the position of eight semicolons is found and stored in a dedicated variable. These are numeric values stating each semicolon position in whole data string starting from first to the last semicolon.



```
pos1:=StrFind(chk,1,";");
pos2:=StrFind(chk,pos1+1,";");
pos3:=StrFind(chk,pos2+1,";");
pos4:=StrFind(chk,pos3+1,";");
pos5:=StrFind(chk,pos4+1,";");
pos6:=StrFind(chk,pos5+1,";");
pos7:=StrFind(chk,pos6+1,";");
pos8:=StrFind(chk,pos7+1,";");
```

Code 9.*Data parsing: decoding fields.*

Once the positions of the semi-colon are known the next step is to identify each specification stored in each field of the data string.'*StrPart*' function has been used to find each part in between the two semicolons and store in the variables as shown in Code 10. It's important to notice here that all the data received is in string form but to retrieve dimensions these fields are converted into numeric by using '*StrToVal*' function.

```
geometry:=StrPart(chk,1,pos1-1);
shape:=StrPart(chk,pos1+1,pos2-pos1-1);
orientation:=StrPart(chk,pos2+1,pos3-pos2-1);
temp:=StrToVal(StrPart(chk,pos3+1,pos4-pos3-1),length);
temp:=StrToVal(StrPart(chk,pos4+1,pos5-pos4-1),width);
temp:=StrToVal(StrPart(chk,pos5+1,pos6-pos5-1),height);
temp:=StrToVal(StrPart(chk,pos6+1,pos7-pos6-1),radius);
operation:=StrPart(chk,pos7+1,pos8-pos7-1);
```

Code 10.*Data parsing: data extraction.*

After the successful retrieval of data, the shape of the workpiece is identified i.e. square, rectangle, circle or irregular and respective functions are called to proceed with planning and execution of motion sequence.

Once the shape of the object is known the gripper of robot move to a reference point i.e. a place where the boundaries of workpiece just start. If the shape is regular it has been considered that the mass is evenly distributed. For smooth balanced pickup, the gripping point of the workpiece is its aligned with the line of symmetry for the regular object. For planning motion robot targets are changed accordingly to the workpiece. Robot target is composed of several fields which are stored in the form of an object as shown in Code 11.Each field in this robot target can be accessed independently, manipulated and stored back to update the motion sequence. For example, if translational value z is needed it can be attained by '*robtarget.trans.z*'.

```
< dataobject of robtarget >
      < trans of pos >
         < x of num >
         < y of num >
         < z of num >
      < rot of orient >
         < q1 of num >
         < q2 of num >
         < q3 of num >
         < q4 of num >
      < robconf of confdata >
         < cf1 of num >
         < cf4 of num >
         < cf6 of num >
         < cfx of num >
      < extax of extjoint >
         < eax_a of num >
         < eax_b of num >
         < eax_c of num >
         < eax_d of num >
         < eax_e of num >
         < eax_f of num >
```

Code 11.*Robot target object.*

The reference robot target is updated in sequence as described in Flowchart 3. In the first step, the gripper arrives at the reference position as shown in Figure 33. First, the z position of the gripper is adjusted accordingly to the height of workpiece. Then the gripper is brought about the gripping position by adjusting the y-translation position in respect to length or radius when orientation is vertical. The gripper will move a distance equal to the radius or half of the length in case of the rectangle and square object. The gripper will lower itself to the original translational ‘Z’ position to grip the workpiece. In case of an irregular object, the dimensions provided by the user are just the pickup point for the workpiece.

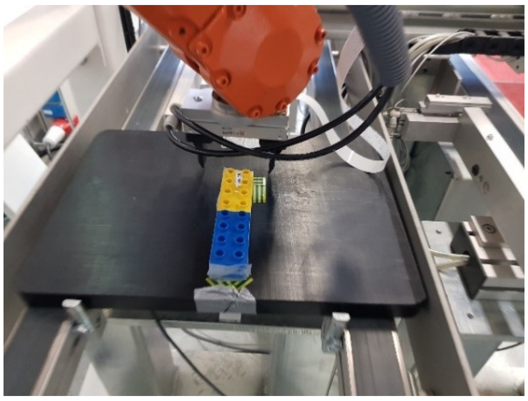


Figure 33. *Workpiece and gripper arrangement for vertical orientation.*

The above mechanism is adopted in case of the vertical orientation of workpiece but if the workpiece orientation is horizontal then the gripper rotates itself and go to new reference position depending upon the orientation of the workpiece. This feature has been included to facilitate user without altering the dimensions of workpiece but just changing the orientation of the workpiece in the knowledgebase. Figure 34 show workpiece arrangement during vertical orientation. The motion planning is similar i.e. first change height then linear motion but in x-axis rather then the y-axis.

