Master's Thesis

Rapid Control Prototyping for Reconfigurable Assembly Workstations

Bo-Bae Kim

Department of System Design and Control Engineering

Graduate School of UNIST

2017

Rapid Control Prototyping for Reconfigurable Assembly Workstations

Bo-Bae Kim

Department of System Design and Control Engineering

Graduate School of UNIST

ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

Rapid Control Prototyping for Reconfigurable Assembly Workstations

A thesis
submitted to the Graduate School of UNIST
in partial fulfillment of the
requirements for the degree of
Master of Science

Bo-Bae Kim

01. 17. 2017

Approved by

Advisor

Duck-Young Kim

ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

Rapid Control Prototyping for Reconfigurable Assembly Workstations

Bo-Bae Kim

This certifies that the thesis of Bo-Bae Kim is approved.

01. 17. 2017

signature

Advisor: Duck-Young Kim

signature

Daeil Kwon

signature

Sang Hoon Kang

Abstract

Diverse customer demands and rapid technology change have led to a paradigm shift in the manufacturing industry, from mass production to mass customization, and eventually to personalization. In the past, manufacturers have faced a challenge to produce a large volume of a product at low cost. Today, they should however produce a very small volume of a highly personalized product at mass production cost. In order to meet these challenges, rapid configuration or reconfiguration of manufacturing systems are crucial. Therefore, many studies have discussed reconfigurable manufacturing systems, emphasizing on dynamic scheduling and flexible shop floor logistics. However, little attention has given to the hardware control and the corresponding software development, although they are very important and time-consuming tasks for manufacturing system reconfiguration.

Therefore, the main objective of this paper is to quickly design, test, and verify the control software both in a virtual and in a real environment. To do this, we propose a procedure of rapid control prototyping consisting of virtual factory construction, control software development and a final calibration procedure. Rapid control prototyping facilitates engineers to quickly develop control software including communication inputs and outputs, prior to constructing a real shop floor. The proposed simultaneous procedure of manufacturing system design and its control software development will significantly reduce the reconfiguration time of a manufacturing system.

Table of Contents

List of Figures

List of Tables

I. Introduction

1.1 Background

Large fluctuations in product demands, changes in customer preferences, and new government regulations generally lead to highly heterogeneous products. In the past, industries focused on achieving cost reduction and quality control. However, in recent years, the goal of industries has changed to having a wide variety of products.

Owing to short product lifecycles, many industries have been shifted from mass production to mass customization, and eventually to personalization (Koren et al., 1999). The present shift to highly heterogeneous products means that manufacturers need to produce high-quality and personalized products at mass production cost. To respond to various changes faster and more cost-effectively, manufacturing systems should focus on the following:

- **Responsiveness**: to large fluctuations in product demands, customer preference changes, and new regulations of government;
- **Retrofit**: for new technologies and product introduction; and
- **Resilience**: to severe faults.

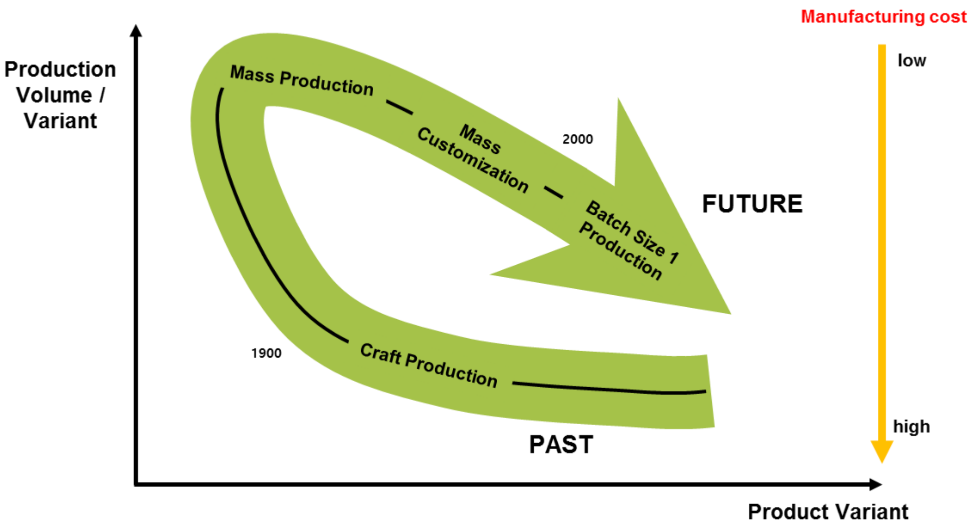

Figure 1.1 Manufacturing paradigm shift and manufacturing cost change, adopted from Koren et al., 1999

1.2 Motivation

In order to payback the machine and factory installation costs during periods of reduced orders, a factory must produce a wide variety of products to meet customer needs. To survive this new paradigm for manufacturing, fast and cost-effective manufacturing systems must be provided to manufacturers. This has led to the emergence of Reconfigurable Manufacturing Systems (RMS) in order to determine the hardware and software that are reconfigurable to increase the manufacturing system life cycle (ElMaraghy, 2005).

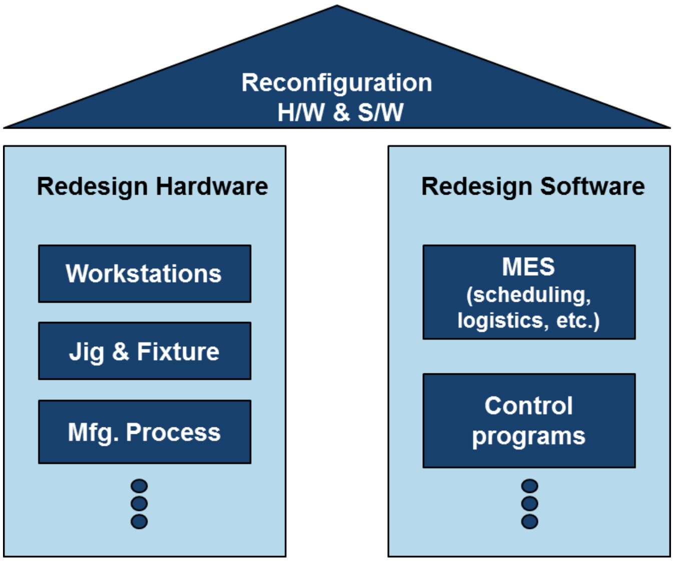

Figure 1.2 Three triggers for new product development

Several studies have focused specifically on optimizing scheduling and logistics. However, few studies have been carried out on specific hardware control and its software; these studies have shed little light on RMS issues. In general, the problem of existing factory installation is that it is time consuming and expensive to implement control software design and verification. This is partly because the actual control software testing is possible only after the hardware has arrived. In order to build lines that produce various products and are free from process change, rapid factory reconfiguration is essential. For this reason, control software must be designed and verified easily and faster.

1.3 Objective

The objective of this thesis is to propose Rapid Control Prototyping (RCP) which is a simulation-based control software development process in a virtual environment. RCP has firstly used in automotive and other industries. It is a process of quickly testing and verifying the control algorithms of the prototype hardware to operate on the test equipment before the development of the control unit for mass production is completed. Therefore, mathematical models in the control algorithms will be imported to the test equipment with I/O information. On the other hands, in the proposed process, the control software will be task sequences and trajectories including position and motion information. The developed control software will be mounted on the existing controller. It can be used by control software engineers to design, test, and verify control software faster. It has two steps: virtual workstation design, and control software development. The advantages of the proposed process are described in detail below.

Using the virtual manufacturing technique, it is possible to model the physical and logical components of manufacturing systems, as well as avoid and verify errors in advance. In addition, control software can be also designed and tested in the virtual environment before they are applied to the real hardware. Consequently, cost and time spent during hardware constructions will be saved.

RCP enables engineer to redesign a fast and flexible process. RCP enables engineers to develop control software faster, including communication input/output (I/O) parameters. It also allows engineers to concentrate on control software designs without the constraints of programming or control languages. To use the RCP, engineers do not need to understand specific code levels because it is possible to automatically generate codes with communication I/O parameters. Interaction between virtual workstations and control software makes it possible to design, test, and verify control software concurrently. Therefore, by simply changing input variables, simulation results can be instantly confirmed in the virtual environment.

1.4 Outline of the thesis

The thesis is divided into five chapters. After this introduction, Chapter 2 surveys research on RMS, Virtual Prototyping (VP) for factories, and Rapid Control Prototyping (RCP) in industry applications. In Chapter 3, the procedure of Rapid Control Prototyping for factory installation is proposed. Comparison of the software development time with and without RCP, and new logic generation and modification for reconfiguration with RCP is discussed in Chapter 4. Finally, the thesis conclusions and future research is described in Chapter 5.

This Page Intentionally Left Blank

II. Literature Survey

The goal of the traditional manufacturing systems was the cost reduction and quality control. Today, manufacturing systems have been developed to adjust various product production and agile to product changes. To challenge these systems, factories should change and reconfigure rapidly. In addition, much scholarly work has been done on the topics of VM technology for rapid factory design to shorten product life-cycle. In industry applications, such as robots, automotive, and plants, numerous studies have attempted to find to rapidly design and test control software, representatively RCP. In this Chapter, researches with respect to reconfigurable workstations in the shop floor level are described with three different perspectives.

2.1 Reconfigurable manufacturing system

Some arguments have been made between Flexible Manufacturing Systems (FMS) and RMS (ElMaraghy, 2005). FMS focuses on variations and built-in flexibility while RMS expects functionality and capacity. In recent years, there have been several accounts that point to realization of RMS than FMS. This is because reconfigurable system is designed for rapid change in structure, as well as in hardware and software components, to quickly adjust production capacity and functionality in response to sudden changes in market (Koren et al., 1999). Many researches have been conducted on RMS with critical issues (Mehrabi et al., 2000)

- **Architecture design**: system components and their interactions like system design
- **Configuration design**: formulations as optimization problems such as planning, scheduling, real-time control, monitoring, and maintenance
- **Control design:** appropriate process variables like system operation

RMS has characteristics of responsiveness, retrofit, and resilience (3R) as mentioned in Chapter 1. Responsiveness means systems capacity is flexible for large fluctuations in product demands, customer preference, and regulations of government. Retrofit allows systems are designed to be ready for both new technologies and production introduction. Resilience implies reliability of severe fault. These characteristics are included in the RMS design issues.

Table 2.1 The summary of the RMS researches with different issues and domains

Authors, Year	Architecture	Configuration	Control	Measurement	Domains
Sims et al., 1997	✓				Enterprise & Factory
Arai et al., 2000	✓				Workstation
Chen, 2001	✓	✓	✓		Workstation & Machine
Landers et al., 2001	✓				Machine
Maeda et al., 2003	✓		✓		Workstation
Zimmermann et al., 2008	✓				Shop floor & Workstation
Naumann et al., 2007	✓				Workstation
Covanich and McFarlane, 2009	✓			✓	Shop floor
Reinhart and Krug, 2012	✓	✓			Machine
Azab et al., 2013	✓				Machine
Otto et al., 2013	✓				Shop floor
Goyal et al., 2013				✓	Machine
Farid, 2013		✓		✓	Shop floor
Hoffman et al., 2014			✓		Workstation
Antzoulatos et al., 2014	✓				Workstation
Bensmaine et al., 2014		✓			Machine
Jatzkowski and Kleinjohann, 2014	✓				Shop floor
Brusaferri et al., 2014	✓	✓			Shop floor & Machine
Zhang et al., 2015			✓		Shop floor
ElMaraghy and ElMaraghy, 2016	✓	✓			Shop floor
Michalos et al., 2016	✓				Shop floor & Machine

Table 2.1 shows researches of the focused issues with different domain. The domains consist of five system levels: machine, workstation, shop floor, factory, and enterprise. RMS has an important issue on architecture designing. From this reason, there have been many researches about that issues over decade. The configuration issues which deal with optimization problems to use system application focused on the targets as shop floors and machines by using industrial standards such as OPC UA (OPC Unified Architecture).

In the control issues, specific hardware control and its software researches have done on the workstation or machine level. The typical RMS issues are RMT (Reconfigurable Machine Tool) and agent-based control architecture.

We can also find some measurement researches on ease of system reconfiguration comparing conventional manufacturing systems with holonic manufacturing systems (Covanich & McFarlane, 2009) and measurement of responsiveness of RMTs: operational capability, machine reconfigurability, responsiveness index (Goyal et al., 2013) with systematic approaches.

2.1.1 Architecture design

Architecture design of RMS is classified into hardware and software system. Reconfigurable assembly systems are presented with a flexible robotic assembly system with decentralized architecture (Maeda et al., 2003). In the same context, much has been said about plug and produce (P&P) architecture that reduces installation time in case of reconfiguration. Figure 2.1 illustrates RMS control requirements.

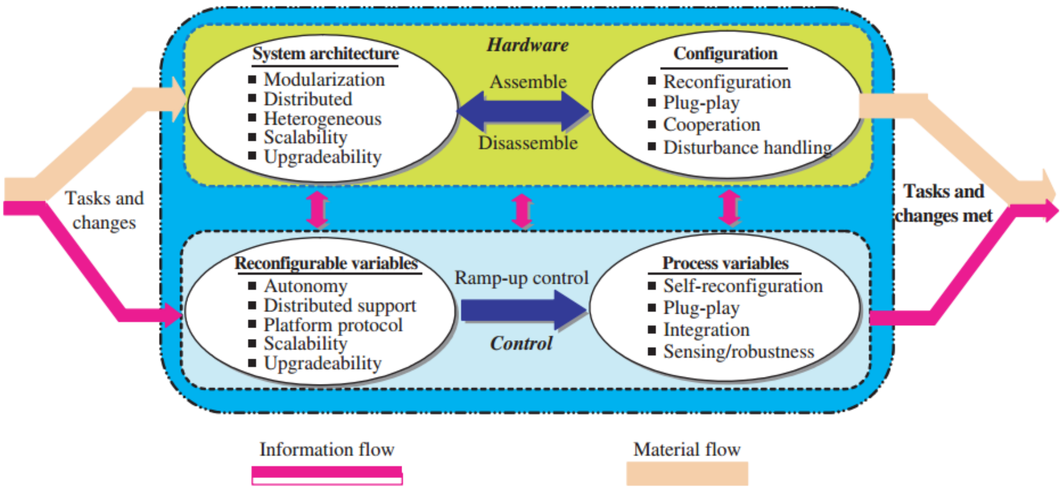

Figure 2.1 Requirements of reconfigurable system control, from Bi et al., 2008

Most of P&P researches normally focus on the architecture. An object-oriented frame-work interoperability for Enterprise Resource Planning (ERP) and Manufacturing Execution System (MES) with plug and play architecture was discussed on the enterprise and factory level (Sims et al., 1997). In the case of the shop floor level, some different levels are introduced for communication and reconfiguration (Jatzkowski & Kleinjohann, 2014; Michalos et al., 2016; Zimmermann et al., 2008). In the same concept of P&P, Holonic Manufacturing Systems (HMS) based multi-agent system for easy and quick reconfiguration without system halt and assembly systems are introduced on the workstation level (Antzoulatos et al., 2014; Arai et al., 2000).

In the machine level architecture, adaptable system structure called RMTs is proposed for the changeable machine and control within part, feature, and cycle time change (Landers et al., 2001). Moreover, two different configuration levels are proposed such as machine level reconfiguration and system level reconfiguration if more major reconstruction is needed (Azab et al., 2013).

The question of communication architecture of P&P is addressed (Otto et al., 2013; Reinhart & Krug, 2012; Reinhart et al., 2010). Whereas, control architectures of P&P are also argued with three layers: application, configuration, communication (Naumann et al., 2007). Especially, interesting from our point of view is the P&P research based on hardware configuration.

