Connectivity, Security and Integration for Cloud Manufacturing

CHEN WANG

Master Thesis
School of Engineering and Management
Department of Production Engineering
The Royal Institute of Technology
Stockholm, Sweden 2017
# Content

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>10</td>
</tr>
<tr>
<td>1.1</td>
<td>Background</td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>The scope of the research</td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td>Organization of the dissertation</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Literature review</td>
<td>13</td>
</tr>
<tr>
<td>2.1</td>
<td>Cloud manufacturing</td>
<td></td>
</tr>
<tr>
<td>2.1.1</td>
<td>Cloud Manufacturing System</td>
<td></td>
</tr>
<tr>
<td>2.1.2</td>
<td>Supporting technologies for CMfg</td>
<td>15</td>
</tr>
<tr>
<td>2.2</td>
<td>Cloud Robotics</td>
<td>16</td>
</tr>
<tr>
<td>2.2.1</td>
<td>Cloud Robotics System</td>
<td>16</td>
</tr>
<tr>
<td>2.2.2</td>
<td>Supporting technologies for Cloud Robotics</td>
<td>17</td>
</tr>
<tr>
<td>2.3</td>
<td>Security</td>
<td>18</td>
</tr>
<tr>
<td>2.3.1</td>
<td>Security threats in CMS</td>
<td>18</td>
</tr>
<tr>
<td>2.3.2</td>
<td>Security solutions</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>Proposed system architecture</td>
<td>23</td>
</tr>
<tr>
<td>3.1</td>
<td>Proposed Cloud Manufacturing System</td>
<td></td>
</tr>
<tr>
<td>3.1.1</td>
<td>Proposed CMS</td>
<td>23</td>
</tr>
<tr>
<td>3.1.2</td>
<td>CMS deployment scenarios</td>
<td>24</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Work flow of Cloud Manufacturing</td>
<td>27</td>
</tr>
<tr>
<td>3.2</td>
<td>Developed CMS Specific for Robotics</td>
<td>29</td>
</tr>
<tr>
<td>3.3</td>
<td>Security Issues for CMS</td>
<td>30</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Security threat sources of CMS</td>
<td>30</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Possible solutions</td>
<td>31</td>
</tr>
<tr>
<td>4</td>
<td>Implementation and system development</td>
<td>33</td>
</tr>
<tr>
<td>4.1</td>
<td>System connectivity</td>
<td>34</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Methodology</td>
<td>34</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Implementation</td>
<td>34</td>
</tr>
<tr>
<td>4.1.3</td>
<td>Functionality examining</td>
<td>36</td>
</tr>
<tr>
<td>4.2</td>
<td>Data acquisition</td>
<td>36</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Methodology</td>
<td>36</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Implementation</td>
<td>37</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Functionality examining</td>
<td>38</td>
</tr>
<tr>
<td>4.3</td>
<td>Data visualization</td>
<td>41</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Methodology</td>
<td>41</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Implementation</td>
<td>41</td>
</tr>
<tr>
<td>4.3.3</td>
<td>Functionality examining</td>
<td>43</td>
</tr>
<tr>
<td>4.4</td>
<td>Remote control/program</td>
<td>46</td>
</tr>
</tbody>
</table>
Sammanfattning


Nyckelord

moln tillverkning, molnrobotik, gateway, anslutning, säkerhet, universell robot
Abstract

This master thesis project aims to connect the industrial robot to the Cloud platform, and evaluate the connectivity and security. To realize better connectivity, security and integration, a modified Cloud Manufacturing System (CRS) architecture is proposed, which is characterized by high modularity, standardization and composability. The architecture’s specific applications in private, public and hybrid cloud are discussed as well. Then, one system architecture with detailed software composition is designed for Cloud Robotics. According to the proposed system architecture, possible security threat sources and corresponding solutions are presented.

During the project, Universal Robot 5 (UR5) is utilized as a practical robot instance to develop a communication routine between KTH Cloud and robots. An Application Program Interface (API) written by Python for Universal Robots and the server is established. The API consists of two modularized part, Gateway Agent and Application Package. The Gateway Agent realizes the connection between the Universal Robot 5 (UR5) and the cloud, while the Application Package can be customized according to specific application and requirements. In this project, three main functions are developed in the Application Package, including data acquisition, data visualization and remote control. Besides, to evaluate connectivity and stability, private robotics cloud system and public robotics cloud system are simulated with KTH Cloud. The hybrid robotics cloud system is discussed as well. Through the results of case studies, the connectivity and integration of Cloud Manufacturing System are verified.

Keywords

cloud manufacturing, cloud robotics, gateway, connectivity, security, Universal Robot
Acknowledgements

I wish to express my sincere gratitude to my supervisor Xi Wang for all the support and encouragement he gave to me during these days, which also help me to try and learn so many interesting things.

Secondly, I sincerely thank my course coordinator Ove Bayard who helped me a lot during the two years of study and this master project.

I also would like to thank Hongyi Liu for his kind help of Universal Robot and my dear classmates who give me sincere suggestions and encouragement.

Besides, I want to thank my friends who has accompanied me to have a unforgettable time in Sweden.

Last but not least, to my parents who keep giving me whole love and support, either morally, financially and physically.
# Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>API</td>
<td>Application program interface</td>
</tr>
<tr>
<td>CIO</td>
<td>Chief Information Officer</td>
</tr>
<tr>
<td>CMfg</td>
<td>Cloud Manufacturing</td>
</tr>
<tr>
<td>CMS</td>
<td>Cloud Manufacturing System</td>
</tr>
<tr>
<td>CNC</td>
<td>Computer Numeric Control</td>
</tr>
<tr>
<td>CR</td>
<td>Cloud Robotics</td>
</tr>
<tr>
<td>CRS</td>
<td>Cloud Robotics System</td>
</tr>
<tr>
<td>DoS</td>
<td>Denial of Service</td>
</tr>
<tr>
<td>ES</td>
<td>Enterprise System</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HCI</td>
<td>Higher-level Cloud Interface</td>
</tr>
<tr>
<td>ICMS</td>
<td>Interoperable Cloud Manufacturing System</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>MaaS</td>
<td>Machine-as-a-Service</td>
</tr>
<tr>
<td>MCloud</td>
<td>Manufacturing Cloud</td>
</tr>
<tr>
<td>OACR</td>
<td>Optimal Allocation of Computing Resources</td>
</tr>
<tr>
<td>RaaS</td>
<td>Robot-as-a-Service</td>
</tr>
<tr>
<td>RBF</td>
<td>Radial Basis Function</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio Frequency Identification</td>
</tr>
<tr>
<td>ROS</td>
<td>Robot Operating System</td>
</tr>
<tr>
<td>RTDE</td>
<td>Real-Time Data Exchange</td>
</tr>
<tr>
<td>RTRC</td>
<td>Real-Time Remote Control</td>
</tr>
<tr>
<td>SCM</td>
<td>Smart Cloud Manager</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>SMEs</td>
<td>Small and Medium-sized Enterprises</td>
</tr>
<tr>
<td>SPI</td>
<td>Service Provider Interface</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>UCloud</td>
<td>User Cloud</td>
</tr>
<tr>
<td>UM</td>
<td>Ubiquitous Manufacturing</td>
</tr>
<tr>
<td>VPN</td>
<td>Virtual Private Network</td>
</tr>
<tr>
<td>WAN</td>
<td>Wide Area Network</td>
</tr>
</tbody>
</table>
List of figures

Fig. 1. Relationship of CM and CR ................................................................. 11
Fig. 2. Thesis Scope ................................................................................... 11
Fig. 3. Organization of the dissertation ......................................................... 12
Fig. 4. ICMS system architecture [4] ............................................................ 14
Fig. 5. Detailed ICMS system architecture [1] .............................................. 14
Fig. 6. Architecture of CRS [20] ................................................................ 17
Fig. 7. Security requirements undermined by security threats [26] .............. 19
Fig. 8. Security solutions for AMI system [26] ........................................... 21
Fig. 9. Gateway security isolation system based on Trusted Computing [29] 22
Fig. 10. Trajectory of a user request [34] ..................................................... 22
Fig. 11. Proposed CMS system architecture .............................................. 24
Fig. 12. CMS scenario specific for private CMfg service platform .............. 25
Fig. 13. CMS scenario specific for public CMfg service platform ............... 26
Fig. 14. CMS scenario specific for hybrid CMfg service platform ............... 27
Fig. 15. Workflow of CMS adapted from [1] ................................................ 28
Fig. 16. Developed CMS Specific for Robotics .......................................... 29
Fig. 17. CMS structure from security perspective ....................................... 30
Fig. 18. Program flow chart ...................................................................... 33
Fig. 19. Program flow chart of connection part ......................................... 35
Fig. 20. Connection part UI ...................................................................... 36
Fig. 21. Screen shot of log ....................................................................... 36
Fig. 22. Program flow chart of data acquisition part ................................... 38
Fig. 23. Data recording part .................................................................... 38
Fig. 24. Screen shot of data recording operation ....................................... 39
Fig. 25. Screen shot of log after data recording operation .......................... 39
Fig. 26. Screen shot of warning ................................................................. 39
Fig. 27. Screen shot of XML file .............................................................. 39
Fig. 28. Screen shot of UR5 robot log ....................................................... 40
Fig. 29. Screen shot of recording data file ................................................ 40
Fig. 30. Program flow chart of history data plotting part ............................. 42
Fig. 31. Program flow chart of real-time data plotting part ......................... 43
Fig. 32. History data plotting part ............................................................ 43
Fig. 33. Screen shot of history data plot operation ..................................... 44
Fig. 34. Screen shot of line chart of actual elbow velocities ....................... 44
Fig. 35. Real-time data plotting part UI ..................................................... 44
Fig. 36. Screen shot of real-time data plotting operation ........................................ 45
Fig. 37. Screen shot of line chart of actual robot voltage ........................................ 45
Fig. 38. Program flow chart of digital output set part .................................................. 46
Fig. 39. Program flow chart of joints movement control part ....................................... 47
Fig. 40. Digital output set part ....................................................................................... 48
Fig. 41. Digital output set part with specific setting ...................................................... 48
Fig. 42. Log for digital output set ................................................................................... 48
Fig. 43. Screen shot of robot’s I/O monitoring UI .......................................................... 49
Fig. 44. Joints movement control part ............................................................................ 49
Fig. 45. Joints movement control part with specific setting ........................................... 49
Fig. 46. Script programming part ................................................................................... 50
Fig. 47. A script programming example ......................................................................... 50
Fig. 48. Screen shot of log for script programming ....................................................... 50
Fig. 49. Screen shot of robots I/O monitoring UI ........................................................... 51
Fig. 50. System Architecture of Private Cloud Case ...................................................... 54
Fig. 51. Reliability of Real-time Data Plotting with Different Plotting Frequencies ........ 56
Fig. 52. System Architecture of Public Cloud Case ....................................................... 57
Fig. 53. System Architecture of Hybrid Cloud Case ..................................................... 59
List of tables