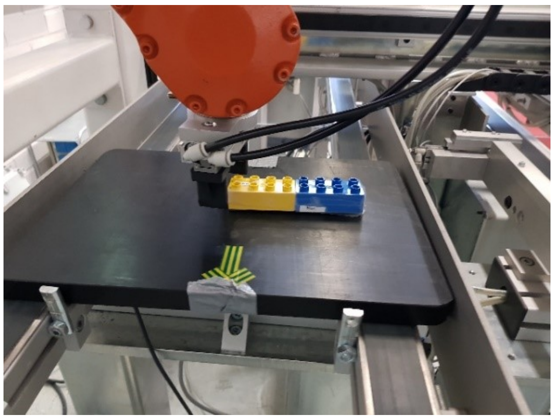
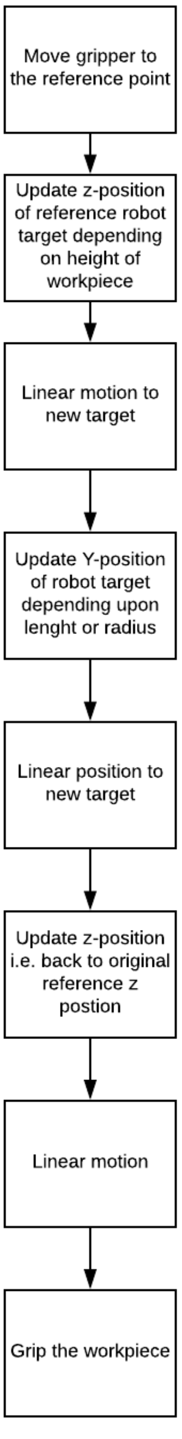


Figure 34. *Workpiece and gripper arrangement for horizontal orientation.*

There are three operations which have been integrated into the system to provide guidelines how different operation can be performed on different workpieces in a single cell. These operations include ‘place’, ’discard’ and ‘assemble’.

In case of ‘place’ and ‘discard’ operations the robot check in the data received for the workpiece. Two locations are predefined and the workpiece is placed on respective 'place' and 'discard' positions. Assemble operation is a special case use case implemented with prime focus of operation than automatic manipulation. Whenever a workpiece has assembled operation in its knowledgebase data the workpiece parallel to the labeled workpiece is assembled with the other piece and placed at 'place' location. This feature has been implemented to provide how different operations can also be done one workpiece using the single robot.


Move gripper to the reference point
Update z-position of reference robot target depending on height of workpiece
Linear motion to new target
Update Y-position of robot target depending upon lenght or radius
Linear position to new target
Update z-position i.e. back to original reference z postion
Linear motion
Grip the workpiece

Flowchart 3.*Generalized motion planning sequence for vertical orientation.*

5.2.6 Overall Communication Diagram

Figure 35 shows an overall sequence diagram of the interaction of system main controller to different system modules. The diagram represents the communication protocol and intermediary steps performed to achieve the system goals. When Robot controller is powered on and the system modules are executed the robot creates a socket server and wait for the client to connect in this case main controller. In next step, robot waits for the pallet containing workpiece. Once the pallet arrives, the robot arrives at camera position where the workpiece is in focus of the camera. Robot controller sends 'camera' message to the main controller through socket communication. The main controller sends HTTP post request to camera agent with '*pic*' message to initiate camera module.

Figure 35 also shows camera module and agent interaction to system modules as explained in Figure 31. The camera module takes pictures of the workpiece and sends to Google cloud vision API for test detection. The result received by camera module is sent to the main controller through HTTP post request. Once label of the workpiece is received a query is generated to access knowledge base for specification through HTTP post request to Fuseki server. The data received is extracted and packed into a special format. The data packet containing specifications of the workpiece is sent to the robot through socket communication. The robot, once receives the data packet it parses the data and retrieves workpiece information to update manipulation mechanism.