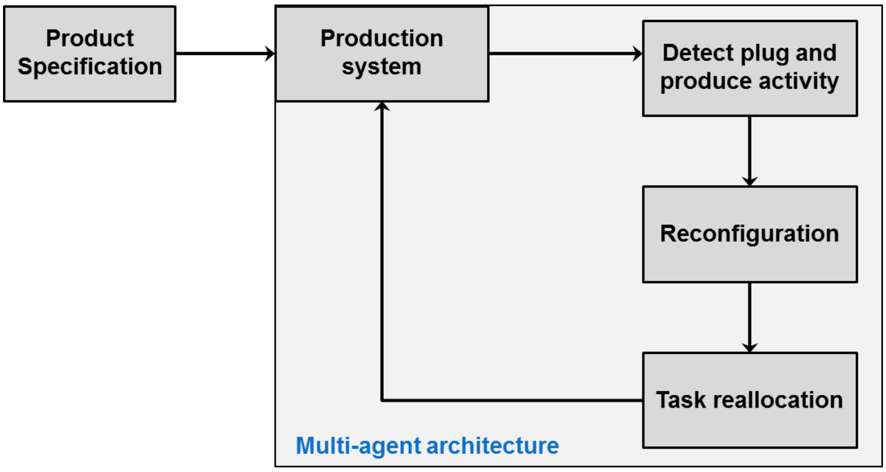

Figure 2.2 Reconfiguration methodology, from Antzoulatos et al., 2014

For the reconfigurable and agile system, system architecture of combining flexible automation and human skill is introduced (Heilala & Voho, 2001). Reconfigurable software also proposed while activating machine level and system level reconfiguration (Azab et al., 2013).

2.1.2 Configuration design

Configuration design of RMS is next process of the architecture design. The methodologies for configuration design find an optimal solution from planning, scheduling, real-time control, monitoring, and maintenance. With regard to the configuration design at higher system level, system simulation which solves time-consuming iterative process is proposed (Adolfsson et al., 2002). Reconfigurable scheduling algorithms are also presented (Steiger et al., 2004; Yi et al., 2006).

In the shop floor level, the configuration of the smart assembly systems is proposed using modular and reconfigurable assembly technology for new trends (ElMaraghy & ElMaraghy, 2016). In addition, virtual avatar based architecture enables effective re-configurability of production systems and OPC UA is used to validate control and communication (Brusaferri et al., 2014). Heuristic problem solving method is suggested using integrated process planning and scheduling like genetic algorithms, simulated annealing, and particle swarm optimization in the machine level (Bensmaine et al., 2014).

2.1.3 Control design

In the control design of RMS issues, reconfigurable robot system is an example. The system should be responded when to meet sudden changes. The control systems in the particular robot are also reconfigured with respect to the quick changes. The concept of control paradigms, such as agent-based technologies are introduced (Hoffman et al., 2014; Shen & Norrie, 1999). The decentralized architecture is proposed for flexible robotic assembly system with high reconfigurability for easy to participate in assembly tasks (Maeda et al., 2003). The new coordinate method is proposed like discrete event control subsystem for reconfiguration and its verification (Zhang et al., 2015). In the machine and robotic level, the concept of the reconfigurable robotic workcell comes from the research of modular robots.

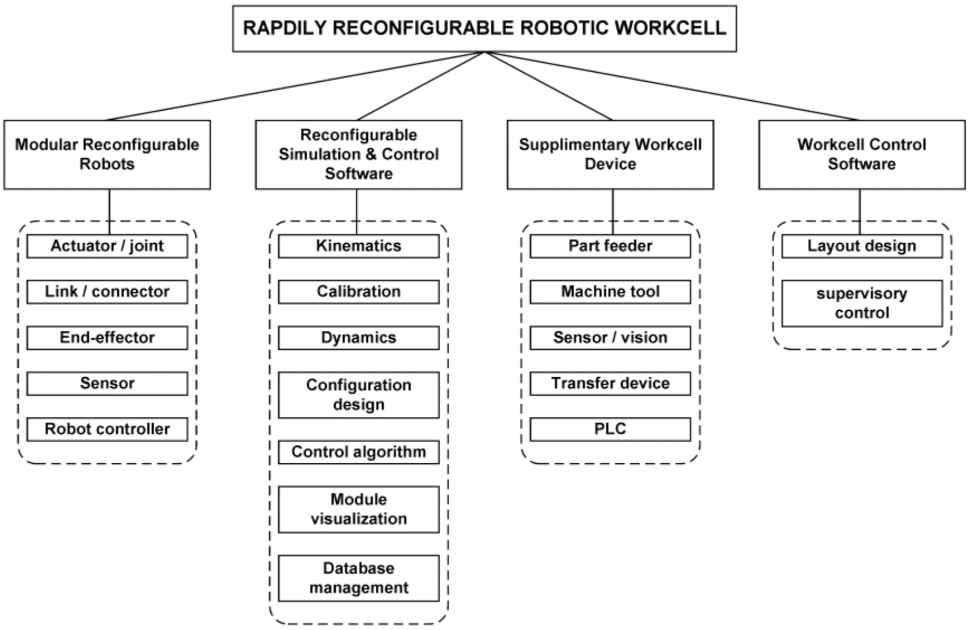

Figure 2.3 Rapidly reconfiguration robotic workcell system, from Chen, 2001

The reconfigurable “plug-and-play” robot control system is one of main researches in the robotic area (Chen, 2001). This research designs hardware component, reconfigurable robot model, robot configuration and optimization. Then robot applied the workcell and simulated using software. In addition, the models are verified through the actual implementation in the robot controller and simulation.

2.2 Virtual manufacturing

This section we will limit ourselves to surveying the virtual manufacturing (VM) scope. The definition of VM is the use of computer models and simulations of manufacturing processes to aid in the design and production of manufactured products. Using VM approach, we can shorten design to manufacturing cycle time, reduce manufacturing and production costs and operation costs. Three paradigms of VM have also been proposed in the report (L.A. Inc., 1994):

- **Design-centered VM:** simulation to optimize the design of product and process
- **Production-centered VM:** simulation capability to manufacturing process model
- **Control-centered VM:** simulation to control models and actual process

The design-centered VM delivers information to the designer during the design phase. The Production-centered VM is a simulation during production to optimize the factory. The control-centered

VM is a simulation that controls machine. Meanwhile, another concept shortening time-to-market is Digital factory which is superclass of the VM for product planning, digital product development, digital manufacturing, sales and support (Kühn, 2006).

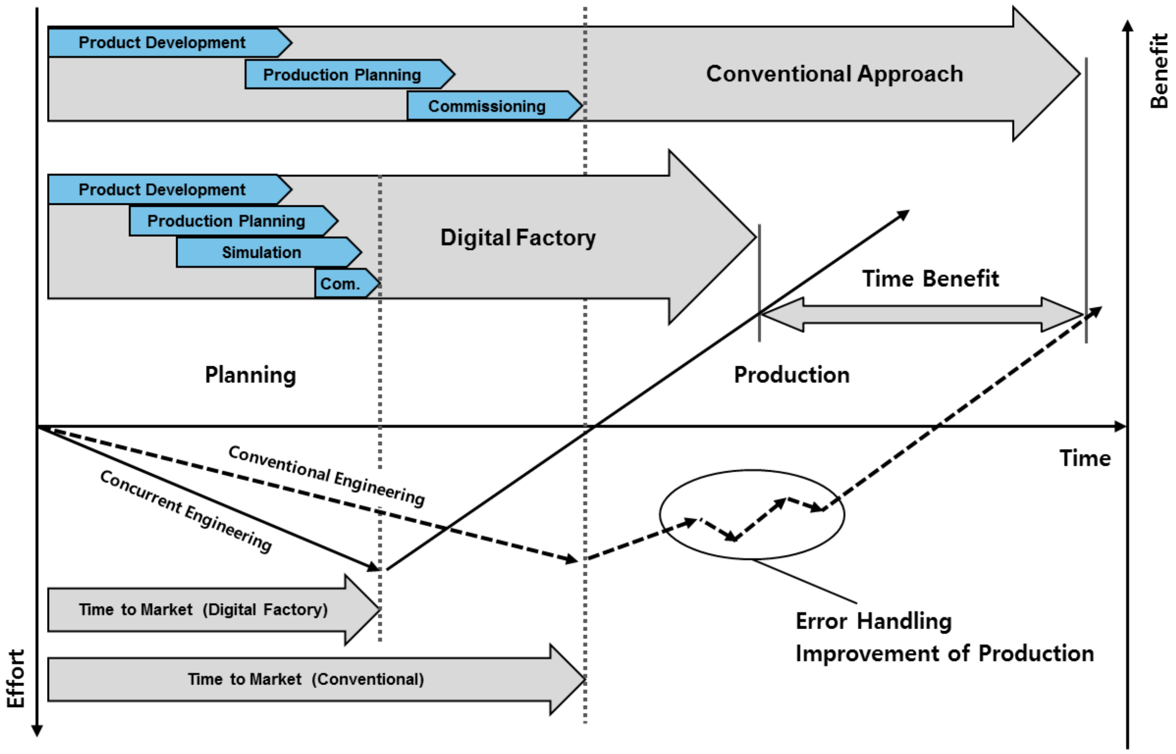

Figure 2.4 Digital Factory - Benefit and Effort, from Kühn, 2006

2.2.1 Design-centered VM

Computer Aided Design (CAD) and Computer Aided Engineering (CAM) software tools are well-known application of Design-centered VM. In the part of CAD/CAM tools, finite element method (FEM) is used for VM (Nayroles et al., 1992). The advantages of FEM are the ability to visualize the distribution of properties and conduct simulation of potentially dangerous, destructive or impractical load conditions. Advantages of virtual design of production system are introduced (Leitão et al., 2009). Virtual Prototyping (VP) techniques are also one of the Design-centered VM (Wang, 2002). This is beneficial to be able to create, test and evaluate virtual prototypes in the production of customized products (Krovi et al., 1999). It is suggested an Integrated Factory Design framework concept that can use different heterogeneous analytical and design tools in the same manufacturing system model in a concurrent and consistent manner (Tolio et al., 2013).

2.2.2 Production-centered VM

Production-centered VM is used for the intent of optimizing the manufacturing processes. It can probably be event-based system. For these reasons, production-centered VM is usually used to validate process and simulation model. The performance of the factory was measured through the development of a common semantic data model representing a virtual factory designed and implemented including both structural and operational aspects of the production system (Kádár et al., 2013). The VM simulation is conducted to evaluate kinematic motions and cycle-time in a sophisticated digital virtual factory for preparation activities in the new process introduced (Park et al., 2005). By adopting digital manufacturing system based on modeling and simulation, it is possible to develop optimized manufacturing line (Choi et al., 2014). Another VM for simulation system is real-time simulation system in the operation planning, scheduling, and control of manufacturing systems (Drake & Smith, 1996). In addition, VP methods and digital manufacturing solutions are now well-suited to play a strategic role in the hybrid reconfigurable system, which combines human resources and machines, design and optimization process (Andrisano et al., 2012).

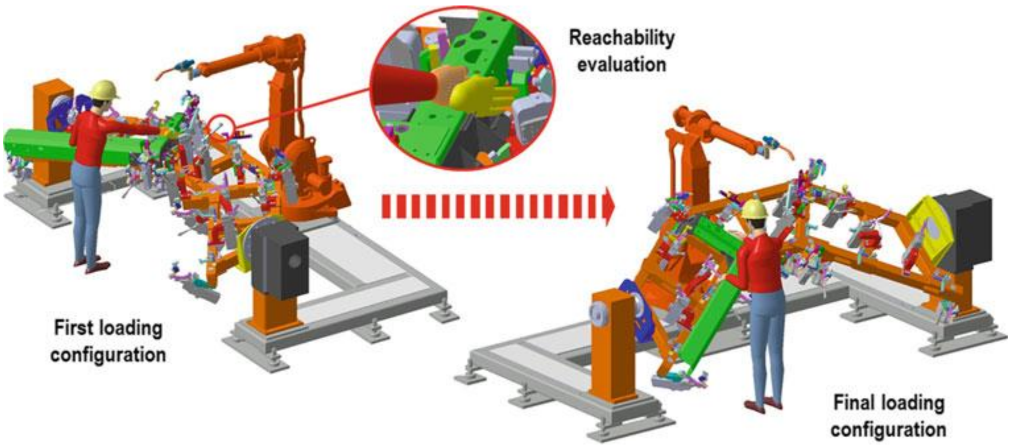

Figure 2.5 Human reachability during the fixturing of chassis subassemblies, from Andrisano et al., 2012

2.2.3 Control-centered VM

Control-centered VM uses machine control models in simulation. To agile manufacturing machinery design and control, the VM integrated approach in order to design, program, test, verify, and deploy control systems is used (Moore et al., 2003). The main goal of a control-centered VM is to enable testing and validation of control software prior to the installation and deployment phases. Under these circumstances, it reduces the overall deployment time and costs by allowing early detection of logical errors or problems related to process design and configuration (Mourtzis et al., 2015).

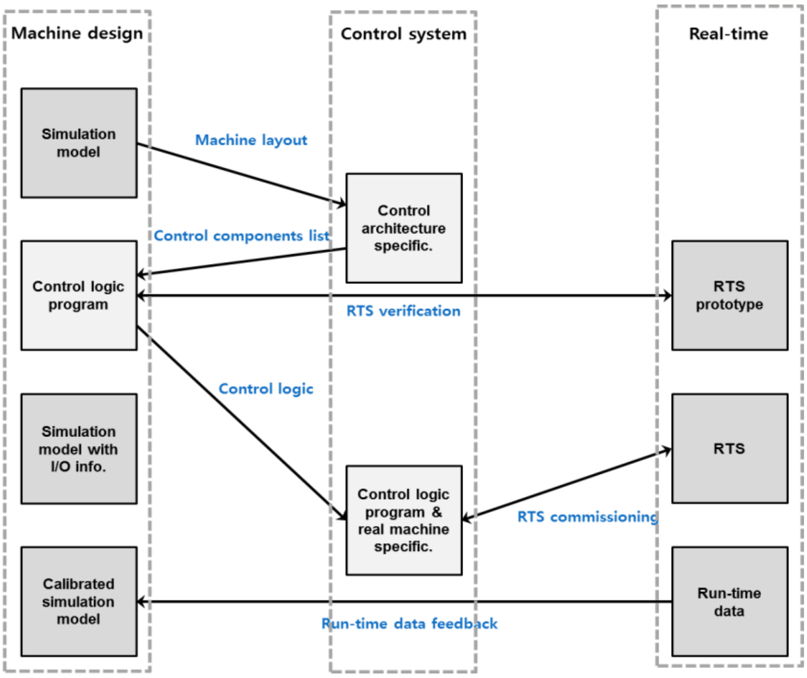

Figure 2.6 Key elements and processes within the integrated approach, adopted from Moore et al., 2003

This approach allows real-time data collection during operation to calibrate the simulation models with different environments like machine design, control system, and real-time. It is proposed that control architecture, which has decentralized layer, is to perform the harmonization and the cooperation between the cell components (Kim & Choi, 2000). The importance of integrating product realization domains is emphasized on the task sequence and control logic that make changes easier (Ahmad et al., 2016).

2.3 Rapid control prototyping in industry applications

RCP is development environment for control system engineer to design and test efficiently and quickly. The designer can focus on control design rather than programming details or debugging control languages (Rubaai et al., 2008). It is widely used to develop complex control software such as Engine Control Unit (ECU) and test the process performance (Kimura & Maeda, 1996; Lee & Park, 2006). The energy system such as smart grid also introduce the RCP concept to control a large amout of distributed generators (Faschang et al., 2013). In the manufacturing industry, robot control system implements with RCP (Chen et al., 2004; Lapusan et al., 2008).

RCP in many industries is used for control algorithm (or simulation model) design and test, and actual drive and controller modeling. Many researches present RCP by using block oriented development tools such as Matlab Simulink as software and dSPACE as hardware for the ease of implementation.