Table 1. RTDE using instruction [39] .................................................................................. 35
Table 2. Data recording results summary ............................................................................. 40
Table 3. Functionality Examining Result of Private Cloud Case ............................................. 54
Table 4. Test Results of Response Delay of Private Cloud Case ............................................ 55
Table 5. Test Results of Reliability of Private Cloud Case ..................................................... 55
Table 6. Functionality Examining Result of Public Cloud Case .............................................. 57
Table 7. Test Results of Response Delay of Public Cloud Case ............................................. 58
Table 8. Test Results of Reliability of Private Cloud Case ..................................................... 58
Chapter 1: Introduction

This chapter presents the motivations of the master thesis. The first section is a brief description of the research background. In the second section, the scope of the research work is defined. Finally, the outline of this dissertation is reported.

1.1 Background

In recent years, Cloud has become a popular technology which gains huge market success globally. Cloud concept indicates a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storages, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. It provides elastic and flexible supports for service-oriented production models. Instead of investing on costly IT equipment or software licenses as a whole, the Cloud users are able to pay for the exact amount of software or hardware usage based on pay-as-you-go principle. It is particularly helpful for Small and Medium-sized Enterprises (SMEs) that are normally short of start-up capitals for new investments or equipment.

It is thus logical and reasonable for manufacturing researchers and stakeholders to adopt Cloud into the manufacturing industry so as to improve current production performance. Cloud-based automation is predicted to alter the automation landscape by means of improved perception, faster planning, accurate modelling, lifelong learning, large-scale systems, new ways of interacting with human, and so forth. The availability of new technologies, e.g. computing Cloud, big data management, machine learning, open platform, and broad bandwidth, and their possible uses in Industrial Robots and Computer Numeric Control (CNC) machines have opened the door to a whole new line of research called Cloud Manufacturing (CMfg).

During the past few years, many comprehensive Cloud Manufacturing system architecture have been proposed. The next step, transforming the theory into a successful business case, however, has not yet been demonstrated. One of the challenges is to connect Industrial Robots, CNC machines and other manufacturing system components to the Cloud platform, and evaluate the connectivity and security. This thesis work will address this challenge.

1.2 The scope of the research

The topic of this master thesis is connectivity, security and integration for Cloud Manufacturing. As shown in Fig. 1, Cloud Manufacturing and Cloud Robotics (CR) are two different concepts, but they overlap partially. Cloud Robotics includes more robot application, not only in industrial environment but also in medical, traffic, service fields, etc. On the other hand, robots are one of the important physical resources in manufacturing industry, especially for Cloud Manufacturing. In this thesis project, Robotics application in Cloud Manufacturing is the discussion scope, as the overlapped part in Fig. 1.
The main research objective is to develop a communication routine between Industrial Robots and KTH Cloud for case study in a cloud environment. Universal Robot UR5 will be taken as the research case. The general research scope is shown in Fig. 2.

1.3 Organization of the dissertation

As shown in Fig. 3, this dissertation consists of six chapters.

Chapter 1 gives an introduction to the research background, challenges and scope of the research. Chapter 2 review literatures of the related research to specify the start point of this master thesis. Proposed system architecture of Cloud Manufacturing (CMfg) and Cloud Robotics (CR) have been discussed in Chapter 3. Chapter 4 reported the detailed implementation and software development to realize the connectivity, data acquisition, data visualization and remote control of Cloud Manufacturing. Verification methodology, results of the software and the case studies of software in different cloud environments are described
in Chapter 5. Finally, conclusions, discussions about research limitations and future work are described in Chapter 6.

Fig. 3. Organization of the dissertation
Chapter 2: Literature review

This chapter consists of a literature review of the research related to this master thesis project. It is consisted of three sections. The first section introduces the conducted work in the CMfg field, including present research situation, supported technology. The second section reports the correlative research about Cloud Robotics. The last section goes through the related work in the cloud-related and gateway-related security field.

2.1 Cloud manufacturing

In the increasingly globalized manufacturing context, customer oriented manufacturing is a promising approach to improving the service quality and competitiveness, in particular for the small and medium-sized enterprises (SMEs) [1]. The rest of this section reports the background, concept, system architecture and application of Cloud Manufacturing System (CMS), and related technologies for CMS.

2.1.1 Cloud Manufacturing System

A. Background

The manufacturing industry is undergoing a major transformation enabled by IT and related smart technologies to provide on-demand computing services with high reliability, scalability and availability in a distributed environment [2]. The transformation from production-oriented manufacturing to service-oriented manufacturing stumbled on a series of problems, such as resources utilization, machines scheduling, information communication, data storage and etc. But the development of Cloud Computing addresses these problems effectively, enabling industries and enterprises do their businesses in that dynamically scalable and virtualized resources are provided as a service over the Internet [2]. Therefore, the Cloud users are able to pay for the exact amount of software or hardware usage, which is particularly helpful for Small and Medium-sized Enterprises (SMEs) that are normally short of start-up capitals for new investments or equipment.

B. Concept

Cloud Manufacturing is defined as a model for enabling ubiquitous, convenient and on demand network access to a shared pool of configurable manufacturing resources (e.g., manufacturing software tools, manufacturing equipment, and manufacturing capabilities) that can be rapidly provisioned and released with minimal management effort or service provider interaction [2].

In CMS, various manufacturing resources can be intelligently connected into internet, and automatically managed and controlled using IoT (Internet of Thing) technologies (e.g., Radio Frequency Identification (RFID), wireless sensor network), then the manufacturing resources and abilities are encapsulated into cloud services that can be accessed, invoked and deployed based on knowledge by using virtualization technologies and service-oriented technologies, and cloud computing technologies [3]. Manufacturing infrastructure, platform and software application in CMfg can be offered as a service to a cloud user [1].
C. System architecture

Besides, the preliminary concept of the Interoperable Cloud Manufacturing System (ICMS) for production management has been proposed in [4]. Integration mechanisms of physical resources are proposed, and customized production planning methods are developed and validated by the case study. As shown in Fig. 4, the system is divided into three layers. The Cloud Layer is service coordinator and supervisor of whole production system [4]. Robots, machines, operators and other devices are included in Physical Resource Layer. Between the Cloud Layer and Physical Resource Layer, it is Local Server that stream information and connect other two layers.

![Fig. 4. ICMS system architecture [4]](image)

---

Lu et al. [5] proposed a hybrid Cloud structure that allowed companies with different business scenarios to deploy different cloud modes. Three typical cloud modes, i.e. private cloud,
community cloud and public cloud are supported in the system. Besides, to enable real-time machine availability and execution status monitoring, a tiered system architecture and IEC 61499 function blocks for prototype implementation are introduced in [6].

A hierarchical ten-layers architecture of cloud manufacturing has been introduced in [7]. Wang and Xu [1] proposed a service-oriented, interoperable Cloud Manufacturing system. As shown in Fig. 5, service methodologies are developed to support two types of Cloud users, i.e., customer user and enterprise user. As public Cloud infrastructure, ICMS consists of three layers, i.e., Smart Cloud Manager (SCM), User Cloud (UCloud), and Manufacturing Cloud (MCloud).

D. Application

Compared with traditional manufacturing, CMfg offers several key advantages, optimizing resource utilization, reducing investment and lower cost, generating new types of business model, reducing the products research and provision time and etc. For example, Wang et al. [8] extended the manufacturing cloud to the remanufacturing sector for electronic wastes managements.

These key advantages enable CMfg to be applied in various manufacturing environment. Firstly, CMfg can be adopted in large enterprises to integrate existing computing resources, software resources and data resources. Build service platform providing advanced products and support subordinate apartments with technologies, software, data service. This will greatly promote the product innovation designed ability. Besides, regional processing resource sharing service platform and manufacturing service supporting platform are two promising application directions of CMfg.

For the consideration of security and safety, and utilization scope, there are primarily four kinds of CMfg service platform, namely:

- a public CMfg service platform;
- a private CMfg service platform;
- a community CMfg service platform;
- a hybrid CMfg service platform [7].

2.1.2 Supporting technologies for CMfg

Cloud Manufacturing is described as a computing and service-oriented manufacturing model developed from existing advanced manufacturing models (e.g., Application service provider, Agile Manufacturing, Networked Manufacturing, and Manufacturing Grid), enterprise information technologies under the support of cloud computing, Internet of things, virtualization and service-oriented technologies, and advanced computing technologies [1].

The impact of emerging Internet of Things (IoT) on Enterprise Systems (ESs) in modern manufacturing has been investigated in [9], and the evolution of manufacturing system paradigms was discussed to identify the requirements of decision support systems in dynamic and distributed environments.