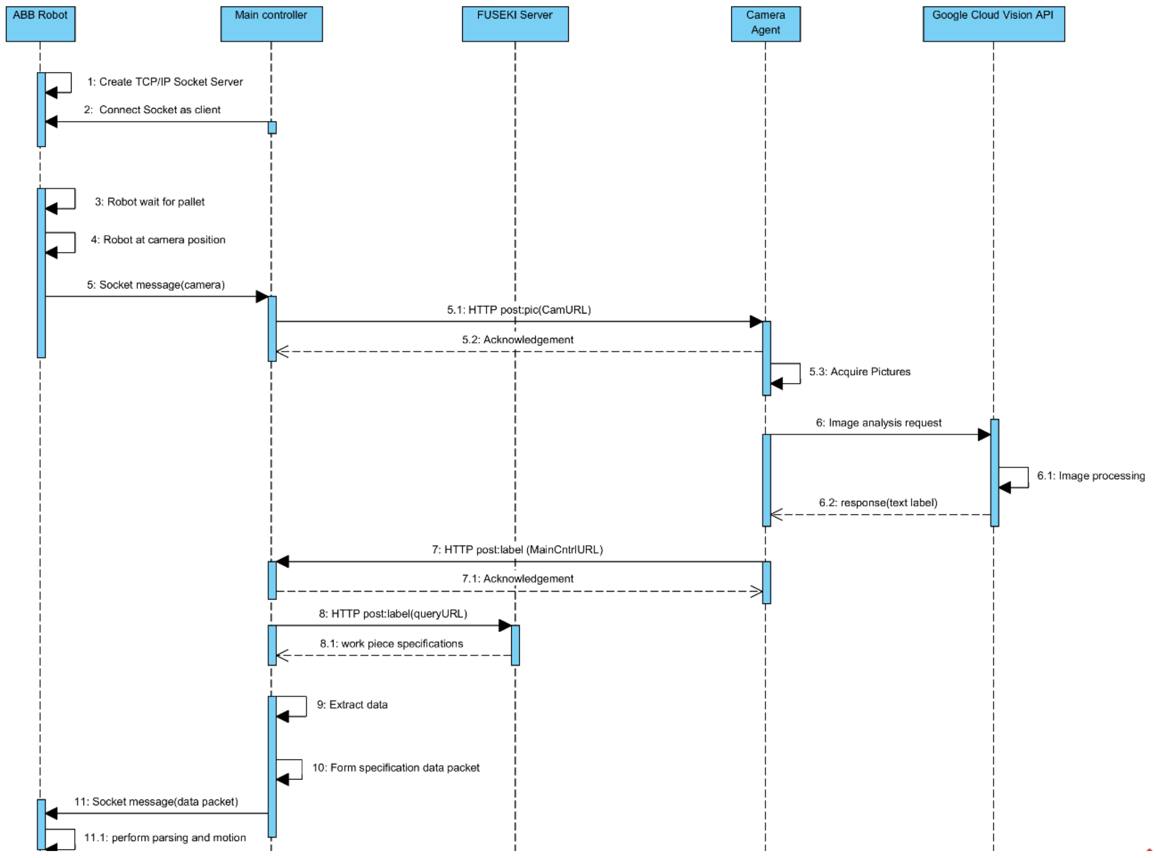


ABB Robot
Main controller
FUSEKI Server
Camera Agent
Google Cloud Vision API
1: Create TCP/IP Socket Server
2: Connect Socket as client
3: Robot wait for pallet
4: Robot at camera position
5: Socket message(camera)
5.1: HTTP post:pic(CamURL)
5.2: Acknowledgement
5.3: Acquire Pictures
6: Image analysis request
6.1: Image processing
6.2: response(text label)
7: HTTP post:label (MainCntrlURL)
7.1: Acknowledgement
8: HTTP post:label(queryURL)
8.1: work piece specifications
9: Extract data
10: Form specification data packet
11: Socket message(data packet)
11.1: perform parsing and motion

Figure 35.*Sequence Diagram of system operation.*

6. RESULTS AND DISCUSSIONS

This chapter discusses the results achieved once the implementation discussed in section four is achieved. Several test cases were executed to analyze system performance, gaps and efficiency. The designed system has several modules which involve software and hardware components. These modules were tested in an individual manner and as a whole system. Following are the results and related discussions regarding system modules and working.

6.1 Vision Module for Workpiece Detection

The vision module used in the system is sufficiently capable of being integrated into the existing robotic systems. The hardware used is portable small which makes it deployable on most of the popular robot grippers. The availability of long cables i.e. 1m and 2m between the camera and the Raspberry Pi controller makes integration easy to the different size of robots with variable arm lengths. The module works on the wireless network; the connection strength has been found quite stable for bidirectional communication. Figure 36 shows the placement of the camera on the gripper and Raspberry Pi attached to the robot arm.

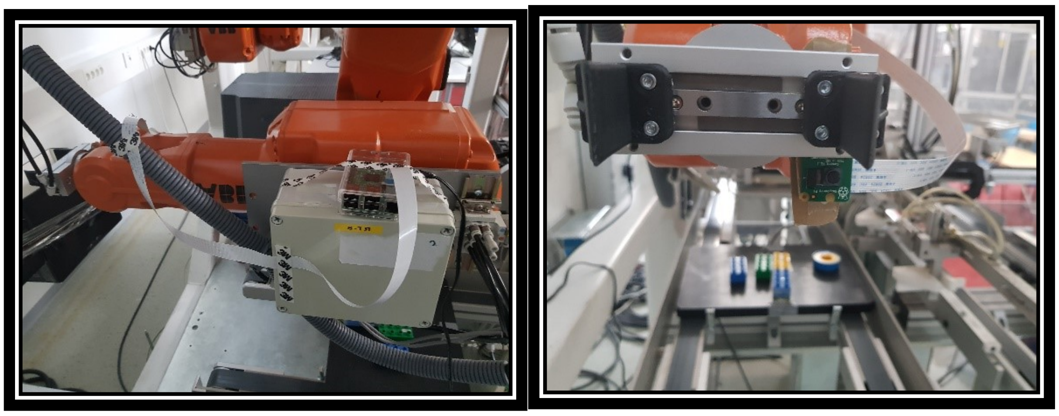


Figure 36.*Camera module attached to ABB robots.*

The camera was found quite good as compared to cheap price but it is affected by the availability of light and the font of the label. Several different fonts were tested the most optimum results were found with labels printed in ‘Calibri (body)’ font with size ‘16’. The printed labels can be seen on workpieces in Figure 38. During the testing change in light often resulted in some unexpected results. These results include some additional characters with original label text or even ‘no response’ from API.

During the testing of the system, Google Cloud Vision API has been found quite effective in the analysis of the text. The API has its complete analysis platform which provides users with the facts and figures regarding the performance of API. In a testing period of 30 days, 90 requests were generating by the system with a median latency of 196 ms as shown in Figure 37. The maximum latency generated was 471 ms. It is important to notice here the latency reading is related to the processing time taken by API in preparation of the results from the time it received the request. The API during the testing never generated any error. This includes error once the request has reached APIs servers i.e. during image analysis. These facts are clear representation how this API can be used in industrial scenarios without any major latency issues if the supporting infrastructure has sufficient processing capabilities.