Most of the purposes of RCP are fast and simple control algorithm design, test, and verification. Sometimes it is used for preliminary verification of dangerous targets such as batteries (Subramanian et al., 2012). The RCP process also has various difficulties. It can be used as a simple tool for research and student education, and on the other hand, for the development of real drives and controllers. Table 2.2 shows the summary of RCP in different code levels and targets. RCP for end users is easy to develop and test control algorithms and it has a simple development process such as system design, HILS modeling and test, and implementation of real hardware. For the developments of real drivers or controllers, it will be complex to code of real-time low level controls.

Table 2.2 The summary of RCP of different code levels and targets

Code level	Target	Authors, Year	Goal	Discription
Control algorithm design and test for end users	Battery Mangement System (BMS) for hybrid/electric vehicles (HEVs)	Subramanian et al., 2012	Develop, calibrate and verify BMS algorithms in a safe and time-efficient manner	Programming, testing, and verification of BMS are time consuming and dangerous. It is used for monitoring cell pack voltage, current, temperature, charge status, discharge status, fault status. System design → HILS model development → Simulation
	Automotives for mechanic, hydraulic, electirc control	Menager et al., 2014	Reduce commissioning time and re-implementation of new industrial systems	It is userd for Bosch Rexroth's mechanic, hydraulic, electric control toolchain development (of a source code converter) and utilizing the HILS preparation phase of the RCP process. User-friendliness, ease of control algorithm development and pre-testing in the simulation environment, avoiding re-impementation (S/W redesign and H/W configuration), open source S/W Set up a simulation model → mounting controller → simulation running with real H/W (HILS)
	Helicopter, Automotive electronic throttle.	Grepl, 2011	Educational approach to people with no prior knowledge of real-time control	It introduces the embedded hardware tool that acts as a bridge between actual hardware and software. Real-time control, easy to use, focusd on data logging and block visualisation Control algorithm development → C-code generation → I/Os tranmission to control H/W → real plant implementation
Controller or drive modeling (designing) and test	ECU for HEVs	Nagaraj and Detrick, 2009	Minimize the time required to develop a control strategy	It acts as a conformation of HILS to test control algorithm development. Control algorithms can be quickly modified and tested through iterative methods. Easily changing control parameters in operating status. Software modeling and automatic code generation, rapid evaluation of complex control strategies HEV model definition → motor simulation → parameter feedback → sensor error simulation → controller test with RCP → HILS
	Automotive electronic throttle	Grepl and Lee, 2010	Model, estimate parameters, and design controller	It is based on controller design of a throttle servo system with non-real time simulation using RCP hardware. System property identification → plant modeling and parameter estimation → optimized plant model definition on HILS → non-linear model definition → experiment
	Motor drive for speed control	Tursini et al., 2013	Develope electronic drive digital controller efficiently and rapidly	It is used for easy and quick evaluation and testing of complex or non-standardized control solutions and rapid development steps for new products through saving time and money. Offline simulation (control schema and parameter definition) → code generation (real-time control and I/O settings) → test and optimisation
Model test and calibration	Mobile robot	Rossmann et al., 2012	Simulation based control design and concurrent engineering.	It conducts RCP for mobile robot for 3D simulation model with motion, self-localization robot and gap reduction between H/W and S/W through virtual and physical sensors. Compressive simulation with Virtual Testbed, easy calibration with virtual and physical environments, effective for motion control and path planning, prediction of the physical sequence of robot motion Real-time Virtual Simulation Database (VSD) construction → control algorithm design, prototyping, test, and verification in virtual environment → H/W implementiation

2.4 Factory planning

The dilemma of factory planning today is not only the design of production systems that last for decades, but also the requirements of dynamic market changes (Schuh et al., 2011). Factory planning must be transparent so that the impact on the production plan can be traced to enable the flexibility needed in production in relation to short-term changes (Büscher et al., 2014).

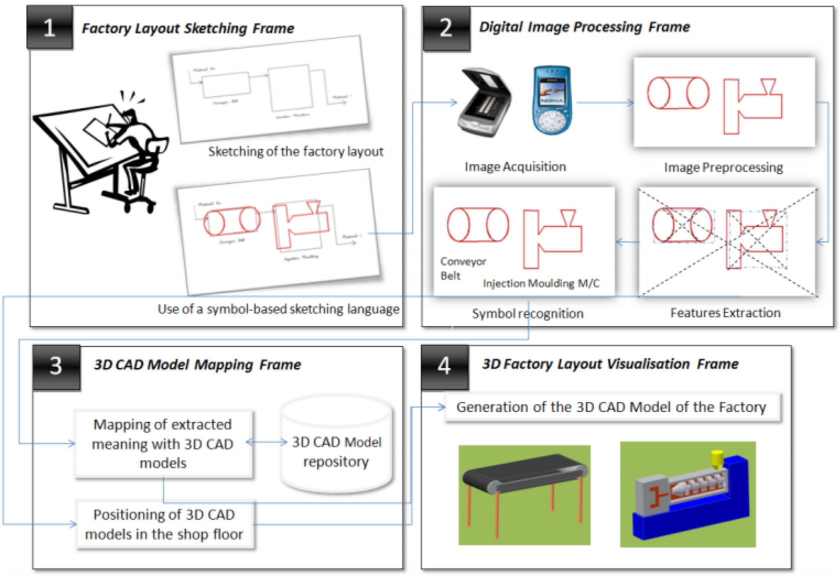

Figure 2.7 Layout design of a sketch-based framework, from Farrugia et al., 2010

In the layout design step, it was developed as a sketch-based framework that allows users to quickly get factory 3D CAD models directly from the factory's paper-based sketches (Farrugia et al., 2010). In the configuration step, a wireless communication interface as smart devices can be used for logistic. Here, Radio-Frequency Identification (RFID) technology is representative (Zuehlke, 2010).

We must correct the operators that are unfamiliar with the new procedures during various errors, control software errors and stabilization time caused by the machine not being properly adjusted by using Hardware-in-the-loop simulation (HILS) (Park & Chang, 2012). For manufacturing execution, MES can be set up and its functions will be customized for users. Software has been developed to support requirements related to real-time, cloud-based, and lean operations as Figure 2.8 (Helo et al., 2014). Calibration should be done in the real hardware installation.

The typical calibration method is using camera (Zhang, 2000). a new and fast calibration method based on Quick Response codes (QR codes) is also proposed (Andersen et al., 2013). The geometric calibration of industrial robots is also conducted. It focuses on reducing effects of measurement noise by appropriately selecting the manipulator configuration in calibration experiments (Wu et al., 2015).

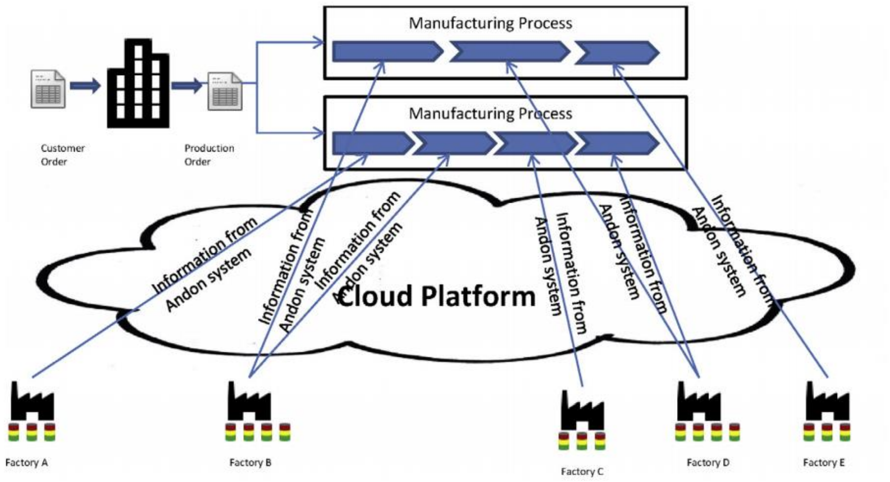

Figure 2.8 Integrating ERP systems with MES systems, from Helo et al., 2014

2.5 Summary of literature

Before we end this section, major findings from literature will be presented. Both RMS and VM have great potential to improve the current process development system. To be sure, several studies have focused on specifically optimizing scheduling and logistics; however, few studies have been carried out on specific hardware control and its software.

In RMS, there has been many studies on machine-level reconfiguration systems, such as RMTs. On the other hands, relatively little research has been carried out on a workstation-level reconfiguration system that requires machine to machine synchronization and scheduling. No less significant is the fact that reconfigurable control systems for the workstation level when lines need to be changed for responsiveness of customers.

In the case of VM, many researchers investigated production-centered VM that deals with scheduling, logistics, material flow, etc. However, many studies have not been conducted in the field of control-centered VM. Control-centered VM cannot produce prototype control software quickly because of challenges, such as complex programming languages. Moreover, it is difficult to simulate reconfigured manufacturing processes in a production-centered VM owing to the lack of a control model and the large amount of control components. Therefore, a rapid control software development process between control-centered VM and production-centered VM is required.

It is necessary to obtain the integration tool that is able to design, test, and simulate the control software easily and quickly, just as RCP has been used in other industries. Table 2.2 shows a comparison of the use of RCP in other industries and in the factory. In short, RCP is mostly used for the step before the HILS in other industry application, while it is used for the rapid control software development independently in the factory.

Table 2.3 Comparison of RCP for other industries and factory

	RCP for other industries	RCP for factory
Target	Automotives (ECU, BMS, etc.), aerospace, robots	Workstations or machines
Goal	• Verifying the control functions against real world signals (Subramanian et al., 2015) • Rapidly testing and iterating control algorithms for the better performance (Mauch et al., 2014)	• Engineers can design, test, and verify control software rapidly in the virtual environment • Using VM technique, it is possible to model the physical and logical components of manufacturing systems, avoid errors in advance
Advantages	• The modeling of software allows the algorithm to be repeatedly and quickly changed (Nagaraj & Detrick, 2009) • It is possible to implement and validate control strategies during the developing process (Bucher & Balemi, 2006)	• Cost and time taken during the H/W construction will be saved • Engineers can concentrate control programs without constraints of the programming or control languages
Procedure	Control software → RCP hardware or HILS → Real ECU or plant	Virtual workstation → Control software → Real workstation
Features	• Real-time control and verification with sensors • Focus on connection between RCP and HILS (HILS for RCP)	• Rapid design, test, and verification of control software in the virtual environment for factory reconfiguration • Robot offline programming (OLP) development procedure without commercial software or standard libraries

There is also a significant difference among procedures. In the case of RCP for other industries, control software for a simulation model is first designed and tested on a simulation hardware such as HILS. Finally, it is implemented on a real plant or ECU. However, in the factory case, a virtual workstation is first designed, then the control software is developed in the RCP. Finally, the control software will be applied on a real workstation.

Most RCP procedures for automotive industry have three levels of development: design level, test level, and application level. In the design level, the simulation model is developed with system design; thereafter, control codes are generated. In the test level, the developed simulation model is tested on RCP or HILS hardware. It is possible to modify the control algorithms and input values of hardware with sensor data in real-time simulation. Finally, in the application level, the developed model is applied in a real plant.

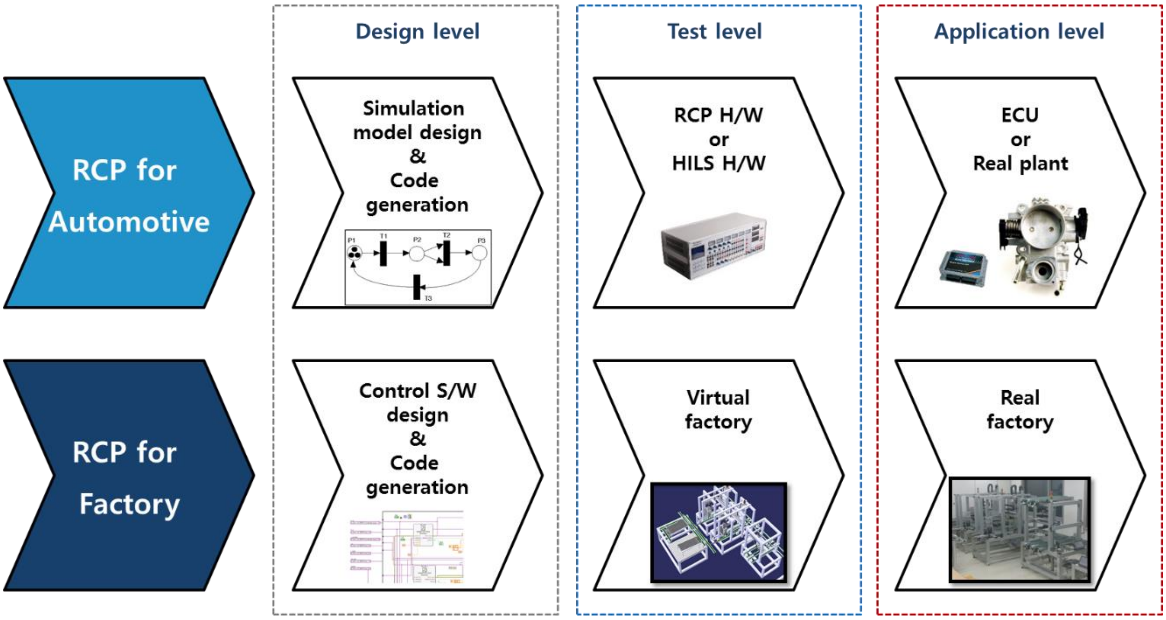

Figure 2.9 Procedures of RCP with three levels in the different industries

In the RCP procedure for manufacturing industry, the test level is the most important. The developed control software is tested mainly in a virtual factory unlike in the automotive industry where it is tested in hardware simulators. Therefore, by using this one step that is different while retaining the same concept of RCP used in many industrial applications, the RCP can be introduced in the manufacturing industry.

This Page Intentionally Left Blank

III. Rapid Control Prototyping for Factory Installation

3.1 The rapid factory installation procedure

Factory design process should be conducted to shorten design time and avoid planning errors in advance. Because of time and cost considerations, a methodical approach is proposed. The factory installation procedure can be divided into hardware configuration and software test and installation. There are six steps for the rapid factory installation as shown in Figure 3.1: layout design, controller configuration, 3D factory modeling and control software design, factory-in-the-loop simulation, factory operation systems installation, and test and calibration

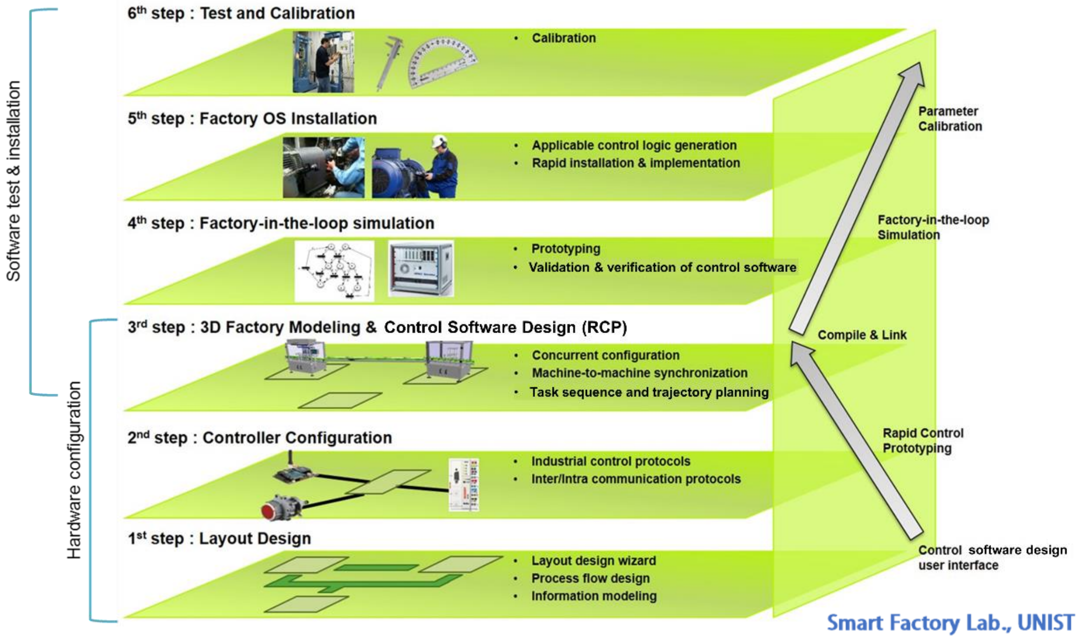

Figure 3.1 6 steps for rapid factory installation

3.1.1 Layout design

To optimally place functional resources in the factory, factory planners should consider constraints such as space limitations. Layout planning plays an important role in factory planning because it needs to integrate the results of previous plans. There are many new layout planning technologies today. There are layout assessment methods using the VPI (Virtual Production Intelligence) platform are provided to enable value-oriented layout planning (Kampker et al., 2013).