Laili et al. [10] proposed a new comprehensive model for Optimal Allocation of Computing Resources (OACR) is proposed in the Cloud Manufacturing system. [11] has examined knowledge-based service composition and adaptive resource planning in a cloud manufacturing environment, in order to develop an integrated networked environment,
allowing fast resource allocation for a given service request, subject to governance policies, resource access policies, resource availability information and etc.

Zhang et al. [12] proposed a work-in-progress management framework based on smart objects such as radio frequency identification/Auto-ID devices and web service technologies in a Ubiquitous Manufacturing (UM) environment, which created opportunities for developing next-generation manufacturing systems with real-time traceability, visibility and interoperability.

2.2 Cloud Robotics

Cloud Robotics Systems (CRSs) take advantage of all the benefits of the Cloud, while at the same time providing users who want to focus on the functionalities of robots in an economic way to investigate their robotic system [13]. The rest of this section reports the background, concept, system architecture and applications of Cloud Robotics (CR), and related technologies for CRSs.

2.2.1 Cloud Robotics System

A. Background

Robots make significant socioeconomic impacts to human lives [14], such as production, security, service and etc. However, robots are limited in terms of computational capacity, memory and storage, having physical constraints such as size, shape, power supply, motion mode and working environment as well [15]. To break these limitations, robotic technologies can be integrated with advanced networking technologies to foster the emergence of networked robotics [16]. However, the advancement of networked robotics is restricted by resource, information, and communication constraints inherent in the existing framework [17]. Emerging advances and the increasing availability of networking in the “Cloud” suggests new approaches where processing is performed remotely with access to dynamic global datasets to support a range of functions [18].

Networked Robotics serves as a stepping stone towards Cloud Robotics. For instance, cloud-enabled network robotics leverages emerging cloud computing technologies to enhance networked robotics removing analytical duties off the robot [16]. Cloud Robotics (CR) was born from the merger of service robotics and cloud technologies, enabling robots to benefit from the powerful computational, storage, and communications resources of modern data centers by offloading computation-intensive and storage-intensive jobs into the cloud [15-16].

B. Concept

The term “cloud-enabled robotics” was presented by James Kuffner for the first time at the IEEE RAS Int. Conference on Humanoid Robotics in 2010, who is the first to point out the potential of distributed networks combined with robotics, primarily to enhance the robot [19]. Cloud robotics is an emerging field of robotics embedded in cloud computing, cloud storage and cloud networking [16].

C. System architecture

Wan et al. [20] described the overall architecture of the Cloud Robotics System (CRS). As shown in Fig. 6, the architecture of CR is mainly composed of two parts: the cloud platform and its related equipment and the bottom facility [20].
Miratabzadeh et al. [16] proposed a software architecture for Cloud Robotics which intends three subsystems in the cloud environment: Middleware Subsystem, Background Tasks Subsystem, and Control Subsystem. A Cloud Robotics architecture focus on localization problem has been proposed in [21], and consists of three sections: Remote Robot, Interface and Cloud.

Hu et al. [17] have proposed a cloud robotics architecture, which is organized into two complementary tiers: a machine-to-machine (M2M) level and a machine-to-cloud (M2C) level, to address the constraints faced by current networked robots.

D. Application

Compared with traditional robots, Cloud Robotics (CR) provides two main advantages. Firstly, CR is able to deal with higher level of computation-intensive tasks as it offloads part of computing to the cloud. Secondly, connecting with cloud means easy access to vast amount of data, which will provide more information and knowledge support for robots. Due to these advantages, CR has a wide range of potential application in the areas of intelligent transportation, environment monitoring, health care, smart home, entertainment, education, defense and etc. [17]. Wan [20] describes some classic applications of the cloud robotics, including SLAM, grasping and navigation. Besides, the concept of “robot-as-a-service” (RaaS) refers to robots that can be dynamically combined to give support to the execution of specific applications [15].

2.2.2 Supporting technologies for Cloud Robotics

Cloud Robotics System involves a number of cloud, networking and embedded system’s technologies, as well as all kinds of wireless communication protocols [20].
Cloud Computing is one of the fundamental driving forces. It is currently driving interest in both academia and industry, combines robot technology with network and Cloud-computing infrastructure that connects amount of robots, sensors, portable devices and most important a data-center [22]. In the field of multi-robot operations, cloud computing has greatly accelerated the speed of development of robotic and automation equipment [20]. Besides, Cloud Robotics has rapidly gained momentum with initiatives by companies such as Google, Willow Garage and Gostai as well as more than a dozen active research projects around the world [22]. For example, [22] used Artificial Neural Network (ANN) for the training of locations with Radial Basis Function (RBF), to establish the communication between the cloud and robot over a large environment using the JAVA-RMI interface and identify the location from the images sent by the robot.

The Cloud Robotics architecture invokes other cloud technologies such as cloud storage and other networking platform arranged on the assistances of congregated infrastructure and shared services for robotics, for instance Robot Operating System (ROS) [3]. ROS and RoboEarth are the two most representative open source technologies in Cloud Robotics field. ROS is a well-known framework, whose open source license allows the improvement of the code reuse rate and development efficiency of this robot operating system [20].

Network connectivity develops with CRS as well. [15] proposed communication protocols and several elastic computing models to handle different applications for Cloud Robotics, extending the computation and information sharing capabilities of networked robotics.

Agostinho et al. [23] presented the design, implementation, and evaluation of a cloud computing environment for networked robotics, aiming to support distributed robotics applications that require large shares of computing resources and very low communication delays. Japanese researchers building the business platform Rapyuta [24] for Cloud Robotics to providing secured customizable computing environments in the cloud.

2.3 Security

The development of CMfg addresses an explosive growth of internet-connected machines and to handle huge amount of user and company data. It is characterized by massive scalability and service-oriented business. Despite the promising potential, privacy and security for CMS and the risks become a potential problem that cannot be neglected. Many chief information officers (CIOs) in large enterprises identify security concerns as the top reason for not embracing the public cloud more aggressively and not benefitting from associated cost optimizations [25]. In order to make CMfg into a practical realization, the security risks need to be better understood and effective solutions need to be proposed.

Besides, the security discussed in this chapter is the security problems caused by CMfg or Cloud Computing, except the security problems in traditional manufacturing environment. This first section of this chapter describes the security issues for CMfg. The second section briefly reports a number of approaches to addressing security issues in CMfg.

2.3.1 Security threats in CMS

Security often tops the list of cloud user concerns [25]. The adoption of Cloud Computing technology in CMS present a series brand new security issues than traditional manufacturing. Fig.7 [26] illustrates the relationships between threats and security requirements. The
following section discusses security threats in CMS from three aspects, data security, gateway security and network security.

![Security Chart]

**Fig. 7.** Security requirements undermined by security threats [26]

### A. Data security

CMfg appear to be a very promising and interesting develop direction that every customers and service provides is able to access or store data in cloud directly or indirectly. However, this mechanism also poses great threat to data security, which is linked with enterprises survival directly. Therefore, it is very important for Cloud administrators to secure the data, not only when it is stored in cloud database, but also in the transmission process.

A data is secure, if it fulfils three conditions: confidentiality, integrity and availability.

- Confidentiality means the data is understandable to the receiver only for all others it would be waste; it helps in preventing the unauthorized disclosure of sensitive information.
- Integrity means data received by receiver should be in the same form, the sender sends it; integrity helps in preventing modification from unauthorized user.
- Availability refers to assurance that user has access to information anytime and to any network. [27].

While Chapter 2 in [28] added other two conditions considering the emergence of new technologies and threats: authentication and accountability.

- Authentication: Assuring the identity of the entity involved in the communication.
- Accountability: Assuring that no entity can deny its participation in a data transfer between them.

Two types of data security threats were discussed in [28], data breaches and data loss. Chapter 3 in [28] also introduces other security threats in cloud computing very elegantly with detailed definitions of nine security threats: data breaches, data loss, account or service hijacking, insecure interfaces and APIs, threats to availability, malicious insiders, abuse of cloud services, insufficient due diligence, and shared-technology vulnerabilities.
B. Gateway security

Gateway plays the role of both connection and isolation, controlling the information stream between different system components. As there is a lot of protected information in the system and needing for information exchange internally and externally, the internal information system security is facing serious challenges [29]. Haupt et al. [30] described the gateway system a tree-tier architecture, front end, gateway API and middle tier, and propose a gateway security model corresponding to the architecture.

In cloud computing system, a cloud storage gateway can provide encryption, authentication, and authorization, but it is a server that resides at the customer premises and exposes cloud storage services as if they were local storage devices [31]. It is same in CMSs, so isolation mechanisms also need to be design carefully to ensure device security and cloud security.

C. Cloud security

Since enterprises are attracted to CMfg due to data savings in IT outlay and accessibility of IT resources, it is necessary to understand the business risks involved. In CMS, most of the important data, knowledge and computing resource are put in the Cloud. Once the Cloud is attacked or threatened by some security problems, all the involved enterprises or customers will suffer tremendous losses. Actually, vulnerabilities are almost inevitable in any CMS, especially in cloud computing system. The vulnerabilities included in cloud computing system, when exploited by attackers, can cause service disruptions, data loss, data theft, etc. [28].

Besides, the cloud’s autonomic and virtualized aspects can bring new threats, such as cross-virtual machine side-channel attacks, or vulnerabilities due to data proliferation, dynamic provisioning, the difficulty in identifying physical servers’ locations or a lack of standardization [25].

The manufacturing cloud is designed to be easily available all the time. Whether it is a private or a public cloud, availability is an undeniable feature that many organizations seek [28]. However, this characteristic can be a double-edged sword. For example, Denial of Service (DoS) attacks aim at making a certain network service unavailable to its legitimate users, by keeping the resources busy such that these resources become unavailable to the users this service was aimed to serve [28].