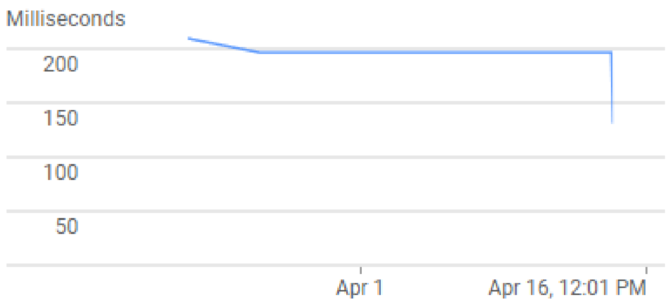


Figure 37.*Median latency graph generated by Google cloud vision API.*

6.2 Communication Protocol for Cyber-Physical System.

The infrastructure developed uses standard communication technologies i.e. HTTP protocol for sharing of information between camera module and knowledgebase. The architecture is based on wireless medium and distributed architecture. Data packet developed for communication with robot contains all the field that is required for automatic manipulation, the user can increase information without any difficulty. A generic parser module is working in the robot, this module uses general tools of string manipulation to extract data from the data packet.

For working of the system, the devices must be in the same network and connected to the internet for access to the cloud. The communication to the cloud is secure due to Google credentials which are used before initiating the request.

The ontology model in Fuseki server is situated in the main controller, where the query module is also present. Both applications are working on the same controller with software communication between the applications. Whenever camera detects label it sends it to the main controller application which is responsible for querying the item from Fuseki server also situated in the same controller. The latency between the query and the results is dependent upon the computational power of controller. In the current implementation, it was found that an approximate average time of 1.2 seconds was taken for the process of query, the response from Fuseki and extraction of data from the received binded data.

6.3 Automatic Manipulation

The focus area of the testing phase was automatic manipulation of the robotic cell for a different type of workpieces. A number of workpieces were restricted due to the gripper of the ABB cell in hand. Different type of samples workpieces was used to test system working which include rectangular, circular and irregular workpieces. Each workpiece was attached with a label corresponding to the data stored in knowledgebase as shown in Figure 38. Testing of automatic manipulation mechanism was done in steps to avoid any collision and damage to robots.

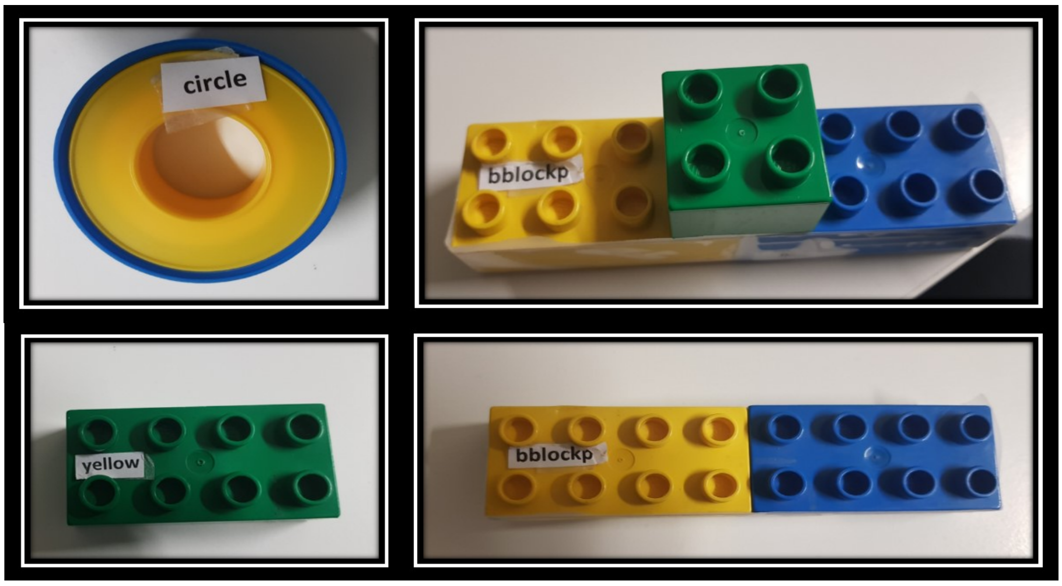


Figure 38.*Sample workpieces.*

In case of regular objects, the robot was successfully able to grip the workpiece through the data retrieved from the knowledge base. Each time when workpiece was changed robot changes its reference robot targets by using the data packet sent by the main controller. In circular workpiece, the gripper adjusts its position precisely accordingly to the radius of the workpiece. In testing with a current gripper, the geometry of gripper and surface of workpiece let to some frequent slipping of the workpiece from fingers of the gripper.

In the situation of an irregular object, the data entered in knowledgebase is the pickup point of the robot. The gripper rises to the height of the pickup point and lowers again to grip the workpiece after horizontal movement. In case of the irregular workpiece, two scenarios were tested as shown in Figure 39. In the situation of case A, the gripper rises to the height of pickup point and then travel to pick up point through horizontal motion. In case B where there are features with a variable height that results in hindrance in the approach of pickup point hence resulting in collision of the gripper to the workpiece.