3.1.2 Controller configuration

A changeable and flexible factory systems require the inevitable adaptation of field device functions to changing production conditions (Schmitt et al., 2014). As the same concept, control and communication protocols which follow industrial standards will be featured to communicate dissimilar field devices. OPC UA that integrates smallest devices in the internet of things is intended to allow application programmers to view network services vertically and consistently (Imtiaz & Jasperneite, 2013). For controller configuration, resources and drivers with corresponding control channel should be specified. Additionally, it is necessary to develop inter and intra communication architectures to ease and safety of data gathering.

3.1.3 3D factory modeling and control software design (RCP)

For implementation of manufacturing processes in the virtual environment, 3D virtual factory should be designed. After building up the 3D model, engineers can quickly develop control software, test them in a virtual environment prior to constructing a real shop floor.

3.1.4 Factory-in-the-loop simulation

Factory-in-the-loop simulation is used for verifying the stability, operation, and fault tolerance. To save development time of factory installation, most tests can be completed before a factory prototyped. Moreover, the developed complex control software will be validated and verified to enhance the quality of testing by iterative simulation in a hardware environment.

3.1.5 Factory OS installation

For the outstanding performance with the systematic waste elimination, we need factory operating systems. By installing factory operation systems, it is possible to generate applicable control codes. We can install the control software and communication drivers for network settings.

3.1.6 Test and calibration

To verify and validate the control software developed in the virtual environment, they will be embedded in the controllers. While adjusting input variables associated with actuators, calibration will be completed in the real environment. This measurement procedure is difficult and time-consuming. This step deals with the geometric calibration of industrial robots and workstations reducing measurement noise effects through calibration experiments.

The procedure follows V-model of the systems development from the control software design user interface to the parameter calibration. The control software must be tested by the parameter calibration. In addition, RCP process must be validated by factory-in-the-loop simulation. This paper particularly aims to 3D factory modeling and control software design (RCP) in the step 3 and test and calibration in the step 6 of the rapid factory installation procedure. Other steps are merely beyond the scope of the present study.

3.2 RCP for factory installation

3.2.1 Definition of RCP

RCP is a simulation-based control software development process in a virtual environment through which control software engineers can design, test, and verify a task sequence and trajectory rapidly. It enables engineers to quickly develop them including communication I/Os, test them in a virtual environment prior to constructing a real shop floor. The developed control software will then be installed and calibrated with real workstations and devices in a shop floor.

As distinguished from Off-line Programming (OLP) which is a robot programming method through a graphical 3D model in a simulator, RCP does not need any costly commercial software or standard libraries. This is because RCP is a process that develops their own robotic libraries. For these reasons, RCP can build up and test libraries for customized robots and simple actuators without their libraries. a RCP also allows engineers to concentrate control software design without constraints of the programming or control languages. They do not need to understand specific code levels because it is possible to automatically generate codes with communication I/Os.

RCP consists of three platforms: virtual workstation, control software, and real hardware described in Figure 3.2. In the virtual workstation, 3D factory is required after assembly procedure is completed. To make a simulate model, motions including controllable units, motion types, coordinate mode, kinematic modeling, and motion constraints should be defined. After that, task sequence based on assembly procedure is listed with relevant units with the task and corresponding positions. To control real hardware, input and output information will be mapped with each port. Finally, control software will be developed. These issues will be described in detail on Section 3.3

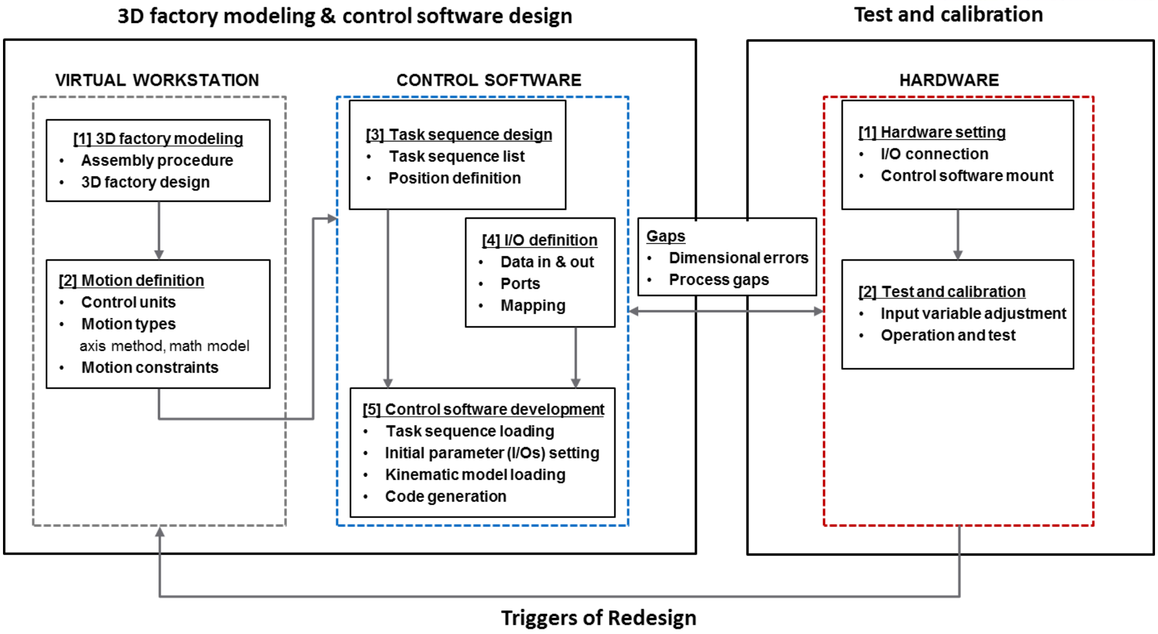

Figure 3.2 RCP architecture

In the test and calibration phase, the real hardware will be tested and adjusted with virtual workstations. After hardware settings, the controllable parameters will be adjusted iteratively within real and virtual workstations.

3.2.2 Advantages of RCP

RCP can reduce the risk of physical hardware damage during testing based on VM technology. The problem of various kinds of malfunction, interference and collision between hardware and software can be tested and found in a virtual environment in advance, so that the control process can be performed more stably when the control software is applied to the actual hardware.

RCP is an easy and fast development process for end users. Task sequences can be rapidly developed, and various I/O can be defined through intuitive block diagram based programming, so that the it can be recognized in the code, and the hardware can be configured based on its information. In addition, mathematical models can be developed in the form of a library to define control motions close to reality.

RCP is a concurrent development process of hardware and software that shortens the overall process development time. In addition, not only the process development time according to the new workstation design but also the software development time due to the hardware reconfiguration can be shortened. Through a simple modification of the control program already developed, the redesign process has a faster development process.

For factory installation, it takes months or years from virtual factory construction, to hardware test, and to eventually software development. To verify rapid and effective control software design and test, we compared the software development time with RCP and the traditional way without RCP in different cases.

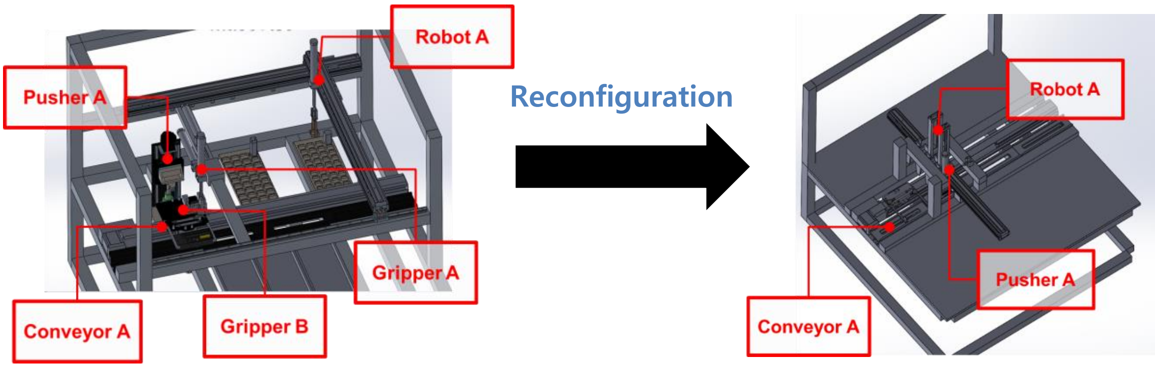

Figure 3.3 New design for vertical assembly workstation and redesign for battery assembly workstation

CASE 1: New design of workstations for a new factory

In general, there will be some steps like workstation design, hardware assembly, software development, and hardware test, when new workstations are introduced. Figure 3.4 shows the time which is required to develop the software with RCP according to the steps.

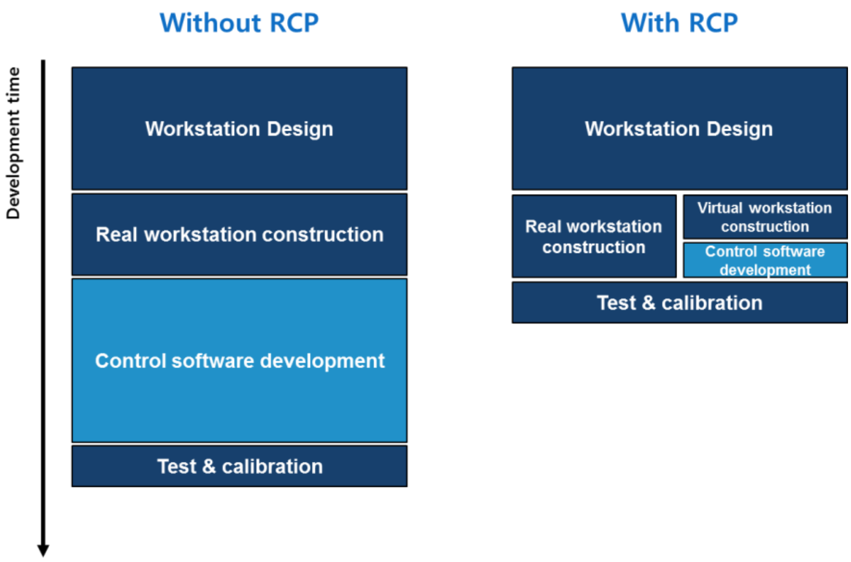

Figure 3.4 Software development time for new design

Both processes with RCP and without RCP spend same time from workstation design, real workstation construction, and test and calibration. Conventionally, control software development is time-consuming. This is usually delaying task until real machine is constructed. Additionally, test and calibration can be progressed immediately after the software development. Therefore, this serial process increases the overall development time for workstations.

Comparing with the development time without RCP, the process with RCP has an additional process like virtual workstation construction. It is for rapid development and testing of control software through virtual environment. This virtual workstation construction allows control engineers to predict how control software works and rectify design errors in advance in real-time without damaging the real hardware equipment.

Control software design will be into action right after virtual workstation design with RCP while the other needs real hardware. After control software design, hardware test and calibration can be done with hardware development concurrently using completed simulation models. Unlike parallel process of using RCP, traditional process works serially on each step. Therefore, total software development time using RCP saves a lot of time through the simultaneous process.

CASE 2: Redesign of workstations for reconfiguration

The case 2 is the assembly workstation is changing from the vertical assembly workstation to battery assembly workstation. The new workstation produces totally different products compared to original products so that the task sequence and input variables should be reconfigured. The tasks have been decreased compared with the previous workstation. However, control software including task sequence and input variables will be entirely modified in response to changed tasks. In terms of hardware, Conveyor A has the same works compared to the vertical workstation. Robot A and Pusher A will be newly designed now.

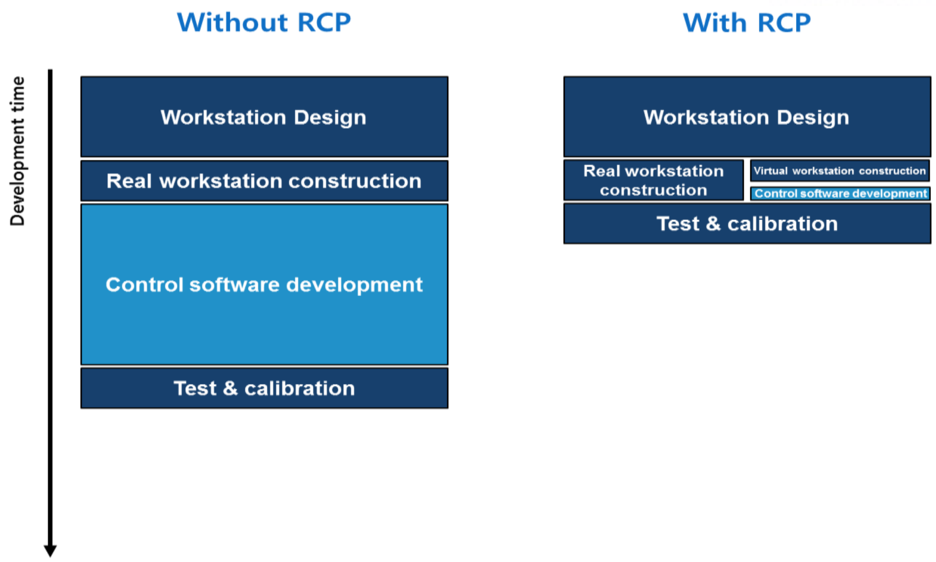

Figure 3.5 Software development time for redesign

In this case 2, we measured software development time for the redesign process with RCP against without RCP as shown Figure 3.5. Most steps will be shorten compared to new designing process of the case 1. The first step is workstation design. We need to redesign the battery magazine module and pushers. Second, they will be built the appropriate position in the workstation. Some parts will be required on machining process. In the third step, there are some different time distributions with RCP and without RCP. The step of control software development is still time-consuming in the process without RCP. That is because control software should be newly developed and verified when the workstation is redesigned.

However, it takes a few days to redesign control software because we can simply add the new actuators and modify the positions. After redesigning control software, we can start verification of the new control software with simulation. Both the third and fourth step can be processed during the second step. From the gap, total software development time will be more reduced in redesign process than in the new design process.

3.3 The RCP procedure for reconfigurable assembly workstations

The RCP procedure is implemented on one of the reconfigurable assembly lines, called a vertical assembly workstation. The assembly lines are applied RMS that enable to produce various products like secondary batteries, electric toothbrushes, and cordless endodontic treatment handpieces.

The reconfigurable assembly line for smart factory consists of 8 workstations: automatic part loading, battery assembly, branching, vertical assembly, assembly robot, screwing, packaging and inspection, and unloading. This assembly line is possible for customer needs to change the layout with reconfiguration and P&P.