2.3.2 Security solutions

In the existing system already exists in a variety of technology solutions for internal information system security, such as firewall technology, Virtual Private Network (VPN) technology, network gateway technology, protocol conversion technology [29]. Fig.8 [26] illustrates some of the many security technologies and policies that can be used to provide the various security requirements in Advanced Metering Infrastructure (AMI) systems. As the AMI system have some aspects in common with CMS, these solutions are worth to learn. The following section chooses three most widely used solutions to discuss, cryptography, trust mechanism and end-to-end security architect.
Fig. 8. Security solutions for AMI system [26]

A. Cryptography
In the cloud confidentiality is obtained by cryptography. Cryptography is a technique of converting data into unreadable form during storage and transmission that it appears waste to intruder [27]. Pansotra and Singh [27] introduced three types of cryptography algorithms: Symmetric algorithms, Asymmetric algorithms and Hashing. Moreover, many cloud storage gateways facilitate the use of encryption techniques and frameworks (e.g. RSA, OpenPGP), whereby the gateway has no access to customer data, as all encryption and decryption happens at the user site [25]. In [32], a data retrieval scheme using attribute-based encryption was suggested.

B. Trust mechanism
Mao and Xu [29] have proposed a new network security isolation system and trusted computing information protection model, which is more secure than traditional security isolation scheme. The system architecture is shown in Fig. 9. All the information will pass through the approval process before interact with outside to ensure data security.

Opportunities are being created for some service providers to offer cloud services that have greater assurance and that employ mechanisms to reduce risk [25].

Wang et al. [33] utilized SSL VPN to maintain security and safety of Cloud system and data. A self-managed mechanism gives companies full control of their businesses and boosts their trust with enhanced privacy protection has been introduced in [5]. It enables companies to set self-defined access rules for each resource.
C. End-to-end security architect

When an architect designs a data center to host a public, private, or hybrid cloud, a primary consideration is end-to-end security [34]. For example, when a cloud user sends a product request by web applications, the data needs to go through a firewall first from a web browser to the Internet. Similarly, the request needs to go through another firewall to the cloud server, to reach distribution switch, a load balancer and application delivery controller. This flow is shown in Fig. 10 [34].
Chapter 3: Proposed system architecture

This chapter presents the proposed architectures of Cloud Manufacturing System (CMS). The first section introduces the detailed CMS architecture modified from ICMS, and the work flow of CM. Besides, three CMS cases specific for private, public and hybrid cloud are discussed. The second section describes the developed system architecture specific for Cloud Robotics, followed by related security issues for CMS in the third section.

3.1 Proposed Cloud Manufacturing System

Wang has proposed that the aim of Cloud Manufacturing is to transform manufacturing businesses to a new paradigm in that manufacturing capabilities and resources are componentized, integrated and optimized globally [1]. Firstly, both the hardware and software need to be componentized with an international standard, which is the necessary condition for integration. Secondly, as the amount of involved hardware and software varies depends on the scale of the manufacturing system, the flat system architecture is unable to run efficiently. It is fundamental to integrated the resources and capabilities properly to build an organized multi-layer system. This multi-layer system increases its information managing capabilities effectively, which also can provide better solutions.

However, the proper integration mechanism of nonphysical resources varies depends on the needs of different companies, especially considering the size of enterprises.

3.1.1 Proposed CMS

Nowadays, CMfg has already started to be applied in practical production environment and built lots of successful business cases. For example, many large enterprises have been working on developing their own manufacturing cloud, to realize better internal resource utilization and strengthen research ability. Besides, cloud service for other fields or for public usage have made great progress in recent years. To fit in the popularity of cloud service and gain further development, CMS should be designed with high modularity, standardization and composability.

Modularity means the degree to which CMS’s components may be separated and recombinated. This increases the flexibility of CMS and provide easier way for system upgrading, which also decrease the system maintenance cost by system components reusing.

Standardization is a meaningful characteristic for cooperation between enterprises and clouds. The standardized items are mainly interfaces between different modules. Similar with some of the computer components, customers can buy specific components with desired performance and price to replace the broken ones, but without extra effort to figure out how these components should be used. This kind of standardized interfaces add more values to components of CMS.

Modularity and standardization are necessary conditions to composability. A highly composable CMS provides recombinant modules that can be selected and assembled in various combinations to satisfy specific requirements. As cloud service is advancing rapidly, manufacturing will become a small part of this new pay-as-you-go business, CMS is promising to combine with other cloud service to develop new business model. These three
characteristics are the insurance for CMS to face tough challenges to realize long-term development.

Based on the requirement discussed above and the ICMS system architecture in [4], an adjusted system architecture is proposed, a Cloud Manufacturing System with better extendability. As shown in Fig.11, the whole system is divided into three layers: (i) Physical Layer, (ii) Gateway Layer, (iii) Cyber Layer.

The Physical Layer includes all the physical resources in traditional manufacturing systems, such as robots, machines, humans and other devices. At this layer manufacturing tasks assigned by Cloud are taken by control units of production devices, e.g. Robot-as-a-Service (RaaS) unit and Machine-as-a-Service (MaaS) unit [4]. Besides, there are communication network among physical resources for inner-layer interaction.

The top layer is the Cyber Layer, which play the role of management, cloud environment building and data storage, even advanced function such machine learning and system development. Except the API for administrators and users, this layer is also equipped with four standardized interface for data communication and event communication, with both lower layers and higher layers.

Between the Cyber Layer and Physical Layer, it is the Gateway Layer that connect and isolate them. Gateway agents are designed to communicate with physical resources. They abstract data from devices and delegate tasks from Cyber Layer to devices. The Gateway Host manages Gateway Agents network.

Fig. 11. Proposed CMS system architecture

3.1.2 CMS deployment scenarios

For the consideration of security and safety, and utilization scope, there are primarily four kinds of CMfg service platform, namely:

- a public CMfg service platform;
- a private CMfg service platform;
- a community CMfg service platform;
- an hybrid CMfg service platform [7].
This section introduces CMS cases with better extendability specific for those four kinds of CMfg service platform.

A. CMS scenario specific for private CMfg service platform

For private CMfg, the Cyber Layer does not need the interfaces with higher layers, as shown in Fig. 12. It is the simplest application of the proposed CMS system, mostly are used inside company or factory. The private cloud is building in local server or private server, based on the built connection with all robots and other equipment. The connection is built through gateway agents, which could be a software package developed according to the different requirements of physical resources. In recent years, the widely usage of Wi-Fi and Bluetooth provide make the development of gateway agents easier. And the gateway host is a control node designed to centralize the management of a number of gateway agents.

The private cloud is similar with a local control center, which has full access to all the information of robot system and full control to it. Besides, the service manager in the private cloud also provide management of database to ensure data security. For example, administrator can monitor the robot remotely, while users can only read general information of robots. The limited and specified access for users and administrators makes private clouds have high level of security.

![Fig. 12. CMS scenario specific for private CMfg service platform](image)

B. CMS scenario specific for public CMfg service platform

The public CMfg and community CMfg both provide service for a group of organizations. In the private robot Cloud, the robots, server and provide Cloud are connected inside the boundary of a manufacturing enterprise or company, while in a community robot Cloud, multiple enterprises share the data and knowledge collaboratively but sensitive information needs to be restrained within the enterprise scope [1]. The proposed system architecture for public cloud is shown in Fig. 13.
In this case, public/community cloud layer does not need the interfaces with higher layers. However, the interface for lower layers will be connected with multi organizations. These organizations join in the public cloud service to share information and other resources. To ensure stable and high-speed data transmission, the Data Pre-process and Sub-event Delegator need to be designed with strong data-handling capacity and good data stream management.

For the Gateway Layer of public cloud, the direct connection between the Gateway Host and public cloud layer is possible. This kind of solution is designed for the companies who build a public cloud and have physical resources as well. However, the more common solution is adding a service manager in the Gateway Layer, to select what can be accessed and controlled by public cloud. This is much better for system security.

Besides, those organizations are possible to be different companies or factories. The services they need from the cloud may vary from each other. Therefore, it is important for the system to be designed with the ability not only to handle different service requirements, but also distinguish the same service components and provide a standard solution for it. From the point of technique, the complexity of database, API, data pre-process and sub-event delegator all increase compared with private CMfg service platform.

C. CMS scenario specific for hybrid CMfg service platform

For the hybrid CMfg service platform, the whole system involves two or more kinds of CMfg service. As shown in Fig.14, the Hybrid Cloud is actually composed by Private Clouds and Public Clouds. This system is designed for high level development of CMS, when different cloud service providers need to cooperate with each other or larger companies absorb the small ones. The modularity, standardization and composability enables CMS can be recomposed to form more functional platform. For instance, this architecture can be applied when new cloud service appears and need to combine existed services.

Besides, the service manager of the private cloud plays a significant role of data filter. In the hybrid cloud, a larger amount of information is exchanged through network, which will expose confidential information to leakage risk. So it is very important to pre-process and encrypt data. The encrypt method and key also need to update frequently.
3.1.3 Work flow of Cloud Manufacturing

Wang [1] proposed a Smart Cloud Manager (SCM) to support manufacturing procedures. As manufacturing decisions are difficult to make due to complexity of manufacturing processes, variety of machines/devices, and uncertainty of resources status [1], a detailed logic flow which specify how the decisions are make is necessary. Based on that logic flow within SCM proposed by Wang [1], a modified one is proposed as shown in Fig.15.

The solution searching part of this logic flow or workflow of CMS has been adjusted, adding feasibility check. Not all the customers’ request has proper solutions in existed system. If the search engine doesn’t get good answers, in other words, the request is unfeasible, the SCM will send the request to service providers to seek possible solutions. Once proper solutions are responded, the capability of CMS or service database will be updated, and then the flow goes on the common way. If the request is beyond service providers’ ability and no response to the request is received, customers request will be declined.