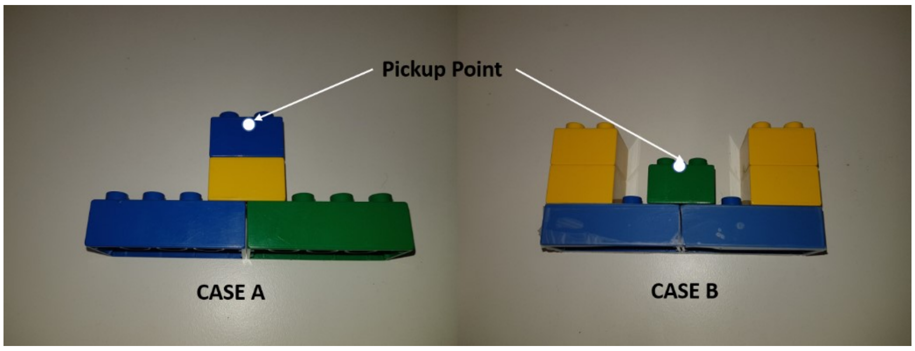


Figure 39.*Irregular workpiece scenarios.*

6.4 Workpiece Based Operations

Another key feature this system provides is associations of different operations to workpieces. Three operations place, discard and assemble were also tested. Two positions were allocated in which robot would pick different workpieces and put them in discarding or placing boxes. It's important to note that operations are based on robot infrastructure, tools and overall hardware. In the current implementation, the operation has been designed in such a manner to provide proof of concept how a single cell can perform different operations depending upon the operation assigned in the knowledge base. Figure 40 shows the two attached boxes for the place and discard operations of the robot.

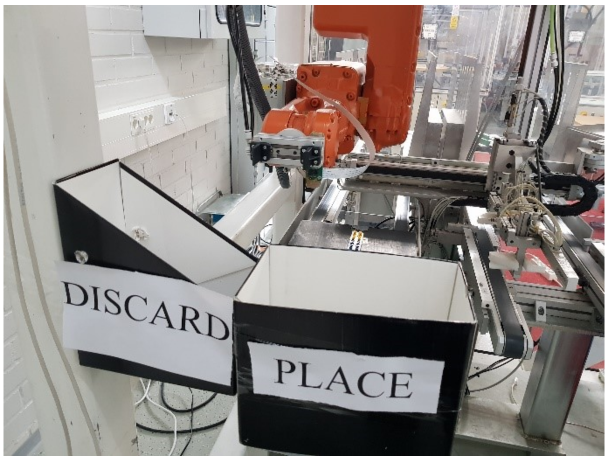


Figure 40.*Place and discard position of the robot.*

In this implementation, an additional scenario has been implemented i.e. assembly operation. In this scenario, as shown in Figure 41 the labeled workpiece is in the focus area of the camera. Whenever the user wants to perform assembly two workpieces will be introduced in the system. In this scenario, the unlabeled workpiece will be picked by the robot and placed on the labeled workpiece. The assembled workpiece is picked and placed in 'place' box attached to the robot. The operation provide basic platform how the assembly can be performed depending upon workpiece information in knowledgebase.

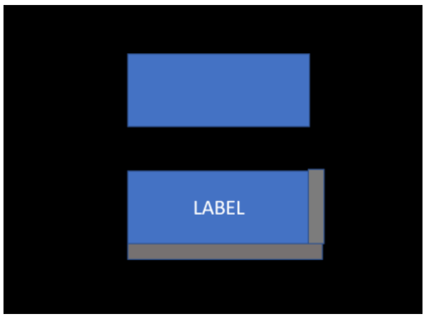


Figure 41.*Assembling scenario.*

6.5 Main Controller

As discussed in approach and system implementation sections the main controller of the system was tested in two computing devices i.e. Raspberry pi and laptop. The laptop provides more autonomy of usage due to the user-friendly interface. It computation power is sufficient and reliable but the main back draw is portability in industrial scenarios. Whereas Raspberry pi is small, support various development infrastructures but during practical scenarios, it has been observed that it often fails to perform the task in a reliable manner. Frequent disruption is seen while testing. The more efficient embedded device is required for the main controller which can support computations without any disruption.

7. CONCLUSIONS

This thesis presents an approach to integrate cloud Robotics in industrial robots for achieving automatic manipulation. The implemented system presents proof of concept how new workpiece can be introduced into the system without alteration of programming of the robot.

7.1 Summary

This thesis presents an approach to integrate cloud Robotics for achieving automatic manipulation. Automatic manipulation is key to developing FMSs. The implemented system presents proof of concept how new workpiece can be introduced into the system without alteration of programming of the robot.

An ontology model has been developed by using protégé to hold basic specifications and attributes related to workpieces. This ontology model is deployed using Fuseki server which can respond to SPARQL queries and update through an HTTP request.

The designed system contains a web interface for including specification of the new workpiece in the system. This web interface is connected to ontology model in Fuseki server through an update application. The interface can be accessed by the user remotely to update knowledgebase. Each workpiece is stored under special name allocated by the user, which is also placed as a label on actual workpiece

The thesis provides guidelines and infrastructure to deploy vision capability to industrial robots by using cloud image processing resources to overcome resource constraints on factory shop floor. The vision hardware used in the implemented system has been selected for its interoperability with existing systems. Google cloud API has been used to perform cloud-based image analysis of pictures. The API has been used to read labels attached to workpiece corresponding to their specification stored in the knowledgebase.

The communication protocol defined for communication between the developed cyber-physical system is based on standard communication technologies i.e. HTTP and TCP/IP socket communication. The data corresponding to the workpiece label is fetched from knowledgebase and packed in the data string. This data packet contains standard eight fields defining workpiece. A parser module has been developed in the robot to decode these messages and utilized them to update robot target points.

The designed system is not only able to handle workpiece and alter its programming runtime but also allow to associate operation to different workpieces depending upon user requirement.