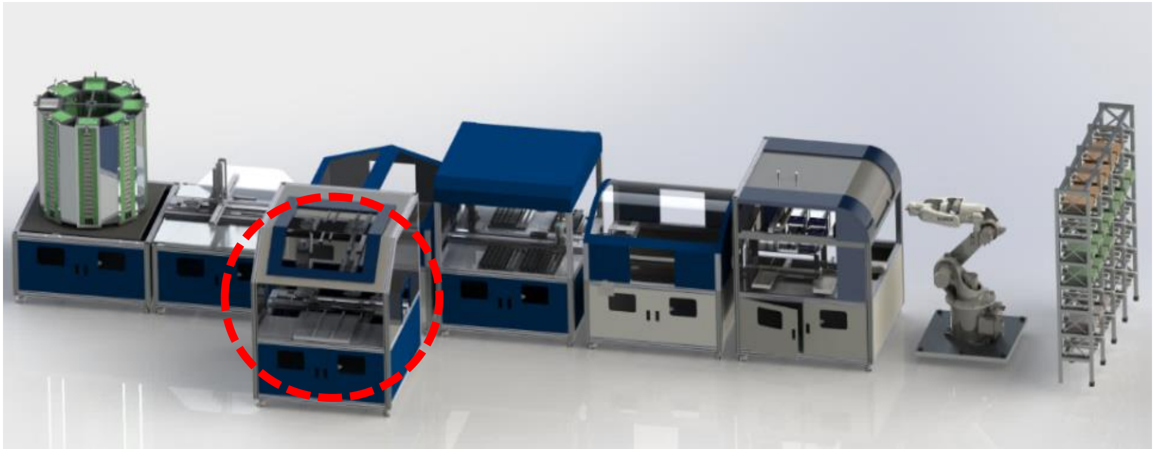

Figure 3.6 Reconfigurable assembly line for smart factory

In this paper, we focused on the vertical assembly workstation that produces different kinds of secondary batteries shown as Figure 3.7. There are four different products produced in the vertical assembly workstation. The secondary batteries are composed of two combinations like shape and capacity. There are two kinds of shape such as O-shaped and square-shaped and capacity such as 14500mAh and 18650mAh.

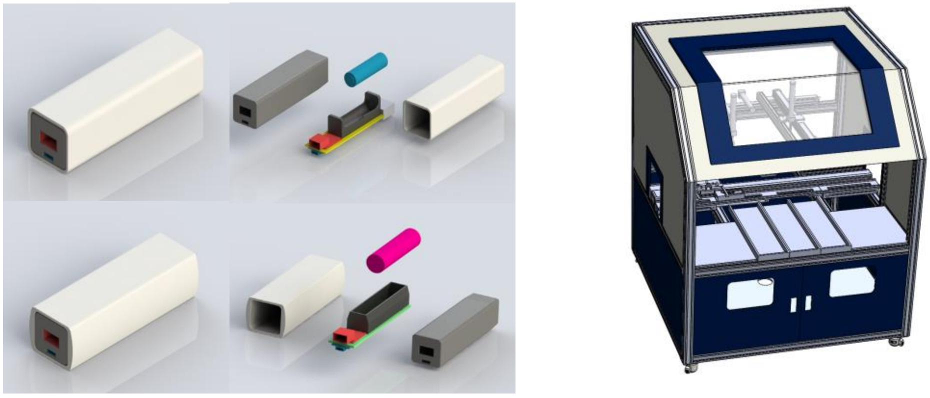

Figure 3.7 Target products (left) and workstation (right)

3.3.1 3D factory modeling and control software design

RMS aims to reconfigure hardware and control resources in the shop floor, in order to quickly respond various customer needs. To meet the customer needs, control software should be changed easily and rapidly. RCP is a simple control software development process that uses pre-defined task sequences, trajectories, motions, and control I/Os. It does not need the specific code levels or low level controls like control loop feedback mechanism or signal controls.

To develop appropriate control software, kinematic models must be designed and installed in virtual workstations. Control software will then be developed and tested with virtual workstations. Figure 3.8 illustrates the key process work packages across the virtual workstation and control software development. After finishing virtual workstation design, developed motion models and I/O information (input and output variables) will be sent to control software. To simulate the virtual model, control software will then drive the simulation model that includes a test sequence and kinematic models. Finally, control software will report the simulation results.

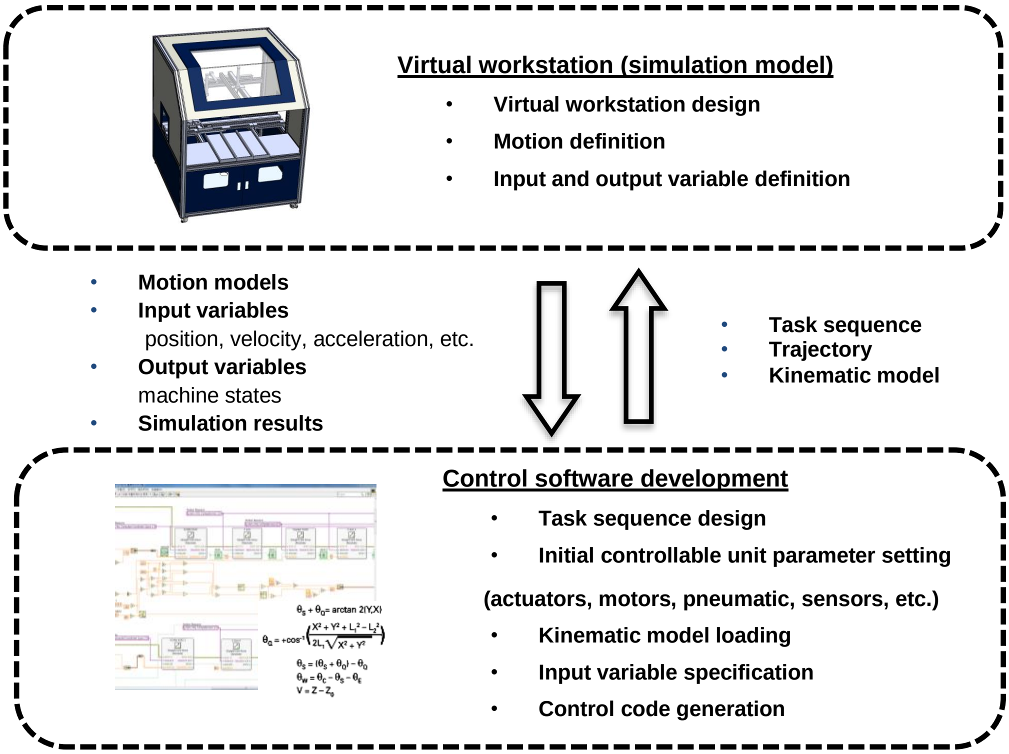

Figure 3.8 RCP procedure for the factory installation

There are five steps for 3D factory modeling and control software design: 3D factory modeling, motion definition, task sequence design, I/O definition, and control software development. The step 1 and 2 are done for virtual workstation design while the others are done for control software development.

Step 1: 3D factory modeling

After the workstation design stage, virtual workstations should be firstly designed to create simulation models. It is possible to draw based on 2D drawings developed in the workstation design stage. Figure 3.9 describes 2D drawing for vertical assembly workstation and assembly units.

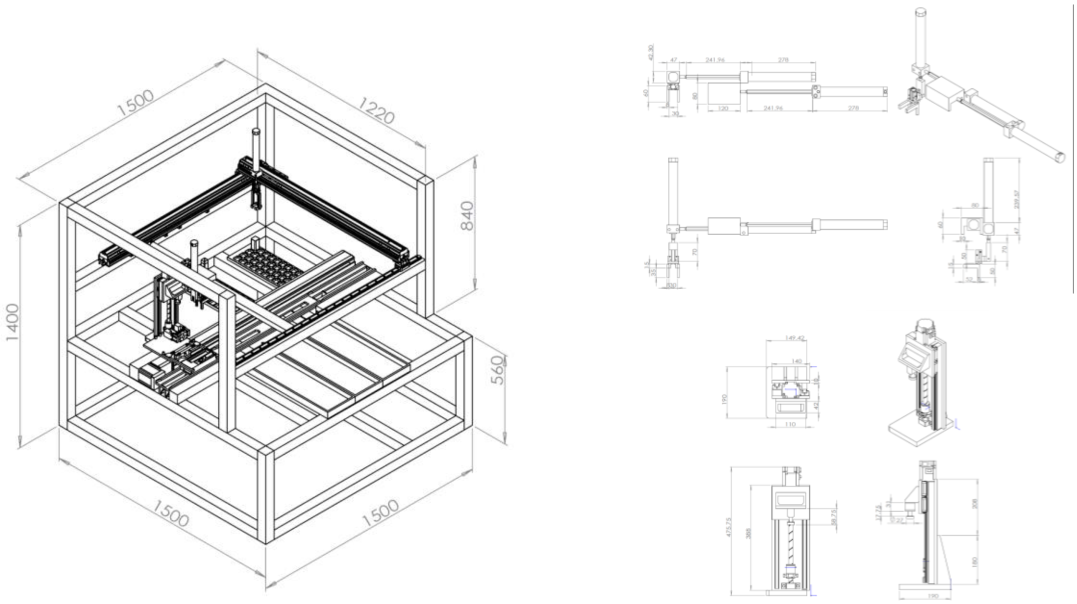

Figure 3.9 2D drawing for vertical assembly workstation

In the vertical assembly workstation, we designed 1500x1500x1700mm size of a cube shape work cell. The workstation is composed of a linear actuator type conveyor, assembly pusher, 2.5 axis gantry, and pneumatic grippers. We used 800mm and 1250mm stroke linear actuators, 200mm and 500mm pneumatic pushers, and 100mm stroke pneumatic grippers in the workstation. Figure 3.10 is 3D virtual model for the vertical assembly workstation and Table 3.1 is hardware specifications of the workstation.

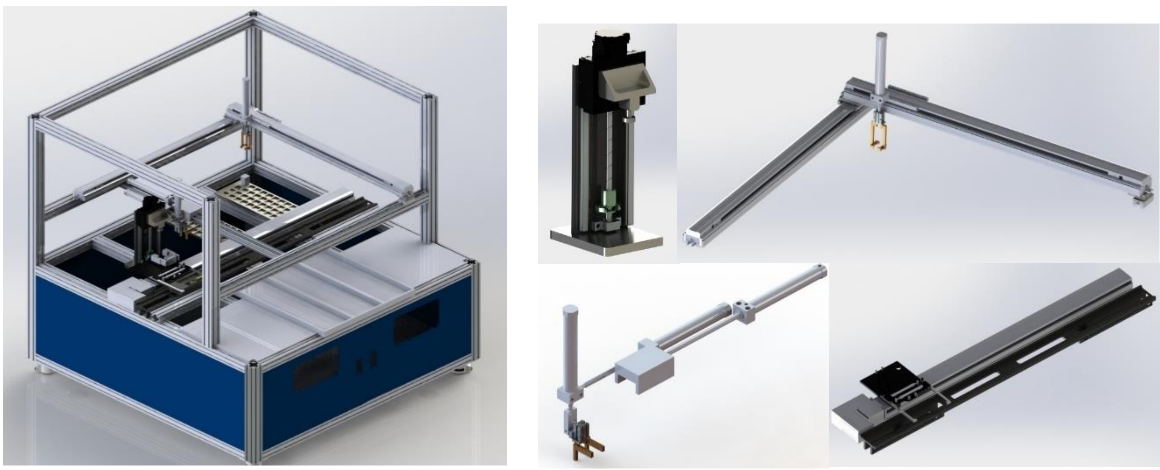

Figure 3.10 3D virtual model for vertical workstation

Table 3.1 Hardware specifications of the vertical assembly workstation

2.5 axis gantry robot (Robot A)	
x-axis stroke	800mm
y-axis stroke	800mm
z-axis stroke	200mm
Max payload	80kg
Max speed	300mm/sec
Weight	10kg
Linear actuator type conveyor (Conveyor A)	
Stroke	1100mm
Max payload	110kg
Max speed	300mm/sec
Weight	15kg
Assembly pusher (Pusher A)	
Stroke	200mm
Max payload	5kg
Max speed	20mm/sec
Weight	4kg
Product gripper (Gripper A)	
Stroke	100mm
Max payload	5kg
Holding gripper (Gripper B)	
Stroke	100mm
Max payload	5kg

To simply the model for simulation, we eliminate complex shapes like many curves that make simulation slow and superfluous parts that are unnecessary for simulation. We have reduced the number of parts by half from 115 to 68.

Step 2: Motion definition

Virtual workstation is divided into fixed parts and movable parts. The control target units involved assembly process among movable parts will be selected as the controllable units. They must be assigned constraints including alignment, orientation of parts and surface coincidence (Vermaak & Niemann, 2015). Figure 3.11 illustrates motion constraints of the controllable units.

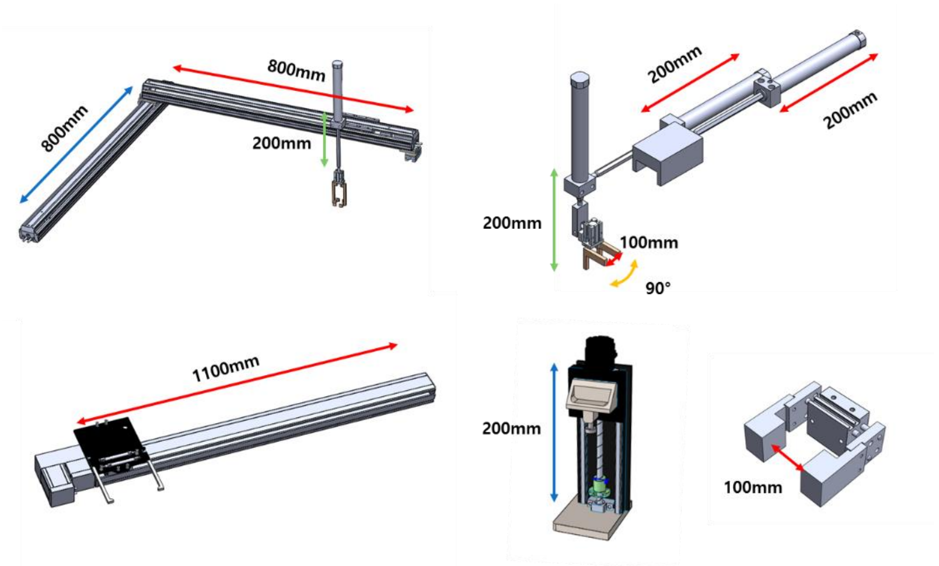

Figure 3.11 Motion constraints of the controllable units

After allocation of the constraints, motion type should be defined. There are two types of motion: linear motion and rotational motion. Linear motion is straight line moving from one point to another. Rotational motion is rotating about an axis. In the case of the vertical assembly workstation, the 2.5 axis gantry type robot has linear motion. The gripper that picks up the PCB modules and final assembly parts has a servo motor to rotate parts from horizontal state to vertical state shown as Figure 3.12.

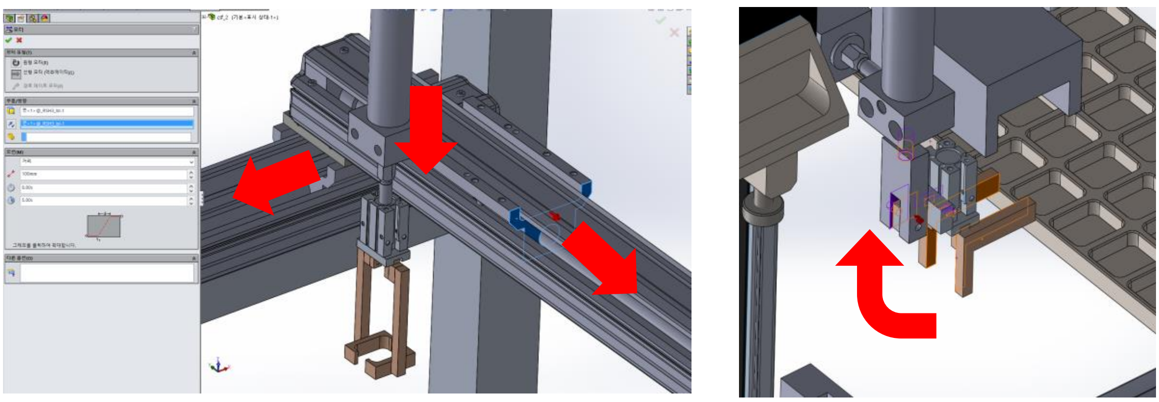

Figure 3.12 Motion type definition

After allocating the constraints, coordinate modes will be defined. Absolute mode is parts move from the program zero or origin whereas relative mode moves from the current position. They can be selected according to the controller and control method, we used absolute move in the case of this workstation.