When the feasibility check is positive, the solution list will be generated and sent back to customer. If the customer is not satisfied with the solution and want to suggest their own suggestions, the suggestion will also be sent to service providers to see if there any cooperation possibilities. Otherwise, the request will be modified, and the flow will start over from the beginning.

This modified workflow adds extendability to CMS. Customers’ requests will be satisfied in a better way. Besides, the mechanism provides an interaction channel between service providers and customers, which will help enterprises to formulate more precise market strategies and stimulates the innovation development. Moreover, capability of the CMS will be strengthened effectively with this mechanism.
Fig. 15. Workflow of CMS adapted from [1]
3.2 Developed CMS Specific for Robotics

As shown in Fig. 16, CMS architecture can be adjusted to robotics application. At the physical layer, devices are divided into robots and external sensors. Robots include the mechanical devices, robot controller and internal sensors. These internal and external sensors detect robots’ states and make visualization and close-loop control possible. The difference is internal sensors data can be captured through corresponding device agents. On the other hand, external sensors need separate controllers or computing devices, as external sensors like cameras or smart meters will occupy a lot of computing resources.

At the Gateway Layer, gateway agents are connected to robots and sensors through TCP/IP protocol. Based on the connectivity that gateway agents provide, a series customizes applications can be developed. As shown in Fig. 16, the application package integrates three most common applications: (i) data acquisition, (ii) data visualization, (iii) remote control. To satisfy the specific requirements of factory manager, cloud administrator, customers and etc., applications with targeted goals can be designed. Moreover, the application package can be easily ported to different platforms, neither local server of factory or cloud service.

The gateway host in Gateway Layer is designed to be the gateway manager, and provide other customized functions as well, such as robot data saving, data visualization, simulation, task allocating, remote control and etc.

The Cyber Layer structure is generally same with system in Chapter 3.1.1, but the specific part may need to be designed to be capable of handling robotics issues. For example, API for end users may require remote real-time monitoring function and service manager may require to provide robots data analysis function, even data mining or machine learning. The application package can also be applied in this layer.

Fig. 16. Developed CMS Specific for Robotics
3.3 Security Issues for CMS

As discussed in Chapter 2.3.1, a secure system has to fulfill five conditions: confidentiality, integrity, availability, authentication and accountability.

- **Confidentiality**: the data is understandable to the receiver only for all others it would be waste; it helps in preventing the unauthorized disclosure of sensitive information.
- **Integrity**: data received by receiver should be in the same form, the sender sends it; integrity helps in preventing modification from unauthorized user.
- **Availability**: Assuring that user has access to information anytime and to any network.
- **Authentication**: Assuring the identity of the entity involved in the communication.
- **Accountability**: Assuring that no entity can deny its participation in a data transfer between them.

Due to the vulnerability inside and outside the system, these security goals require the employment of a series of security mechanisms, services and devices. But vulnerabilities differ from each other and only specific actions are effective to the specific kind of vulnerabilities. Therefore, possible sources of system vulnerabilities need to be identified. Besides, corresponding solutions can be considered when design the system.

3.3.1 Security threat sources of CMS

As shown in Fig. 17, CMS is a semi-open system, the employment of Local Area Network (LAN) and Wide Area Network (WAN) open the door to external threats. Those interfaces designed for users and administrators and connections between layers, are also possible portals for hacking. If the vulnerabilities are exploited by attackers, it can lead to service disruptions, data loss, etc. For instance, misuse by authorized users and administrators or by hackers will make inappropriate changes to the database and system programs. Besides, attackers may also target availability of the cloud by camouflage and occupying computing resource. Therefore, the interfaces (HCI, API, SPI) need to be equipped with proper security mechanisms and services.

![Fig. 17. CMS structure from security perspective](image-url)
Except the possible external attack, errors or problems occur inside CMS will also threat system security. For example, if the gateway gets the control of CNC machines, how to protect machines from the danger of network viruses from the Cyber Layer or data errors imported from transmission.

### 3.3.2 Possible solutions

If CMfg is to be successful, it must be trusted by its users. Therefore, system security solutions need to be clarified to show how trust can be achieved for security and privacy.

#### A. Security Solutions for the Cyber Layer

As discussed in Chapter 3.3.1, interfaces that connect the Cyber Layer and outside are the portals. There are three kind of interfaces as shown in Fig. 17:

- HCI: Higher-level Cloud Interface (Data Integration & Event Receiver)
- API: Application Programming Interface
- SPI: Service Provider Interface (Data Pre-process & Sub-event Delegator)

Firstly, all of the interfaces need security mechanisms to isolate cyber core from external attack, such as fine-grained access control and user authentication. Furthermore, although service composition is simplified in CMfg, some services might have a malicious source. The Data Pre-process part in SPI should evaluate the reliability of service source and decline the potential harmful request.

Secondly, external threats are not the only resource. As customers shift to cloud models, they shift their focus from systems (which they used to control) to data and how that will be treated by other entities on their behalf [25]. It is the same for both service providers and service requestors, requiring adequate protection to the service data and personal data. Even problems happen, business continuity should be ensured, which requires recovery and backup mechanism. Moreover, legal frameworks have been instrumental and key to the protection of users’ personal and sensitive information [25]. For example, European law limiting cross-border data transfers also might prohibit the use of cloud computing services to process this data if data would be stored in countries with weak privacy protection laws, and notification may be required [35].

In general, in the cloud or Cyber Layer, establishing risks and obligations, implementing appropriate operational responses and dealing with regulatory requirements are more challenging than with traditional server architectures [25]. But comprehensive security actions can prevent attack or problems effectively.

#### B. Security Solutions for the Physical Layer

The Gateway is the connection between the Cyber Layer and Physical Layer. To ensure data security, encryption of data, at rest, in transit, and eventually in use, becomes a basic requirement, yet it comes with a performance cost (penalty) [34]. The encryption algorithms can be divided into two groups: symmetric and asymmetric ones. For example, International Data Encryption Algorithm was proposed by James Massey and Xuejia Lai in 1991, which is considered as the best symmetric key algorithm [27].

Other trust techniques include signing, machine authentication, secure key storage, and attestation [34]. To keep information secure, the key should be with the client only and not
stored on the cloud itself, and to prevent key loss, the client would need to have a backup copy of the data, somewhere else, or even off-line backup [28].

Besides, as the task will delegated from the Cyber Layer to the Physical Layer may be unpractical or harmful for devices, it is important for the Gateway to distinguish tasks or orders. Moreover, when conflicts exist, which part takes the priority? This should also be clearly clarified.
Chapter 4: Implementation and system development

The concrete implementation is to develop a software component which make it possible to:

- Set Application program interfaces (APIs) with Universal Robots via the controller.
- Establish Gateway integration between Universal Robots and KTH cloud.
- Design proper modularized system architecture for following development of cloud database and security mechanism.

Based on the provided Real-Time Data Exchange (RTDE), a Python client named Real-Time Remote Control (RTRC) is developed. The RTDE synchronize external executables with the UR controller, for instance URCaps, without breaking any real-time properties [36].

![Program flow chart](chart.png)

Fig. 18. Program flow chart

As discusses in Chapter 3.2, RTRC includes the Gateway Agent part and Application Package part. The Gateway Agent part is realized by connection build subprogram. The Application Package part consists of data acquisition, data visualization and robot control. The software construction is shown in Fig.18, and corresponding methodology, implementation and functionality examining are presented in the following sections.
4.1 System connectivity

4.1.1 Methodology

A. Coding with Python language

RTDE clients can be implemented in different languages that support socket communication, and the purpose of this RTDE client library, written in Python, is to provide an easy starting point and show some example applications [36]. Some functionality has been developed for Python 2.7.11, which is a mature open-source language. Based on these existed function, other program modules are developed.

B. Building connection with TCP/IP protocol

UR robot can communicate with outside equipment through TCP/IP (Transmission Control Protocol/Internet Protocol) protocol [37]. Each robot is assigned with a unique network address. TCP/IP also ensures the confidentiality and integrity of data communication. Besides, as TCP/IP protocol does not rely on specific network transmission hardware, it can be integrated in various kinds of network, such as Ethernet, Token Ring Network, Token Ring Network and etc. This provide the RTDE client a wider application possibility.

C. UI developed by Qt Creator

To match with the Python environment, Qt Creator is chosen to be the user interface developing tool. Qt Creator is a cross-platform C++, JavaScript and QML integrated development environment which is part of the Software Development Kit (SDK) for the Qt GUI (Graphical User Interface) Application development framework [38]. The GUI layout and forms designer it provides, can help to develop the user interface conveniently. Then transform it to a Python file and combine it with the main program.

4.1.2 Implementation

The RTDE interface is by default available when the UR controller is running [36]. The first functionality is to build the connection between the UR robot and the gateway/computer.

The RTDE functionality has provide the setup procedure. On connection to the RTDE interface, the client is responsible for setting up the variables to be synchronized. Any combination of input and output registers can be specified. To achieve this the client sends a setup list of named input and output fields that should be contained in the actual data synchronization packages. The definition of a synchronization data package format is referred to as a recipe [36].

In return the RTDE interface replies with a list of the variable types or specifies that a specific variable has not been found. Each input recipe that has been successfully configured will get a unique recipe id. The list of supported field names and their associated data types can be found below. When the setup is complete the data synchronization can be started and paused [36]. And the using instruction of RTDE is shown in Table 1.
Real-time Data Exchange (RTDE)

<table>
<thead>
<tr>
<th>Transmit</th>
<th>Robot State messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver</td>
<td>Various data</td>
</tr>
<tr>
<td>Port no.</td>
<td>30004</td>
</tr>
<tr>
<td>Frequency/Hz</td>
<td>125</td>
</tr>
</tbody>
</table>

Table 1. RTDE using instruction [39]

Based on the setup procedure, the connection function is developed. The detailed program flow chart is shown in **Fig. 19**.