7.2 Future Work

The current designed system provides the basic platform and communication structure for the implementation of cloud-based robotic FMS. This developed system although achieved its most of objectives but there are still several areas which can be explored to develop more functionalities in the system.

The current implementation is working on a limited number of workpieces, by representing workpiece specification and geometry with more attributes, the designed system can accommodate more types of workpieces.

The system has been tested on with Raspberry Pi controller which was not able to support the desired computation in a reliable manner. The developed infrastructure can be deployed in the controller with more processing power and consistency in operations that will help to achieve robustness for industrial environments.

Although simple operations have been performed on the bases of data from knowledge-base it can be enhanced to a system which can use automatic manipulation concept developed in the thesis to perform dynamic assembly of workpieces. The data received can be used to manipulate the robot target in such manner that assembly of various items can be achieved. To achieve this type of system the test environment should support precision and able to retain its calibrations.

The camera module used in this system serve the task at hand but different modules are needed to be tested to enhance the working efficiency to avoid failure of systems in case of wrong image analysis. The focus of the camera is often affected by the large variations in the height of workpiece this can be explored further by dynamically changing focus range with respect to the height of workpieces.

The ontology model developed for workpiece specification has been deployed locally but in future, it can also be deployed on the cloud that will reduce computation load on local hardware hence making the system more efficient and responsive in industrial scenarios. Cloud model can be used by multiple robotic cells instead of integrating separate model for every robot.

REFERENCES

[1] Design of reconfigurable manufacturing systems, in: Journal of Manufacturing Systems, 2010, pp. 130-141.

[2] H.A. ElMaraghy, Flexible and reconfigurable manufacturing systems paradigms, International Journal of Flexible Manufacturing Systems, Vol. 17, Iss. 4, 2005, pp. 261-276. Available (accessed ID: ElMaraghy2005): https://doi.org/10.1007/s10696-006-9028-7.

[3] J. Browne, D. Dubois, K. Rathmill, S.P. Sethi, K.E. Stecke, Classification of flexible manufacturing systems, The FMS magazine, Vol. 2, Iss. 2, 1984, pp. 114-117.

[4] M. Rüßmann, M. Lorenz, P. Gerbert, M. Waldner, J. Justus, P. Engel, M. Harnisch, Industry 4.0: The future of productivity and growth in manufacturing industries, Boston Consulting Group, Vol. 9, 2015.

[5] H. Kagermann, J. Helbig, A. Hellinger, W. Wahlster, Recommendations for implementing the strategic initiative INDUSTRIE 4.0, Forschungsunion, Berlin, 2013.

[6] Position paper on The Impact of Robots on Productivity, Employment and Jobs, International Federation of Robotics (IFR), web page. Available (accessed 28.02.2018): https://ifr.org/ifr-press-releases/news/position-paper.

[7] Reconfigurable Machine Tools, in: CIRP Annals, 2001, pp. 269-274.

[8] Industry 4.0 Impacts on Lean Production Systems, in: Procedia CIRP, 2017, pp. 125-131.

[9] Yang Liu Yu Peng Bailing Wang Sirui Yao Zihe Liu, Review on Cyber-physical Systems, 自动化学报：英文版, Vol. 4, Iss. 1, 2017, pp. 27-40. http://lib.cqvip.com/qk/61504X/201701/90687266504849554849484852.html.

[10] E.A. Lee, Computing Foundations and Practice for Cyber-Physical Systems: A Preliminary Report, 2007, Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2007/EECS-2007-72.html.

[11] A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems, in: Manufacturing Letters, 2015, pp. 18-23.

[12] C. Perera, C. H. Liu, S. Jayawardena, M. Chen, A Survey on Internet of Things From Industrial Market Perspective, IEEE Access, Vol. 2, 2014, pp. 1660-1679. Available (accessed ID: 1).

[13] A. Whitmore, A. Agarwal, L. Da Xu, The Internet of Things—A survey of topics and trends, Information Systems Frontiers, Vol. 17, Iss. 2, 2015, pp. 261-274. Available (accessed ID: Whitmore2015): https://doi.org/10.1007/s10796-014-9489-2.

[14] O. Vermesan, P. Friess, Internet of Things : Converging Technologies for Smart Environments, River Publishers, Aalborg, 2012.

[15] Hao Chen, Xueqin Jia, Heng Li, A brief introduction to IoT gateway, IET International Conference on Communication Technology and Application (ICCTA 2011), pp. 610-613.

[16] C. S. Bontu, S. Periyalwar, M. Pecen, Wireless Wide-Area Networks for Internet of Things: An Air Interface Protocol for IoT and a Simultaneous Access Channel for Uplink IoT Communication, IEEE Vehicular Technology Magazine, Vol. 9, Iss. 1, 2014, pp. 54-63. Available (accessed ID: 1).

[17] Juha Petäjäjärvi, K. Mikhaylov, M. Pettissalo, J. Janhunen, J. Iinatti, Performance of a low-power wide-area network based on LoRa technology: Doppler robustness, scalability, and coverage, International Journal of Distributed Sensor Networks, Vol. 13, Iss. 3, 2017, pp. 1550147717699412. Available (accessed doi: 10.1177/1550147717699412; 25): https://doi.org/10.1177/1550147717699412.

[18] M. Simsek, A. Aijaz, M. Dohler, J. Sachs, G. Fettweis, 5G-Enabled Tactile Internet, IEEE Journal on Selected Areas in Communications, Vol. 34, Iss. 3, 2016, pp. 460-473. Available (accessed ID: 1).