A kinematic model is required for stability of movement and designation of the correct trajectory. Adjusting velocity and acceleration gives stability to part transfer and assembly and applying inverse kinematics changes the joint parameter values to the desired position values of the end effector.

Step 3: Task sequence design

In order to design control software easily, the task sequence will be firstly designed to process tasks. The task sequence refers to assembly procedures in the workstation design stage in the Figure 3.13. In this process, trajectory should be considered and verified to avoid collision among the machines in the virtual workstation.

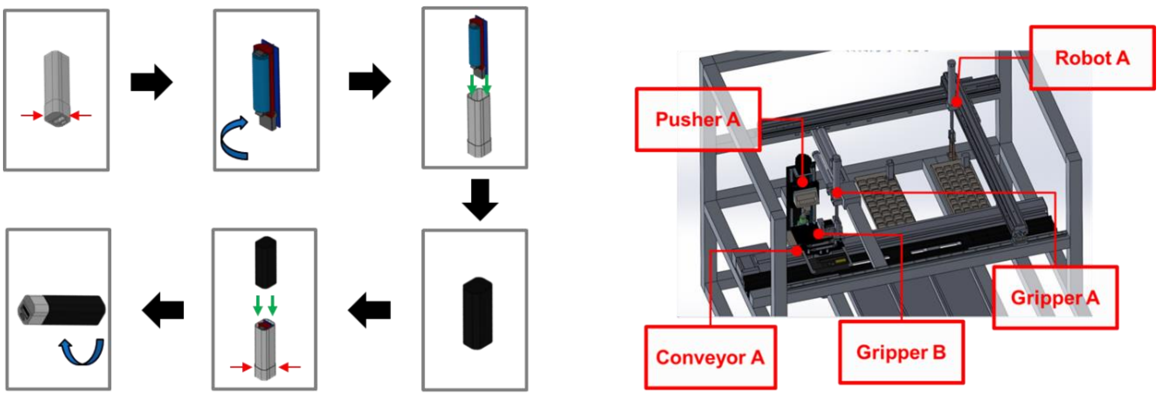

Figure 3.13 Assembly process by the vertical assembly workstation (left) and major controllable units (right)

The specific task sequence of the assembly workstation is listed on Table 3.2. Originally, the vertical assembly workstation has 45 tasks but this table is a main task sequence with the relevant units and corresponding positions. These data will be mapped to I/O information in the next step.

Table 3.2 Task sequence for the vertical assembly workstation

Seq.	Task	Relevant units	Positions
1	Conveyor A moves to CON_POS1	Conveyor A	CON_POS1 (350)
2	Robot A picks up the case from the case pallet	Robot A	RB1_POS1 (370,0)
3	Robot A moves the case to the RB1_POS2	Robot A	RB1_POS2 (515,290)
4	Gripper B holds on the case	Gripper B	Gripper ON
5	Conveyor A moves to CON_POS2	Conveyor A	CON_POS2 (255)
6	Gripper A picks up the product and moves on the assembly stage	Gripper A	GR1_POS2 (280), GR1_POS1 (0)
7	Pusher A pushes the product on the case	Pusher A	PU1_POS1 (90)

Step 4: I/O definition

The configuration of the communication environment is generally tedious and complex process. Especially, there are a lot of components of the communication in the manufacturing industry. Therefore, signal I/Os must be defined for performing hardware control, testing diagnostic functionality and managing those data. In addition, it is possible to consider the number of ports of the required controllers and their positions by defining I/Os in advance. Table 3.3 is an example of I/Os for the vertical assembly workstation.

Table 3.3 Example of I/O mapping table for the vertical assembly workstation

Pin in	Mapping	Pin out	Mapping
PORT 0	SERVO ON	PORT 0	SERVO STATE
PORT 1	ORIGIN	PORT 1	EMERGENCY STOP
PORT 2	JOG+ (POSITION+)	PORT 2	ORIGIN STATE
PORT 3	JOG - (POSITION-)	PORT 3	BUSY
PORT 4	VELOCITY	PORT 4	END
PORT 5	ACCELERATION	PORT 5	IN-POSITION
PORT 6	EMERGENCY STOP	PORT 6	ALARM
PORT 7	GRIPPER ON	PORT 7	GRIPPER ON
PORT 8	PUSHER ON	PORT 8	PUSHER ON

Input data from I/O table are input variables that concerned with hardware controls such as positions, velocity, acceleration, etc. Most output data will be states of equipment to inform starting or ending tasks. In the case of the output data can be used to collect sensor data for machine diagnostic and product quality controls.

Step 5: Control software development

The control software refers to a key part of a software program that drives the workstations in the factories. To develop control software, task sequence is firstly loaded with corresponding controllable units. After that, hardware I/Os will then be initialized. When setting the initial points of the hardware I/Os, we can synchronize the virtual workstation with the control software so that it is easy to set the start points. Kinematic model can be loaded or coded for the realistic moves or specific position control. Then, input variables are specified with desired value. Finally, the control software code will be automatically generated.

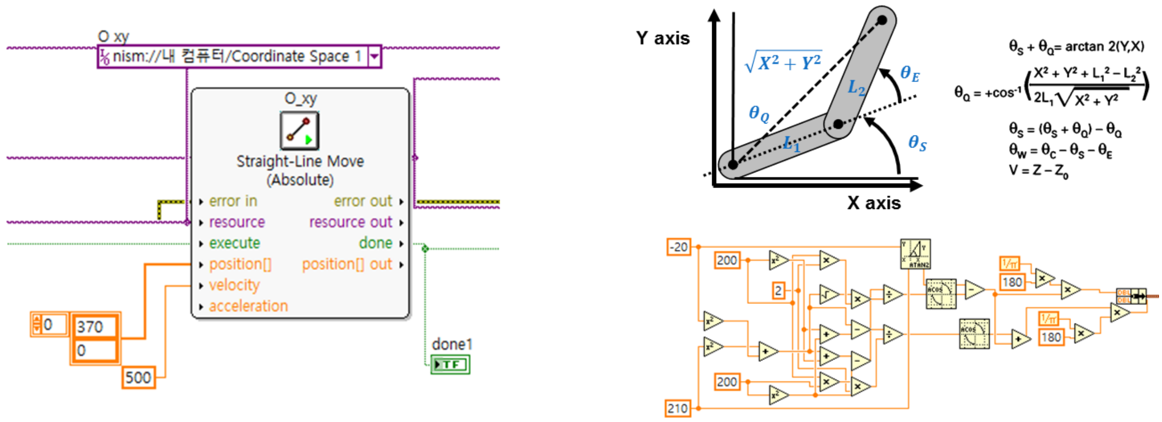

Figure 3.14 Initial parameter settings and kinematic model loading

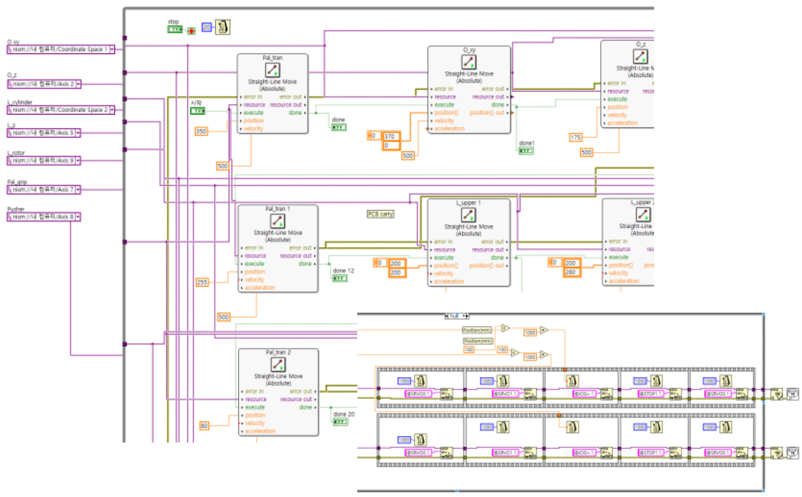

Figure 3.15 Control software codes for RCP

After virtual workstation provides communication I/O information, control software sends corresponding I/O data to the simulation model. Control software also input the specified values to the motion model information to perform the simulation and retransmit the result to the control software. Using this software, we can test and verify the trajectory concurrently in the virtual environment. In addition, we can confirm interference, task sequence, robot trajectory, and cycle-time.

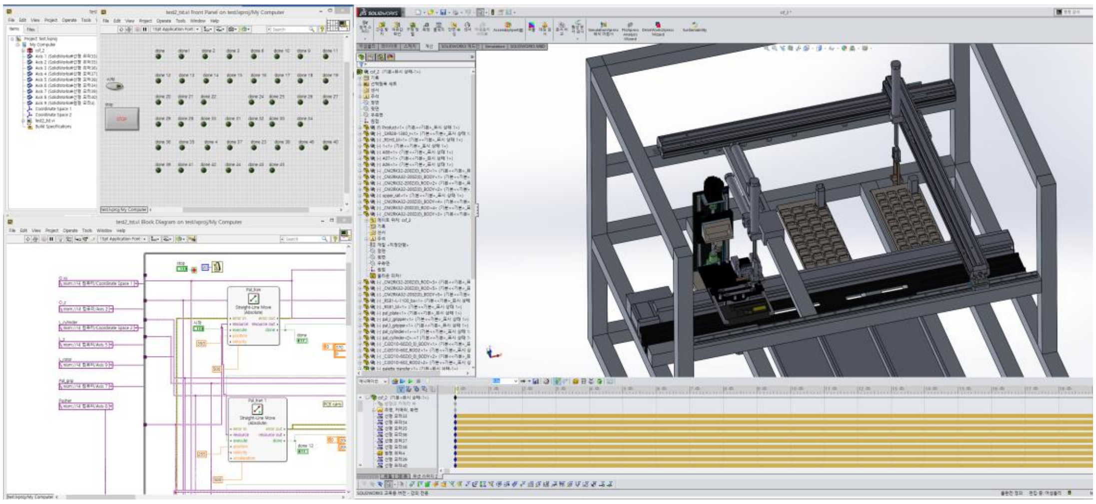

Figure 3.16 Control software simulation

3.3.2 Test and calibration

Control software will be tested and verified with real workstations iteratively. The developed control software can be easily modified by changing input variables upon request of manufacturing process change. Comparing the virtual to the real environment, there are some reasons for difference in coordinate space.

Table 3.4 Gaps between virtual and real environment

Dimensional errors	Process gaps
• Assembly errors • Machining tolerances	• Instability in the moving units of a real workstation • Different move trajectory • Noise in the electric signals

Dimensional errors generated during the construction of a real workstation have assembly errors and machining tolerances. Both errors can occur when parts in the workstation are manufactured differently from the actual drawings. Process gaps can be usually generated by controller or driver types. In this case, there are some units that move unstably in a real workstation due to their physical characteristics such as velocity, acceleration or even hardware driver properties unlike simulation results. Another process gap is a trajectory that moves differently from control codes tested in the virtual workstation. The last gap is from noise in the electric signals in control systems. For those reasons, we need to test and calibrate in the real hardware to adjust the coordinate space.

Example: Dimensional errors of an overhead crane

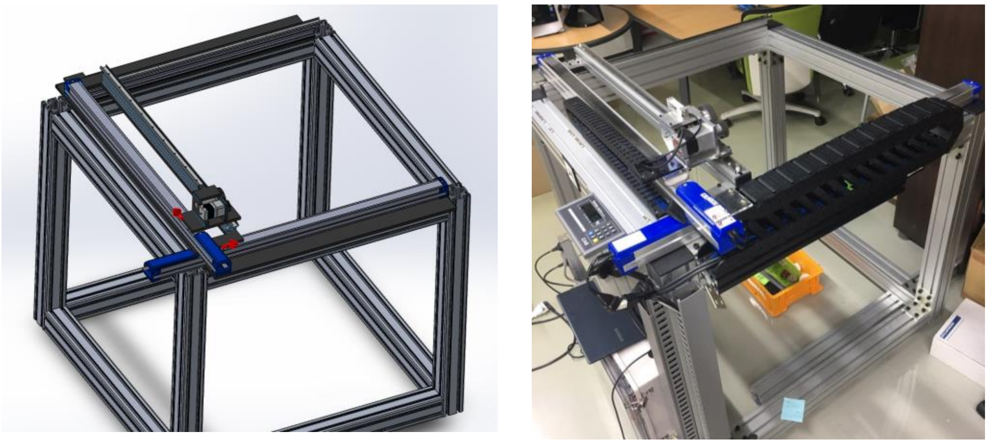

Figure 3.17 Virtual crane testbed (left) and real crane testbed (right)

To test and calibrate, we tested on crane testbed in a laboratory. We mounted the pre-designed control software into the controller and drove the real workstation hardware according to the parameter values. We first set up the properties of a single machine. Input variables such as velocity and acceleration will be initialized before operation. For calibration, we need to select reference points to compare coordinates. Reference point 1 is located on top of the south-east aluminum profile. Reference point 2 is on the edge of the crane head. In this step, we compared origin (0, 0) and (100, 100) in the x-axis and y-axis between simulation result and hardware control result.

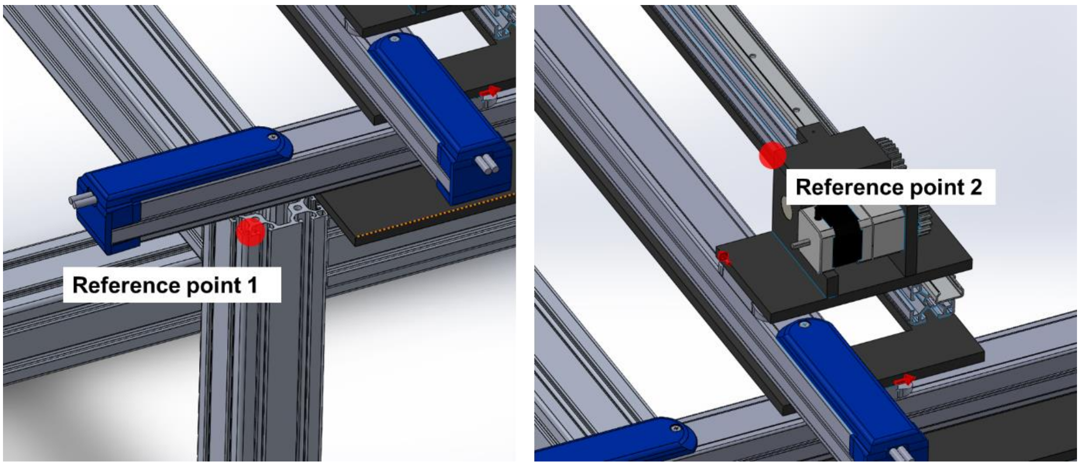

Figure 3.18 Reference points on the 3D CAD model

Through simulation, we can measure origin of x and y easily in the 3D CAD tool. By using the point values, we can mount the points and drive the real hardware. The distances of x and y axis among the reference points are 89.33mm and 144.83mm respectively in the virtual environment. However, x and y value of the real hardware that are measured physically are 130.5mm and 200.5mm respectively. By reflecting the gap of the distances on the control software, coordinate modification will be possible. This process will be conducted again on position (100, 100).