![Program flow chart of connection part](image)

**Fig. 19.** Program flow chart of connection part

Since the program is executed, all the global variables are initialized. The start connecting and stop connecting part are executed with two parallel threads. Once the buttons are clicked, the
threads will be triggered. As the buttons are clickable all the time, it is necessary to check the present state before reaction, in case that user makes some improper operation.

Firstly, the connection state needs to be checked. If the connection with robot has been built, a warning will be given to the user to show that connection has been built and only one connection is available at the same time. If not, the user inputs which specifies robot ID address, port number and data configure file will be read. Sequentially, the connection can be set up. If any error happens during the connection building process, the connection will be cancelled and the corresponding warning will pop up. Otherwise, the connection is built successfully and the message will be shown in the log menu. The connection state variable will also be set to True.

Similarly, if the Stop Connect button is clicked, the connection state will be check first to see if it is connecting with robot now. If no connection is built, a warning will pop up to remind user that it is an improper operation. Otherwise, the connection will be stopped immediately.

4.1.3 Functionality examining

The user interface is shown in Fig.20. In the left section some default settings for UR robot IP address (Internet Protocol Address), port number and configure file have been written.

As shown in Fig.21, when the Connect or Stop Connect button is clicked, and the operation is finished, corresponding information will show in Log with the operation time.

4.2 Data acquisition

4.2.1 Methodology

A. Data acquisition based on RTDE

The data acquisition function is developed based on the RTDE. The RTDE interface provides a way to synchronize external applications with the UR controller over a standard TCP/IP connection, without breaking any real-time properties of the UR controller [36]. The synchronization is configurable and can for example involve the following data:
• Output: robot-, joint-, tool- and safety status, analog and digital I/O's and general purpose output registers;
• Input: digital and analog outputs and general purpose input registers [36].

The record output data from robot will be save in CSV (Comma Separated Values) format for analysis or other usage.

B. Data transition and storage with CSV file output

As the robot data is saved to a CSV file with serialized data, which is not easy to read, it is necessary to pre-process it and transform the data to a more-friendly way with necessary data instructions. Moreover, the robot data file is named by the start-recording date, which is helpful for the package process and data relocation.

4.2.2 Implementation

The RTDE client works in a synchronization loop. When the synchronization loop is started, the RTDE interface sends the client the requested data in the same order it was requested by the client. Furthermore, the client is expected to send updated inputs to the RTDE interface on a change of value. The data synchronization uses serialized data [36].

The RTDE generally generates output messages on 125 Hz. To make it adjustable the program loop is changed and provides 8 choices, from 125Hz to 1Hz. The detailed program flow chart is shown in Fig.22. Start recording and stop recording are executed in two parallel threads. Similar with the connection part, check the client state first. If it is recording now, the shown a warning to user to remind that the operation is not proper.

If the user input “Samples” is not changed by users, which means “Samples” equals zero, the client will keep recording robot data until it is stopped by users or any error happens. If “Samples” is specified, the client will only record the specified numbers of robot data.

Similar with the Connection Build part, a series of operation checking will be done before starting data record. As the robot connection is only available for one client at one time, only one data recording process can only be executed.

Besides, during the recording process, the real-time sample numbers or sample process will be shown with the time with 1Hz refreshing frequency. This is designed to inform user the recording process.

After the recording process is finished, there will be a data-file-transformation module to export the CSV file to XML file in a readable format. The time stamp that recording is started, robot version, sample frequency and sample numbers will be recorded in the file header. The XML file is also named as the recording start time. As the absolute timestamp is difficult to understand for users, the time column is transformed to relative timestamps. In other words, the data file provides the time stamp that recording is started and how many seconds has passed since the recording is started.

As the Universal Robot provide highly detailed robot data, both electrical and mechanical parameters, this data acquisition enables various functions development, such as real-time robot availability, execution status monitoring both locally or remotely, robot program simulation and etc.
4.2.3 Functionality examining

The user interface is shown in Fig. 23. The output file will be named with default names in a special folder. If the user specifies the output file name, it will also be copied and saved with the name.
For example, we want to record robot data with 25Hz sampling frequency and totally 200 samples. The inputs are set as Fig.24 shows and Log is shown in Fig.25.

![Fig. 24. Screen shot of data recording operation](image1.png)

![Fig. 25. Screen shot of log after data recording operation](image2.png)

![Fig. 26. Screen shot of warning](image3.png)

After the operation is down, open the XML file. As shown in Fig.27, the header records the information about start recording time, sampling frequency and sample numbers. To shown the total data rows, rows from the 13th to 206th are hidden. The start row number is 10 and the end row number is 209, which means 200 samples are recorded as planned. Besides, the time interval between two samples is 0.04 second, which means the sampling frequency is 25Hz.

![Fig. 27. Screen shot of XML file](image4.png)

**Fig.28** is the screen shot of the teaching pendant and on the Log page, which shows some of the most commonly used robot parameters. At the same time, robot data are recorded into a
XML file, shown as Fig. 29. And the data row recorded at the same time stamp with robot Log is highlighted with yellow background. Five robot data were picked out to examining the data authority. It is assumed that the data that shown in robot’s Log are right and accurate enough to be a reference.

Fig. 28. Screen shot of UR5 robot log

Fig. 29. Screen shot of recording data file

Table 2 shows the summary results for Fig. 28 and Fig. 29. As all of rounded value of the picked data recorded in XML file are equal to the data shown in robot Log. It verifies the integrity of Data Recording function.

<table>
<thead>
<tr>
<th>No.</th>
<th>Data Name</th>
<th>Data shown on teaching pendant</th>
<th>Data recorded in XML file</th>
<th>Integrity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joint control current: Elbow</td>
<td>1.7 A</td>
<td>-1.6724 A</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>Joint temperature: Wrist3</td>
<td>31.2 °C</td>
<td>31.2285 °C</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>Actual main voltage</td>
<td>48.1 V</td>
<td>48.1302 V</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>Actual joint voltage: Base</td>
<td>47.7 V</td>
<td>47.7270 V</td>
<td>✓</td>
</tr>
<tr>
<td>5</td>
<td>Actual joint voltage: Elbow</td>
<td>47.9 V</td>
<td>47.8640 V</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 2. Data recording results summary
4.3 Data visualization

4.3.1 Methodology

As the output and input data of robot are available with adjustable frequency, it is possible to visualize these data in a more understandable form. For example, monitoring of the power consumption status, robot I/O manipulating and etc. This data visualization function is designed to plot robot data to help better understand and analysis of data.

A. Data reading with python csv library

As Python has well-developed csv module, it is easy to read csv files to get specified robot data. The RTDE client has provided a program module named csv_reader, which makes the plotting function feasible.

B. Plotting line graph with Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms [40]. Matplotlib can be used in Python, so proper graphics can be drawn after read the time stamp and corresponding robot data. The most common chart is line chart for robot monitoring. For example, the factory may need real-time surveillance of robots’ power consumption. In this case, the real-time data plotting can be applied.

C. Multi Thread to ensure instant interaction

To ensure the instant reaction to user’s requests when part of the program is running, like data plotting or record, multi thread is used in the program. The main thread will take charge of user interaction, and the sub-threads are responsible for different sub-tasks.

D. UI developed by Qt Creator

The data visualization integrates two functions: history data plotting and real-time data plotting. To shown it in a friendly way, a user interface has been developed as well.

4.3.2 Implementation

A. History Data Plotting

With Matplotlib library, data plotting is easy to realize. Most of the program is used to prevent misuse. If everything is right, a picture that shows how the specific robot parameter vary over time will be pop up. Some of the parameters such as joint temperature, actual joint velocities, actual joint positions are six-dimensions vectors. So joints need to be specified as well. Those six-dimensions vectors are named with suffix: “-6”. The detailed program flow chart is shown in Fig.30.
Fig. 30. Program flow chart of history data plotting part

B. Real-time Data Plotting

The real-time data plotting part is a little bit different from the history data plotting. The program flow chart is shown in Fig.31. As the real-time data need to be transported from the robot instantly, the connection has to be built first. When new data is received, they will be appended to the old data array and the line chart will be refreshed with particular frequency.
4.3.3 Functionality examining

A. History Data Plotting

The user interface is shown in Fig. 32. Data visualization is split into two pages, one for History Data Plotting and the other for Real-time Data Plotting.

In History data file, the data file path need to be specified. Then choose the robot data that you want to plot. If it is a six-dimensions vector, the joint need to be specified as well. If not,
some warnings will pop up. As shown in Fig.33, the operation is done to plot actual elbow velocities in file robot_data.csv. And the line graph is shown in Fig.34.

![Fig. 33. Screen shot of history data plot operation](image33.png)

![Fig. 34. Screen shot of line chart of actual elbow velocities](image34.png)

**B. Real-time Data Plotting**

![Fig. 35. Real-time data plotting part UI](image35.png)
The user interface of real-time data plotting is shown in Fig. 35. Similar to the recording part, the sampling frequency is adjustable from 125Hz to 1Hz. The time span is also changeable from 0 minutes to 10 minutes and the default setting is 0. The resolution of time span is 0.1 minute.

To plot the real-time robot data, the connection has to be built first, otherwise a warning will be shown. After the sampling frequency, time span and robot data are specified, as shown in Fig. 36, the line graph can be plotted as shown in Fig. 37. Besides, it is easy to zoom in or out with the tools shown on the tool bar of the graph.