[19] A. Luder, P. Gohner, B. Vogel-Heuser, Agent based control of production systems, IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society, pp. 7416-7421.

[20] F.L. Bellifemine, G. Caire, D. Greenwood, Developing Multi-Agent Systems with JADE, 1. Aufl. ed. Wiley, GB, 2007.

[21] C.M. Jonker, J. Treur, Agent-based simulation of reactive, pro-active and social animal behaviour, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 584-595.

[22] N.R. Jennings, M. Wooldridge, Applications of Intelligent Agents, in: N.R. Jennings, M.J. Wooldridge (ed.), Agent Technology: Foundations, Applications, and Markets, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, pp. 3-28.

[23] S. Poslad, Specifying protocols for multi-agent systems interaction, ACM Transactions on Autonomous and Adaptive Systems (TAAS), Vol. 2, Iss. 4, 2007, pp. 15.

[24] Standard FIPA specifications, Foundation for Intelligent Physical Agents, web page. Available (accessed 01.03.2018): http://www.fipa.org/repository/standardspecs.html.

[25] F. Bellifemine, A. Poggi, G. Rimassa, JADE–A FIPA-compliant agent framework, London, pp. 33.

[26] P.D. O'Brien, R.C. Nicol, FIPA — Towards a Standard for Software Agents, BT Technology Journal, Vol. 16, Iss. 3, 1998, pp. 51-59. Available (accessed ID: O'Brien1998): https://doi.org/10.1023/A:1009621729979.

[27] S. Bussmann, N.R. Jennings, M. Wooldridge, Multiagent systems for manufacturing control, Springer, Berlin ; Heidelberg ; New York ; Hong Kong ; London ; Milan ; Paris ; Tokyo, 2004.

[28] M. Wooldridge, N.R. Jennings, D. Kinny, The Gaia Methodology for Agent-Oriented Analysis and Design, Autonomous Agents and Multi-Agent Systems, Vol. 3, Iss. 3, 2000, pp. 285-312. Available (accessed ID: Wooldridge2000): https://doi.org/10.1023/A:1010071910869.

[29] Effective use of cloud computing in educational institutions, in: Procedia - Social and Behavioral Sciences, 2010, pp. 938-942.

[30] The NIST definition of cloud computing, Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, 2011.

[31] B.P. Rimal, E. Choi, I. Lumb, A Taxonomy and Survey of Cloud Computing Systems, 2009 Fifth International Joint Conference on INC, IMS and IDC, pp. 44-51.

[32] From cloud computing to cloud manufacturing, in: Robotics and Computer-Integrated Manufacturing, 2012, pp. 75-86.

[33] W. Tsai, X. Bai, Y. Huang, Software-as-a-service (SaaS): perspectives and challenges, Science China Information Sciences, Vol. 57, Iss. 5, 2014, pp. 1-15. Available (accessed ID: Tsai2014): https://doi.org/10.1007/s11432-013-5050-z.

[34] M. Washam, Automating Microsoft Azure Infrastructure Services, First edition ed. O'Reilly, Beijing, 2014.

[35] M. Boniface, B. Nasser, J. Papay, S. C. Phillips, A. Servin, X. Yang, Z. Zlatev, S. V. Gogouvitis, G. Katsaros, K. Konstanteli, G. Kousiouris, A. Menychtas, D. Kyriazis, Platform-as-a-Service Architecture for Real-Time Quality of Service Management in Clouds, 2010 Fifth International Conference on Internet and Web Applications and Services, pp. 155-160.

[36] W. Dawoud, I. Takouna, C. Meinel, Infrastructure as a service security: Challenges and solutions, 2010 The 7th International Conference on Informatics and Systems (INFOS), pp. 1-8.

[37] Y. Jadeja, K. Modi, Cloud computing - concepts, architecture and challenges, 2012 International Conference on Computing, Electronics and Electrical Technologies (ICCEET), pp. 877-880.

[38] James J. Kuffner, Cloud-Enabled Robots, IEEE-RAS International Conference on Humanoid Robots, 2010, Nashville, TN.

[39] K. Kamei, S. Nishio, N. Hagita, M. Sato, Cloud networked robotics, IEEE Network, Vol. 26, Iss. 3, 2012, pp. 28-34. Available (accessed ID: 1).

[40] Network robot systems, in: Robotics and Autonomous Systems, 2008, pp. 793-797.

[41] I.M. Delamer, J.L.M. Lastra, Loosely-coupled Automation Systems using Device-level SOA, 2007 5th IEEE International Conference on Industrial Informatics, IEEE, pp. 743-748.

[42] G. Hu, W. P. Tay, Y. Wen, Cloud robotics: architecture, challenges and applications, IEEE Network, Vol. 26, Iss. 3, 2012, pp. 21-28. Available (accessed ID: 1).

[43] G. A. McGilvary, A. Barker, M. Atkinson, Ad Hoc Cloud Computing, 2015 IEEE 8th International Conference on Cloud Computing, pp. 1063-1068.

[44] R. Limosani, A. Manzi, L. Fiorini, F. Cavallo, P. Dario, Enabling Global Robot Navigation Based on a Cloud Robotics Approach, International Journal of Social Robotics, Vol. 8, Iss. 3, 2016, pp. 371-380. Available (accessed ID: Limosani2016): https://doi.org/10.1007/s12369-016-0349-8.