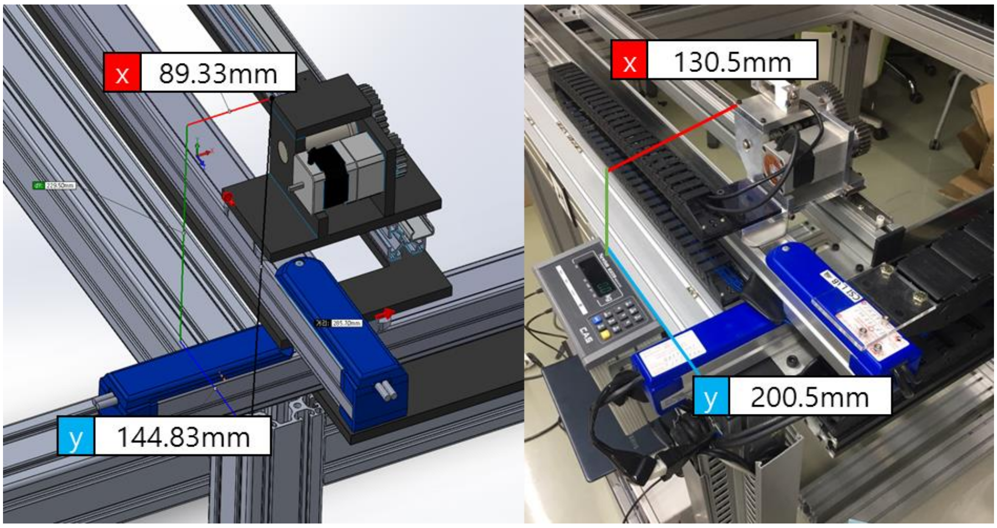

Figure 3.19 Virtual coordinate (left) and real coordinate (right)

Example: Process gaps of an overhead crane

It is difficult to reduce the process gap in automation processes because this errors are not shown in the simulation step. In the case of an overhead crane testbed, objects hanging from hooks at the end of the wire rope cannot be considered in simulation. Therefore, it is necessary to repeat the actual hardware test and software revision continuously to reduce errors.

The object was placed on the hook of the overhead crane testbed and the developed control software was executed. In the first iteration, the object hit the testbed frame by the high-acceleration of the crane. Therefore, the start and stop accelerations were halved and the second iteration was performed. In the second iteration, the object did not collide strongly with the frame, but was constantly shaken, not stabilized. In the third iteration, the acceleration was reduced to 25% and the transfer operation was performed, resulting in almost no object shaking. In the case of a process error, it is necessary to reduce the actual input variable changes repeatedly.

Once the control software is design or redesigned in the software, newly developed control software will be mounted in controllers. Then, input variables will be adjusted using a control panel while changing specific values. Finally, the logics are operated and tested on the real workstation hardware.

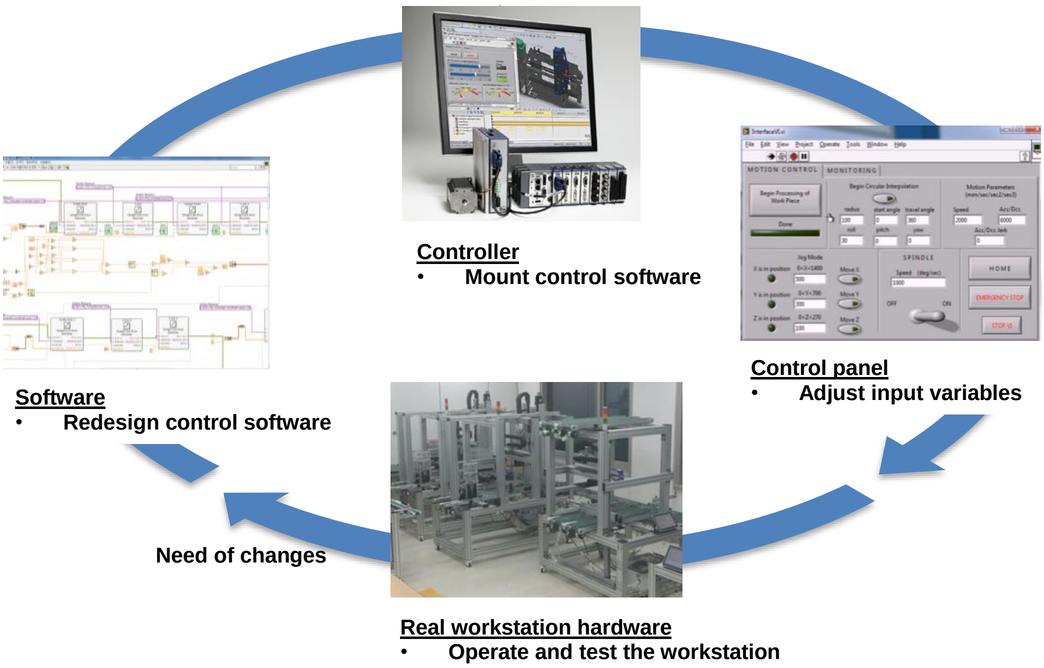

Figure 3.20 Test and verification loop

This Page Intentionally Left Blank

IV. Discussion

This paper addresses the steps from software development to actual hardware test and calibration through RCP. We already compared the development time for workstations both with RCP and without RCP in the Section 3.2. In the case of process without RCP, the entire workstation development time took a long time due to the serial development process and the long development time of control software that must need real workstations at risks of damaging hardware. On the other hands, total workstation development time with RCP is highly reduced due to parallel development process of the real workstation and the software development based on the virtual workstation. Based on those rationales, we will discuss three different experimental test cases.

We now begin to discuss about whether RCP is effective process in practical implementations. We measured the development time for three different workstations from a workstation design to test and calibration.

4.1 Development time for the overhead crane testbed

We compared the entire workstation development time for the overhead crane testbed in a laboratory. The crane testbed consists of a 3-axis gantry robot with 7 tasks. It starts from the origin and finishes the pick and place works and returns to the origin.

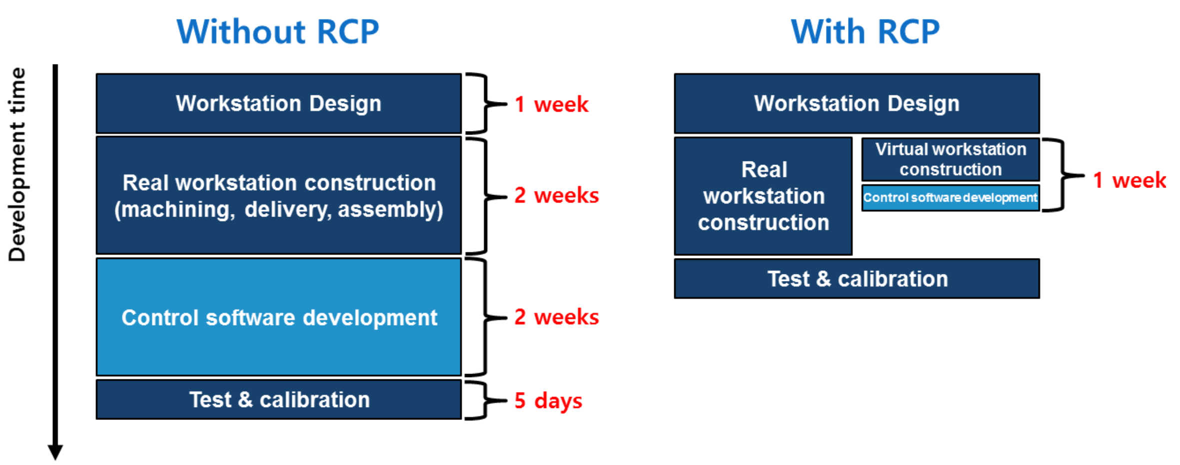

Figure 4.4 Development time for the overhead crane testbed

In the workstation design step, we determined workstation development goals and subdivided tasks. In the case of the crane testbed, the tasks were carried out using two linear actuators and trolleys, since it is a workstation simulating a shipbuilding block assembly. The work procedure was set up for the purpose of transportation, and the workstation concept design was created to produce realistic 2D or 3D drawings. When the drawing was finished, we commissioned the workpiece to fit the workpiece drawing and purchase the necessary hardware such as linear actuators and corresponding drivers. At this time, there was a delay in hardware assembly work due to machining, delivery, and assembly in actual hardware construction.

Once the hardware is built, the program is built on hardware. The program consists of simple position controls and systems to determine the weight of an object hanging the crane. In the test and calibration, the control software was modified to eliminate the dimensional errors and process gaps as mentioned in the example in Section 3.3.2.

In the case of the RCP process, the 3D models could be easily downloaded through the purchase of commercial linear actuators. This allowed us to shorten the time to build a virtual workstation. We rapidly created a control program based on RCP in a virtual environment. Due to the relatively simple task, the control program was completed a week before the actual hardware arrived.

4.2 Development time for the vertical assembly workstation

As mentioned in Section 3.3, we compared the entire workstation development time for the vertical assembly workstation in a laboratory. The workstation consists of a 2.5-axis gantry robot, a transfer system, a pusher unit based on linear actuator, and two pneumatic cylinders and grippers. It has 45 tasks including transferring, pushing, rotating, picking and placing.

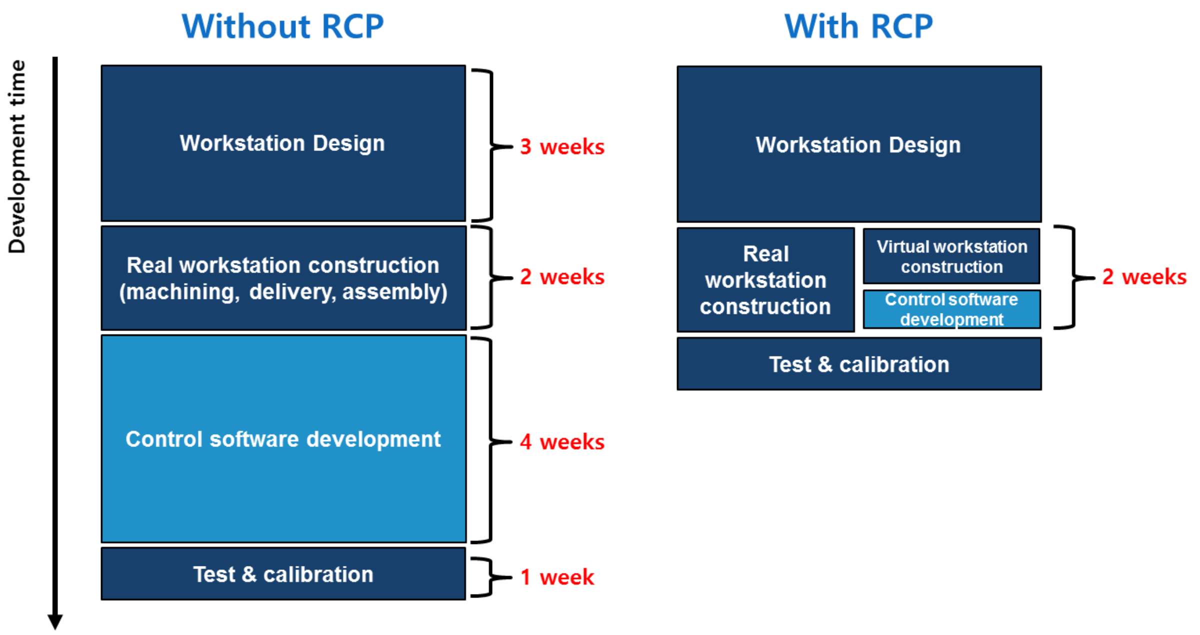

Figure 4.5 Development time for the vertical assembly workstation

The workstation design took a long time because of the many tasks and corresponding five different controller units. After the workstation design was finished, it takes two weeks to assemble the workpieces required to assemble the units, such as gripper fingers and assembly pusher fabrication, and the entire workstation assembly. In the control software development step, it took a lot of time to specify the points of each unit and design the control sequence for the many tasks.

On the other hand, for the workstation development process using RCP, it took a relatively short time to design and assemble virtual models such as cell frames, gripper fingers, and component pallets except commercial hardware such as actuators, grippers and pneumatic cylinders. Based on that virtual workstation, it took only about a week to build control software for 45 tasks. Therefore, the actual hardware and control software completed almost simultaneously while shortening the overall workstation development time.

4.3 Development time for a SCARA robot workstation

The Selective Compliance Assembly Robot Arm (SCARA) robot workstation consists of two 4-axis SCARA robots, two transfer systems and three pneumatic pushers as shown Figure 4.3. It has 37 tasks including assembling, screwing, and transferring. Unlike other experiments, the SCARA Robot workstation did not make real hardware, so we measured the control software development time through the RCP process.

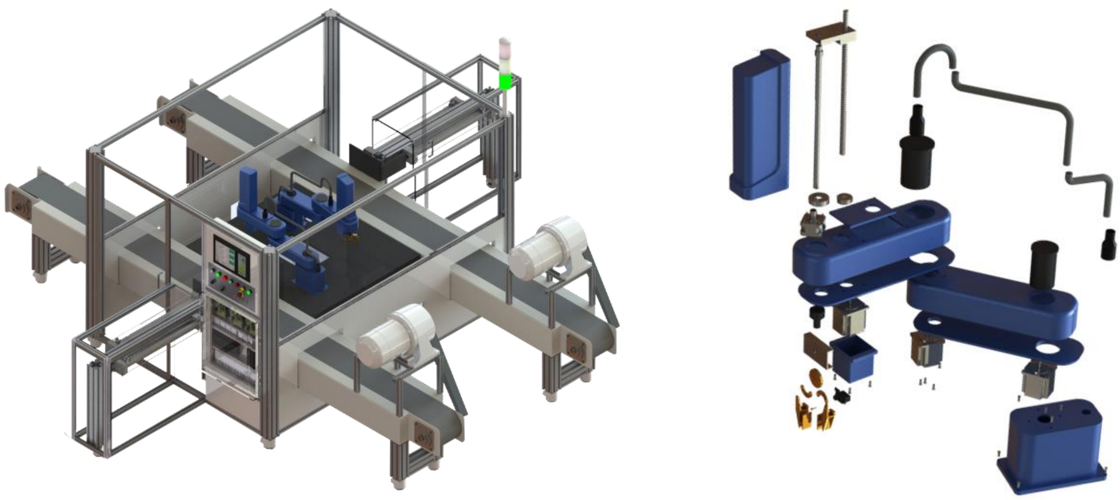

Figure 4.6 A virtual SCARA robot workstation (left) and an exploded view of SCARA robot (right)

In the case of SCARA robot workstation, the complexity of the process and the large number of parts make it time-consuming to design workstations and build virtual workstations. In the case of control software development, triangular function was applied to SCARA robot and servo gripper for controls. In order to develop the software, the workspace is coordinated with the center axis of the

SCARA robot and the driving angle of the robot is determined according to the assembly position using the inverse kinematics. Then the mathematical models were loaded into the software and the control was executed. The servo gripper also designed a program that simultaneously controls the wrist servo axis and the finger servo axis. In addition, delay was implemented to prevent interference and collision for the parallel operation of two SCARA robots.