**Fig. 36.** Screen shot of real-time data plotting operation

**Fig. 37.** Screen shot of line chart of actual robot voltage
4.4 Remote control/program

4.4.1 Methodology

A. Connection building with Socket Connection

Programming in the teaching pendant or import blocks of Script code into robot is the most common situation. To realize remote programming, a connection to send Script file needs to be established. [41] introduced how to make a script that entirely control the robot remotely from a host computer by Script programming. In this case the program language Python and the Python environment are used to communicate with the Robot, especially the Socket library. It is possible to send a script file through the Ethernet to the TCP port 30002 on the UR robot. As all the script file can be sent, the remote control is functionally same with local programming.

B. UI developed by Qt Creator

The user interface that enable users to control robot remotely is also designed by Qt Creator. All of other the functions (Connection build, Data acquisition and Data visualization) are arranged on one page, and the control functions are integrated on another.

4.4.2 Implementation

![Program flow chart of digital output set part](image)

Fig. 38. Program flow chart of digital output set part
For convenience, three most commonly used functions are developed with separate sub-windows. The first one is to set digital output. There are eight digital outputs and other I/O in the UR robot control box. These controller I/O are electrical interfaces with other equipment. Therefore, easy access with I/O is helpful for equipment installation and debugging, such as external sensors installation.

As shown in Fig.38, the work flow of Script Programming Part starts from initialization, then build socket connection and send Script. If messages received, it means the operation is done successfully. Otherwise, a warning will pop up to remind users of possible errors. Besides, if the robot need to be programmed to do a repetitive work, the loop can be written in the Script.

Fig. 39. Program flow chart of joints movement control part
The second function is Joints Movement Control Part, to move robot’s joints with specific velocities. In robot move screen the joint positions expressed in degrees i.e. -360 to 360 degrees. With the movement of sliders, corresponding joints of robot will spin specific degree.

As shown in Fig.39, the work flow of Joints Movement Control Part starts from initialisation and split into two sub flows. The left one is about building socket connection with UR robot. The right on is the main work flow, which will check connection state first. If no connection has been built, a warning will pop out. Otherwise, robot’s beginning state will be approached by reading joints’ position data and the sliders will be initialized. Then once any of these control sliders changed, the robot will be control to realize corresponding movements and renew joints’ position. In the procedure of moving, the sliders will be disabled. This is designed to prevent the robot from crashing with itself or getting stuck.

To realize the full functional remote control, a sub-window for Script input is also designed, to control the robot to do some complex work. Detailed script programming method can be found in the Universal Robot URscript manual. The work flow of Script Programming Part is same with Digital Output Set Part.

4.4.3 Functionality examining

A. Digital Output Set Part

As stated above, the Control function is composed of three sub-parts. The digital output set interface is shown in Fig.40.

![Fig. 40. Digital output set part](image)

To set one or several digital outputs true, click the checkbox and then click “Set true” button. For example, digital outputs 1,3,5,6,7 need to be set true. The interface shows like Fig.41. After the button is clicked and operation is finished, the log will provide the records of setting, as shown in Fig.42. Besides, new digital outputs state can also be checked on the monitoring screen, as shown in Fig.43.

![Fig. 41. Digital output set part with specific setting](image)

![Fig. 42. Log for digital output set](image)
B. Joints Movement Control Part

As shown in Fig. 44, the interface of joints movement control part is composed of sliders to set specific degree of every joint and moving velocity.

After “Connect” button is clicked, the actual degree of every joint will be shown in the right side. And the default setting for joints moving velocity is 0.1.

![Joints movement control part with specific setting](image)
Once the “Start Control” is clicked, the remote control is enabled. UR robot will be control to move corresponding to sliders. As shown in Fig.45, the “Start Control” button will also be changed to “Stop Control”, which means the second click will stop the remote control.

C. Script Programming Part

As shown in Fig.46, the interface of script programming part is simpler than the other two. It is composed of script input and two buttons to clear script of send script.

![Fig. 46. Script programming part](image)

**Fig. 46.** Script programming part

**Fig.47** shows the script programming example and results when setting digital output with this function.

![Fig. 47. A script programming example](image)

**Fig. 47.** A script programming example

After the script is sent and received by robot successfully, the news will pop out in the Log part, as shown in Fig.48. And the execution results shown in robot’s reaching pendant is same as planned. Digital output 0 has been set to TRUE.

![Fig. 48. Screen shot of log for script programming](image)

**Fig. 48.** Screen shot of log for script programming
Fig. 49. Screen shot of robots I/O monitoring UI
Chapter 5: Robotic case studies and experiment results

This chapter describes the validation experiments and testing results of the developed system in different cloud applications. The first section introduces the test items of the Real-Time Remote Control (RTRC) and corresponding methods. The flowing three section handle the specific application of RTRC in private, community and hybrid cloud, and the experiments results are explained.

5.1 Test method

A complete software development process includes a comprehensive test and evaluation. There are various assessment methods. For example, criteria-based assessment is a quantitative assessment of the software in terms of sustainability, maintainability, and usability, which are derived from ISO/IEC 9126-1 Software engineering — Product quality [42]. The assessment involves checking whether the software, and the project that develops it, conforms to various characteristics or exhibits various qualities that are expected of the software [42]. On the other hand, tutorial-based evaluation provides a pragmatic evaluation of usability of the software in the form of a reproducible record of experiences [43].

Considering the development of the RTDE client with UR robot is still on the beginning stage, several terms from user perspective are selected to realized a based software evaluation/assessment.

The first one is functionality examining, which is used to check whether the software can realize the function it aims to do. Secondly, response speed is important for quality evaluation of TCP/IP connection between the client and robots. Last but not least, reliability is also chosen to be assessed, which is measured in terms of mean time between failures (MTBF) [44]. Detailed test plan as follows.

5.1.1 Functionality examining

To test if the software realizes functions that it was designed to do, a series of simulating operations that users may do need to be completed. It also includes improper operations. The software should be designed to have ability to distinguish improper operations and guide users to do the right ones.

So the first step is to check if all the designed functions realized. The second step is to operate with wrong sequence or method and check if the software reacts properly to wrong operations.

5.1.2 Response delay

Software response delay is a characteristic that influences users’ experience significantly. Many factors decide the length of response delay directly or indirectly, such as the design of programs, connection quality of network, working state of robots and servers, stochastic errors and etc.

In this project, the design of programs and limitations of robot itself are two points that are focused on. For example, a good program design will decrease the length of some complex
codes, which will decrease the response delay. Besides, the TCP/IP connection with robot has its own speed and bandwidth limitations.

Therefore, two functions are chosen to do the response delay test. The first one is Joints Movement Control, where longer response delay may lead to bad operability. The second one is Connection Build, which is mostly decided by the TCP/IP protocol and UR robot’s system limitation.

To acquire a convincing response delay of this two functions. A number of repetitive test need to be done and the results need to be analyzed by statistics methods to eliminate the effect of random factor.

5.1.1 Reliability

Most of these designed software functions have been realized reliably, which means failure occurs rarely, such as Robot Data Record, History Data Plotting and etc. Compared with other functions, Real-time Data Plotting fails more frequently. So the reliability test will be applied to Real-time Data Plotting to evaluate it and analysis possible reasons.

However, Real-time Data Plotting is not a continuous operation. So the mean time between failures cannot be applied. Based on the concept, the complement of average failure rate can be used to represent the reliability. If no error occurs in 10mins when Real-time Data Plotting is operated, it is assumed as one successful operation. Otherwise, it fails.

5.2 A case study based on a private cloud

When the Real-Time Remote Control (RTRC) is employed in a private cloud, the cyber layer does not need to interact with higher layers. It is managed by the organization it serves. The private cloud is built in local server or private server.

5.2.1 Implementation method

Take a small factory that are fabricated with several robots as an example. The factory needs a private cloud which is designed to mainly handle inside-factory information. In this case, the simplest realization is to use one PC to act as local server and gateway. As shown in Fig.50, both the Gateway Agent and Application Package will be run on this PC, so that full of Physical layer and Gateway layer are built.

The Gateway connection is built through Wi-Fi. In this case, a number of robots can be connected with and controlled by the PC. Adding necessary Cyber layer components, a whole private cloud environment can be built and central management can be realized as well.
5.2.2 Experiment results

A. Functionality examining

Actually, the function test has been done in the Functionality Examining in Chapter 4.1–4.4. The main and sub functions of RTDE client and the test results are listed in Table 3.

<table>
<thead>
<tr>
<th>No.</th>
<th>Main Functions</th>
<th>Sub Functions</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>①</td>
<td>Connection Build</td>
<td>Connection Build</td>
<td>✓</td>
</tr>
<tr>
<td>②</td>
<td>Data Acquisition</td>
<td>Robot Data Record</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data File Transformation</td>
<td>✓</td>
</tr>
<tr>
<td>③</td>
<td>Data Visualization</td>
<td>History Data Plotting</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Real-time Data Plotting</td>
<td>✓</td>
</tr>
<tr>
<td>④</td>
<td>Remote Control</td>
<td>Digital Output Set</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Joints Movement Control</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Script Programming</td>
<td>✓</td>
</tr>
</tbody>
</table>

*Table 3. Functionality Examining Result of Private Cloud Case*

B. Response delay
The test results of response delay are shown in **Table 5**. Response delay of Connection Build is the time that starts from the UI getting the connection demand and ends at connection with robot being built successfully, which is closely related to the TCP/IP protocol and UR robot’s system limitation. The second response delay is for Joints Movement Control, which is the time starts from the UI receiving control demand and ends at any joints of robot starting movement.