[45] B. Kehoe, A. Matsukawa, S. Candido, J. Kuffner, K. Goldberg, Cloud-based robot grasping with the google object recognition engine, 2013 IEEE International Conference on Robotics and Automation, IEEE, pp. 4263-4270.

[46] J. Wan, S. Tang, Q. Hua, D. Li, C. Liu, J. Lloret, Context-Aware Cloud Robotics for Material Handling in Cognitive Industrial Internet of Things, IEEE Internet of Things Journal, Vol. PP, Iss. 99, 2017, pp. 1. Available (accessed Y1:).

[47] S. Thrun, J.J. Leonard, Simultaneous Localization and Mapping, in: B. Siciliano, O. Khatib (ed.), Springer Handbook of Robotics, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 871-889.

[48] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Galvez-Lopez, K. Haussermann, R. Janssen, J.M.M. Montiel, A. Perzylo, B. Schiessle, M. Tenorth, O. Zweigle, R. van de Molengraft, RoboEarth, IEEE Robotics & Automation Magazine, Vol. 18, Iss. 2, 2011, pp. 69-82. http://ieeexplore.ieee.org/document/5876227.

[49] D. Hunziker, M. Gajamohan, M. Waibel, R. D'Andrea, Rapyuta: The RoboEarth Cloud Engine, 2013 IEEE International Conference on Robotics and Automation, pp. 438-444.

[50] L. Riazuelo, M. Tenorth, D. Di Marco, M. Salas, D. Galvez-Lopez, L. Mosenlechner, L. Kunze, M. Beetz, J. D. Tardos, L. Montano, J. M. M. Montiel, RoboEarth Semantic Mapping: A Cloud Enabled Knowledge-Based Approach, IEEE Transactions on Automation Science and Engineering, Vol. 12, Iss. 2, 2015, pp. 432-443. Available (accessed ID: 1).

[51] M. Tenorth, A. Clifford Perzylo, R. Lafrenz, M. Beetz, The RoboEarth language: Representing and exchanging knowledge about actions, objects, and environments, 2012 IEEE International Conference on Robotics and Automation, pp. 1284-1289.

[52] J. Wan, S. Tang, H. Yan, D. Li, S. Wang, A.V. Vasilakos, Cloud robotics: Current status and open issues, IEEE Access, Vol. 4, 2016, pp. 2797-2807. http://ieeexplore.ieee.org/document/7482658.

[53] Capability Matchmaking Procedure to Support Rapid Configuration and Re-configuration of Production Systems, in: Procedia Manufacturing, 2017, pp. 1053-1060.

[54] N.J. Nilsson, Logic and artificial intelligence, Artificial Intelligence, Vol. 47, Iss. 1-3, 1991, pp. 31-56.

[55] T.R. Gruber, A translation approach to portable ontology specifications, Knowledge acquisition, Vol. 5, Iss. 2, 1993, pp. 199-220.

[56] S. Fan, W. Liu, X. Zhang, X. Luo, The Knowledge Description of Teaching Resource and Its Application, 2010 International Conference on Intelligent Computing and Cognitive Informatics, pp. 156-159.

[57] T. Gruber, Ontology, in: L. LIU, M.T. ÖZSU (ed.), Encyclopedia of Database Systems, Springer US, Boston, MA, 2009, pp. 1963-1965.

[58] N.F. Noy, D.L. McGuinness, Ontology development 101: A guide to creating your first ontology, 2001.

[59] J. Pérez, M. Arenas, C. Gutierrez, Semantics and Complexity of SPARQL, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 30-43.

[60] SPARQL Query Language for RDF, W3C, web page. Available (accessed 27.02.2018): https://www.w3.org/TR/rdf-sparql-query/.

[61] SPARQL 1.1 Query Language, W3C, web page. Available (accessed 28.02.2018): https://www.w3.org/TR/sparql11-query/.

[62] B. Ramis Ferrer, An ontological approach for modelling configuration of factory-wide data integration systems based on IEC-61499, Tampere University of Technology, 2013, 76 p. Available: http://dspace.cc.tut.fi/dpub/handle/123456789/21541.

[63] Protégé, Stanford University, web page. Available (accessed 27.02.2018): https://protege.stanford.edu.

[64] IRB 140, ABB, web page. Available (accessed 06.04.2018): http://new.abb.com/products/robotics/industrial-robots/irb-140.

[65] Raspberry Pi 3 Model B+, raspberrypi org, web page. Available (accessed 06.04.2018): https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/.

[66] Camera Module V2, raspberrypi org, web page. Available (accessed 06.04.2018): https://www.raspberrypi.org/products/camera-module-v2/.

[67] Apache Jena Fuseki, Apache Jena org, web page. Available (accessed 06.04.2018): https://jena.apache.org/documentation/fuseki2/.

[68] Ontology versus Database, in: IFAC-PapersOnLine, 2015, pp. 220-225.

[69] Google Cloud Vision API Documentation, Google LLC, web page. Available (accessed 06.04.2018): https://cloud.google.com/vision/docs/.

[70] A. Hussnain, B. Ramis Ferrer, J. L. Martinez Lastra, Towards the Deployment of Cloud Robotics at Factory Shop Floors: a Prototype for Smart Material Handling,IEEE 1st International Conference on Industrial Cyber-Physical Systems,accepted for publication, 2018.