<table>
<thead>
<tr>
<th>Response delay</th>
<th>Connection Build</th>
<th>Joints Movement Control</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Samples number</strong></td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>0.1044 s</td>
<td>0.1728 s</td>
</tr>
<tr>
<td><strong>Minimum</strong></td>
<td>0.0078 s</td>
<td>0.0545 s</td>
</tr>
<tr>
<td><strong>Maximum</strong></td>
<td>0.4426 s</td>
<td>0.4836 s</td>
</tr>
<tr>
<td><strong>Variance</strong></td>
<td>0.0115 s²</td>
<td>0.0096 s²</td>
</tr>
</tbody>
</table>

**Table 4.** Test Results of Response Delay of Private Cloud Case

As shown in **Table 4**, the fluctuation of the response delay is large. This is may caused by instability of program execution and Wi-Fi connection, and it is correlated with the computing ability of physical instruments as well, such as the PC and robot. However, the average response delays are both short enough to ensure fluent usage of the software.

**C. Reliability**

<table>
<thead>
<tr>
<th>Functions</th>
<th>Real-time Data Plotting</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Plotting frequency</strong></td>
<td>125Hz</td>
</tr>
<tr>
<td><strong>Operating numbers</strong></td>
<td>30</td>
</tr>
<tr>
<td><strong>Failed operation numbers</strong></td>
<td>3</td>
</tr>
<tr>
<td><strong>Failure Rate</strong></td>
<td>10%</td>
</tr>
<tr>
<td><strong>Reliability</strong></td>
<td>90%</td>
</tr>
<tr>
<td><strong>Average operating time when failure occurs</strong></td>
<td>263.49 s</td>
</tr>
</tbody>
</table>

**Table 5.** Test Results of Reliability of Private Cloud Case

The test result of reliability of Real-time Data Plotting in private cloud case is shown in **Table 6**. All tests are finished with same plotting time span: 1min. In 30 times of simulating operations of 125Hz plotting frequency, there are 3 times that operating time last shorter than 10 minutes, and the average time is 263.49s. As discussed in test plan, the reliability is 90%. The reliability has an increasing trend when plotting frequency decreases, as shown in **Fig.51**.
The computing load increase sharply when plotting frequency increases as the data buffer increases and the figure plots more precisely. When the computing resource is limited, especially the PC or robot has other work at the same time, the error rate increases.

To solve this problem, better plotting method that consumes less computing resource or more powerful processor may help.

![Reliability of Real-time Data Plotting](image)

**Fig. 51.** Reliability of Real-time Data Plotting with Different Plotting Frequencies

### 5.3 A case based on a public cloud

In this case study, the Real-Time Remote Control (RTRC) is developed with a public cloud. The difference between the private cloud and public cloud is that public cloud is a proprietary network that provides scalable services. The network needs to be built on a public platform.

#### 5.3.1 Implementation method

KTH Cloud is selected for this case study to assess the practicality of RTRC developed with a public cloud. **Fig.52** explains the system architecture of this case, which the Gateway Agent and Application Package are run in the KTH Cloud platform. Besides, KTH Cloud serves other research project as well. It simulates the public cloud environment for cloud manufacturing.

The access of KTH Cloud can be achieved through any computing terminal with internet, and the detailed realization is similar with private cloud case. The difference is that the server is from a public cloud instead of local instruments.

The Gateway connection is built through Wi-Fi as well. In this case, robots belong to one company or factory can be connected with and controlled by one VMware built in KTH Cloud. There are other VMware developed for other companies, factories and organizations as well.
5.3.2 Results analysis

A. Functionality examining

The function test process of public cloud case is same with private cloud case. The main and sub functions of RTDE client and the test results are listed in Table 6.

<table>
<thead>
<tr>
<th>No.</th>
<th>Main Functions</th>
<th>Sub Functions</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>①</td>
<td>Connection Build</td>
<td>Connection Build</td>
<td>✓</td>
</tr>
<tr>
<td>②</td>
<td>Data Acquisition</td>
<td>Robot Data Record</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data File Transformation</td>
<td>✓</td>
</tr>
<tr>
<td>③</td>
<td>Data Visualization</td>
<td>History Data Plotting</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Real-time Data Plotting</td>
<td>✓</td>
</tr>
<tr>
<td>④</td>
<td>Remote Control</td>
<td>Digital Output Set</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Joints Movement Control</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Script Programming</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 6. Functionality Examining Result of Public Cloud Case

B. Response delay
Table 7 shows the test results of response delay of public cloud case. The fluctuation of the response delay is large. This is may caused by instability of program execution and Wi-Fi connection, and it is correlated with the computing ability of physical instruments as well, such as the PC and robot. However, the average response delays are both short enough to ensure fluent usage of the software.

<table>
<thead>
<tr>
<th>Response delay</th>
<th>Connection Build</th>
<th>Joints Movement Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>Samples number</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>Average</td>
<td>0.1737 s</td>
<td>0.1933s</td>
</tr>
<tr>
<td>Minimum</td>
<td>0.0089 s</td>
<td>0.0731s</td>
</tr>
<tr>
<td>Maximum</td>
<td>0.5072s</td>
<td>0.5379s</td>
</tr>
<tr>
<td>Variance</td>
<td>0.0213s²</td>
<td>0.0174s²</td>
</tr>
</tbody>
</table>

Table 7. Test Results of Response Delay of Public Cloud Case

C. Reliability

<table>
<thead>
<tr>
<th>Functions</th>
<th>Real-time Data Plotting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plotting frequency</td>
<td>125Hz 63Hz 25Hz 1Hz</td>
</tr>
<tr>
<td>Operating numbers</td>
<td>30 30 30 30</td>
</tr>
<tr>
<td>Failed operation numbers</td>
<td>2 1 1 0</td>
</tr>
<tr>
<td>Failure Rate</td>
<td>6.7% 3.3% 3.3% 0%</td>
</tr>
<tr>
<td>Reliability</td>
<td>93.3% 96.7% 96.7% 100%</td>
</tr>
<tr>
<td>Average operating time when failure occurs</td>
<td>274.03 s 419.83 s 473.61 s /</td>
</tr>
</tbody>
</table>

Table 8. Test Results of Reliability of Private Cloud Case

5.4 A case study based on a hybrid cloud

In this case study, the application of Real-Time Remote Control (RTRC) in a hybrid cloud is discussed, in order to imitate the practical application better. The hybrid cloud integrates not only private clouds but also public clouds.

5.4.1 Implementation method

According to the discussion in the private and public cloud case study, the system architecture is proposed in Fig.53. As illustrated, the hybrid cloud is a higher level cloud which may integrates public cloud and private cloud.
When several private clouds or public clouds need to be reorganized into a new one, the hybrid cloud, the main part of RTRC (Application Package) can still be used. The Application Package can be employed in any cloud platform when needed. However, the Gateway Agent does not need to be copied. It is only needed in the lower layer to provide the connection between physical sources and private cloud.

The data transmission between private clouds and public clouds employs internet transmission or WAN. For example, web end is one of the most popular way to share information remotely. The request-response patterns can be applied for the two-way communication between private and public clouds. Based on the effective and high-transmission-rate connection, the higher level cloud can get access of this by the permission of lower level cloud systems. The permission is managed by cloud Service Management.

**Fig. 53. System Architecture of Hybrid Cloud Case**

Besides, this kind of system architecture makes the recombination of different level of clouds possible. When new cloud service appears, it is easy to absorb it into a relatively comprehensive cloud service platform. It is also helpful for the cooperation of clouds across different fields.

Due to the limited time, the web end has not been developed in this project, so only the application of RTRC in a hybrid cloud is presented.
Chapter 6: Conclusions and future work

This chapter summarizes the conclusions, the limitations and the future work related to develop approaches in this project. Based on the results in Chapter 5, several conclusions have been found. These conclusions are reported in the first section of the current chapter. The second section identifies the limitations of the current implementation that needs further improvements, and the last section states the plans for the future work.

6.1 Conclusions

This master thesis project is focused on developing an industrial Cloud Robotics System and evaluate the connectivity and security. Cloud Manufacturing System scenarios specific for private, public and hybrid cloud are proposed. The characteristics of each scenario and the difference between them are presented. Then, system architecture and work flow designed for Cloud Robotics are presented. According to the proposed system architecture, possible security sources and possible solutions are discussed.

A python software is developed for the Universal Robot 5 to construct a simplified Cloud Robotics System. The software consists of two modularized part, Gateway Agent and Application Package. The Gateway Agent realizes the connection between the Universal Robot 5 (UR5) and the cloud, while the Application Package can be customized according to specific application and requirements.

To evaluate connectivity and stability, private robotics cloud system and public robotics cloud system are simulated with KTH Cloud. The hybrid robotics cloud system is discussed as well. The results of the case study show that:

1. The connectivity of Cloud Manufacturing System can be realized by API. The customized API is a good solution for various legacy robots and machines, to connect them with Cloud.

2. Secondly, the proposed scenarios can realize system integration effectively. The system architecture improves the system modularity, standardization and composability, which makes it easier to expand and update system. It also has high level of compatibility.

The novelty of this work includes:

1. proposing new system CMS architecture for better modularity, standardization and composability;

2. employing Universal Robot in CMS;

3. developing a private Cloud Robotics case;

4. discussing the application of developed API (RTRC) in public and hybrid CMS

6.2 Limitations of the project

A few limitations were identified during the study and they can be summarized in the following points:

- The security solutions for Cloud Robotics System are not realized and tested.
• The software evaluation covers a series of items related software performance. In this project, only three aspects are selected to discuss detailedly. It is not very comprehensive.

• The proposed system architecture is verified through the case study, but no practical production cases are provided.

• There are several robotic applications like ROS are widely used nowadays. Therefore, the CRS also need to consider the integration with other applications, not only RTRC.

6.3 Future work

The future work for the Cloud Robotics Systems will be centered on the following aspects:

• Developing an effective security system to enhancing data security.

• Improving the evaluation methods of CRS for further development.

• Enhancing the composability of CRS to make it possible to integrate other robotics applications.
References