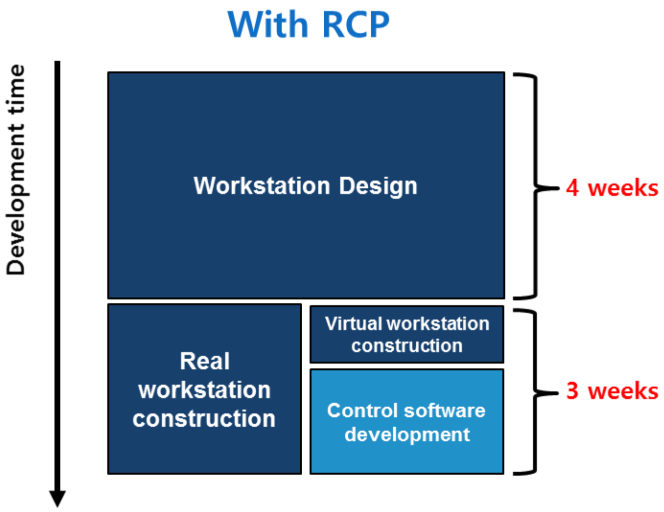

Figure 4.7 Development time for a SCARA robot workstation

When developing the control software based on such complex task sequence and trajectory, not only software development time but also real workstation construction time is significantly increased. However, with RCP, it is possible to develop the software in advance through the virtual environment and to build the actual hardware in parallel, which can significantly reduce the overall development time.

V. Conclusion and Future Research

In order to payback the installation costs of machines and factories during periods of reduced orders, a factory must produce a variety of products to meet customer needs. To achieve this goal, this thesis proposed the RCP process for rapid control software development and verification in reconfigurable workstations. The RCP procedure can be divided into two main phases: (i) 3-D factory modeling and control software design (in section 3.3.1), and (ii) test and calibration (in section 3.3.2) in the rapid factory installation procedure.

To design and test control software faster, which include task sequence and motions with I/O information to manage data in the virtual environment, a 3-D CAD model is required. Furthermore, the actuators need to be configured to control the parts that will move and how they should move. In this case, motion should be defined for a simulation model including motion types, motion constraints, and kinematic models. To generate control codes, task sequences will be designed with initial hardware parameters (e.g., actuators, motors, and pneumatic, sensors, etc.) To represent system motions with reality, developed mathematical model will be incorporated. Finally, input variables (e.g., positions, velocity, acceleration, etc.) will be specified.

In the test and calibration step, the control software can be validated in a real workstation. Through this process, we can detect gaps. In the case of dimensional errors, assembly errors or machining tolerances occur during the actual workstation construction. Process gaps, on the other hand, are caused by unstable movements of moving units in a real workstation, different motion trajectories between virtual and real workstations, and noise from electrical signals in a control system. We can minimize dimensional (position) errors and process gaps between virtual and real workstations by iteratively testing and verifying control software with real workstations.

On completion of control software design or redesign, it will be mounted on the controllers. The developed control software can easily be modified by changing input variables upon request by the manufacturing process change. The specified values in the logic will drive real machines. By means of user interface, we can test the control software and increase control accuracy while adjusting parameters in the real hardware.

The main contribution of this thesis is to reduce required the reconfiguration time of a manufacturing system to quickly respond to diverse customer demands through shortening hardware control and corresponding software development time that are very important and time-consuming tasks for manufacturing system reconfiguration by designing, testing, verifying control software both in a virtual and in a real environment. It is a practical implementation that can be used in the manufacturing industry. I proposed a method to design and test control software quickly during the time of actual hardware construction through virtual environment construction. This has shown that I can shorten the development time from workstation design to hardware construction, software development, and test and calibration. In contrast to existing methods that are the sequential processes of developing control software after commissioning actual hardware, we proposed RCP that control software can be developed in a virtual environment which reduce the risk of damaging hardware by interference or collision. In the case of a problem that cannot be found in the simulation like unstable moving by the process gaps, it can be solved by iterating input variable modifications in the control software and the actual hardware application process.

Future research will need to consider increasing control accuracy with sensor data. It will possibly improve test and calibration. In addition, one aspect not covered in this thesis is factory-in-the-loop simulation in step 4 and factory OS installation in step 5. Following the V-model of system development, the RCP process can be validated by factory-in-the-loop simulation that makes it possible to test complex real-time embedded manufacturing systems. The factory OS also allows rapid installation and implementation within hardware and software.

Reference

1. Adolfsson, J., Ng, A., Olofsgård, P., Moore, P., Pu, J., & Wong, C.-B. (2002). Design and simulation of component-based manufacturing machine systems. *Mechatronics, 12*(9), 1239-1258.
2. Ahmad, M., Ahmad, B., Harrison, R., Alkan, B., Vera, D., Meredith, J. O., & Bindel, A. (2016). A framework for automatically realizing assembly sequence changes in a virtual manufacturing environment. *Procedia CIRP*.
3. Andersen, R. S., Damgaard, J. S., Madsen, O., & Moeslund, T. B. (2013). Fast calibration of industrial mobile robots to workstations using QR codes. *Proceedings of the 2013 44th International Symposium on Robotics*.
4. Andrisano, A. O., Leali, F., Pellicciari, M., Pini, F., & Vergnano, A. (2012). Hybrid Reconfigurable System design and optimization through virtual prototyping and digital manufacturing tools. *International Journal on Interactive Design and Manufacturing, 6*(1), 17-27.
5. Antzoulatos, N., Castro, E., Scrimieri, D., & Ratchev, S. (2014). A multi-agent architecture for plug and produce on an industrial assembly platform. *Production Engineering, 8*(6), 773-781.
6. Arai, T., Aiyama, Y., Maeda, Y., Sugi, M., & Ota, J. (2000). Agile assembly system by "plug and produce". *CIRP Annals-Manufacturing Technology, 49*(1), 1-4.
7. Azab, A., ElMaraghy, H., Nyhuis, P., Pachow-Frauenhofer, J., & Schmidt, M. (2013). Mechanics of change: A framework to reconfigure manufacturing systems. *CIRP Journal of Manufacturing Science and Technology, 6*(2), 110-119.
8. Bensmaine, A., Dahane, M., & Benyoucef, L. (2014). A new heuristic for integrated process planning and scheduling in reconfigurable manufacturing systems. *International Journal of Production Research, 52*(12), 3583-3594.
9. Bi, Z. M., Lang, S. Y., Shen, W., & Wang, L. (2008). Reconfigurable manufacturing systems: the state of the art. *International Journal of Production Research, 46*(4), 967-992.
10. Brusaferri, A., Ballarino, A., Cavadini, F. A., Manzocchi, D., & Mazzolini, M. (2014). CPS-based hierarchical and self-similar automation architecture for the control and verification of reconfigurable manufacturing systems. *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation*.
11. Büscher, C., Voet, H., Meisen, T., Krunke, M., Kreisköther, K., Kampker, A., Schilberg, D., & Jeschke, S. (2014). Improving Factory Planning by Analyzing Process Dependencies. *Procedia CIRP, 17*, 38-43.
12. Bucher, R., & Balemi, S. (2006). Rapid controller prototyping with Matlab/Simulink and Linux. *Control Engineering Practice, 14*(2), 185-192.
13. Chen, C.-H., Tsai, H.-L., & Tu, J.-C. (2004). Robot control system implementation with rapid control prototyping technique. *Proceedings of the 2004 IEEE International Symposium on Computer Aided Control Systems Design*.
14. Chen, I.-M. (2001). Rapid response manufacturing through a rapidly reconfigurable robotic workcell. *Robotics and Computer-Integrated Manufacturing, 17*(3), 199-213.
15. Choi, S., Sung, N., Shin, Y., & Noh, S. D. (2014). The Integrated Design and Analysis of Manufacturing Lines (II)-Continuous Design, Analysis and Optimization through Digital Virtual Manufacturing. *Transactions of the Society of CAD/CAM Engineers, 19*(2), 148-156.
16. Covanich, W., & McFarlane, D. (2009). Assessing ease of reconfiguration of conventional and Holonic manufacturing systems: Approach and case study. *Engineering Applications of Artificial Intelligence, 22*(7), 1015-1024.
17. Drake, G. R., & Smith, J. S. (1996). Simulation system for real-time planning, scheduling, and control. *Proceedings of the 28th Conference on Winter simulation*.
18. ElMaraghy, H., & ElMaraghy, W. (2016). Smart Adaptable Assembly Systems. *Procedia CIRP, 44*, 4-13.
19. ElMaraghy, H. A. (2005). Flexible and reconfigurable manufacturing systems paradigms. *International Journal of Flexible Manufacturing Systems, 17*(4), 261-276.
20. Farid, A. M. (2013). An axiomatic design approach to production path enumeration in reconfigurable manufacturing systems. *2013 IEEE International Conference on Systems, Man, and Cybernetics*.
21. Farrugia, P., Francalanza, E., Attard, G., & Borg, J. (2010). Factory Planning Through Paper-based Computer-Aided Sketching. *Integrated Design and Manufacturing in Mechanical Engineering - Virtual Concept*.
22. Faschang, M., Kupzog, F., Mosshammer, R., & Einfalt, A. (2013). Rapid control prototyping platform for networked smart grid systems. *Proceedings of the 2013 39th Annual Conference of the IEEE Industrial Electronics Society*.
23. Goyal, K. K., Jain, P. K., & Jain, M. (2013). A novel methodology to measure the responsiveness of RMTs in reconfigurable manufacturing system. *Journal of Manufacturing Systems, 32*(4), 724-730.

24. Grepl, R. (2011). Real-Time Control Prototyping in MATLAB/Simulink: Review of tools for research and education in mechatronics. *Proceedings of the 2011 IEEE International Conference on Mechatronics*.
25. Grepl, R., & Lee, B. (2010). Model Based Controller Design for Automotive Electronic Throttle *Recent Advances in Mechatronics* (pp. 209-214): Springer.
26. Heilala, J., & Voho, P. (2001). Modular reconfigurable flexible final assembly systems. *Assembly Automation, 21*(1), 20-30.
27. Helo, P., Suorsa, M., Hao, Y., & Anussornnitisarn, P. (2014). Toward a cloud-based manufacturing execution system for distributed manufacturing. *Computers in Industry, 65*(4), 646-656.
28. Hoffman, K., Basson, A. H., & le Roux, A. (2014). Towards Alternatives for Agent Based Control in Reconfigurable Manufacturing Systems *Enabling Manufacturing Competitiveness and Economic Sustainability* (pp. 237-242): Springer.
29. Imtiaz, J., & Jasperneite, J. (2013). Scalability of OPC-UA down to the chip level enables "Internet of Things". *2013 11th IEEE International Conference on Industrial Informatics*.
30. Jatzkowski, J., & Kleinjohann, B. (2014). Towards self-reconfiguration of real-time communication within Cyber-Physical Systems. *Procedia Technology, 15*, 54-61.
31. Kádár, B., Terkaj, W., & Sacco, M. (2013). Semantic Virtual Factory supporting interoperable modelling and evaluation of production systems. *CIRP Annals-Manufacturing Technology, 62*(1), 443-446.
32. Kampker, A., Kreisköther, K., Burggräf, P., Meckelnborg, A., Krunke, M., Jeschke, S., & Hoffmann, M. (2013). Value-oriented layout planning using the Virtual Production Intelligence (VPI). *Proceedings of the 24th Annual Production and Operations Management*.
33. Kim, S.-C., & Choi, K.-H. (2000). Development of Flexible Manufacturing System using Virtual Manufacturing Paradigm. *International Journal of the Korean Society of Precision Engineering, 1*(1), 84-90.
34. Kimura, A., & Maeda, I. (1996). Development of engine control system using real time simulator. *Proceedings of the 1996 IEEE International Symposium on Computer-Aided Control System Design*.
35. Koren, Y., Heisel, U., Jovane, F., Moriwaki, T., Pritschow, G., Ulsoy, G., & Van Brussel, H. (1999). Reconfigurable manufacturing systems. *CIRP Annals-Manufacturing Technology, 48*(2), 527-540.
36. Krovi, V., Kumar, V., Ananthasuresh, G., & Vezien, J.-M. (1999). Design and virtual prototyping of rehabilitation aids. *Journal of Mechanical Design, 121*(3), 456-458.
37. Kühn, W. (2006). Digital factory: simulation enhancing the product and production engineering process. *Proceedings of the 38th Conference on Winter simulation*.
38. Landers, R. G., Min, B.-K., & Koren, Y. (2001). Reconfigurable machine tools. *CIRP Annals-Manufacturing Technology, 50*(1), 269-274.
39. Lapusan, C., Matis, V., Balan, R., Hancu, O., Stan, S., & Lates, R. (2008). Rapid control prototyping using Matlab and dSpace. Application for a planar parallel robot. *2008 IEEE International Conference on Automation, Quality and Testing, Robotics*.
40. Lee, W.-T., & Park, S.-B. (2006). Technical Trends of an Automotive Electronic Controller Development using Real-time Simulation Technique. *Journal of the Korean Society for Precision Engineering, 23*(9), 23-30.
41. Leitão, P., Mendes, J. M., & Colombo, A. W. (2009). Smooth migration from the Virtual design to the real manufacturing control. *2009 7th IEEE International Conference on Industrial Informatics*.
42. Maeda, Y., Kikuchi, H., Izawa, H., Ogawa, H., Sugi, M., & Arai, T. (2003). An easily reconfigurable robotic assembly system. *Proceedings of 2003 IEEE International Conference on Robotics and Automation*.
43. Mauch, S., Reger, J., Reinlein, C., Appelfelder, M., Goy, M., Beckert, E., & Tünnermann, A. (2014). FPGA-accelerated adaptive optics wavefront control. *Proceedings of SPIE 8978 MEMS Adaptive Optics*.
44. Mehrabi, M. G., Ulsoy, A. G., & Koren, Y. (2000). Reconfigurable manufacturing systems and their enabling technologies. *International Journal of Manufacturing Technology and Management, 1*(1), 114-131.
45. Menager, N., Worschech, N., & Mikelsons, L. (2014). A toolchain for Rapid Control Prototyping using Rexroth controllers and open source software. *Proceedings of the 10th International Modelica Conference*.
46. Michalos, G., Sipsas, P., Makris, S., & Chryssolouris, G. (2016). Decision making logic for flexible assembly lines reconfiguration. *Robotics and Computer-Integrated Manufacturing, 37*, 233-250.
47. Moore, P., Pu, J., Ng, H., Wong, C., Chong, S., Chen, X., Adolfsson, J., Olofsgård, P., & Lundgren, J.-O. (2003). Virtual engineering: an integrated approach to agile manufacturing machinery design and control. *Mechatronics, 13*(10), 1105-1121.
48. Mourtzis, D., Papakostas, N., Mavrikios, D., Makris, S., & Alexopoulos, K. (2015). The role of simulation in digital manufacturing: applications and outlook. *International journal of computer integrated manufacturing, 28*(1), 3-24.
49. Nagaraj, S. C., & Detrick, B. (2009). HIL and RCP tools for embedded controller development in hybrid vehicles. *Proceedings of 2009 IEEE Vehicle Power and Propulsion*.
50. Naumann, M., Wegener, K., & Schraft, R. D. (2007). Control architecture for robot cells to enable Plug'n'Produce. *Proceedings of 2007 IEEE International Conference on Robotics and Automation*.

51. Nayroles, B., Touzot, G., & Villon, P. (1992). Generalizing the finite element method: diffuse approximation and diffuse elements. *Computational mechanics, 10*(5), 307-318.
52. Otto, J., Böttcher, B., & Niggemann, O. (2013). Plug-and-Produce: Semantic Module Profile. *Proceedings of 2013 Dagstuhl-Workshop "Modellbasierte Entwicklung eingebetteter Systeme"*.
53. Park, S. C., & Chang, M. (2012). Hardware-in-the-loop simulation for a production system. *International Journal of Production Research, 50*(8), 2321-2330.
54. Park, T.-K., Kim, G.-Y., Noh, S.-D., & Park, Y.-J. (2005). Virtual Manufacturing for an Automotive Company (V)-Parametric Modeling of the Digital General Assembly Shop using Object-Oriented Methods. *Journal of Korean Institute of Industrial Engineers, 18*(1), 94-103.
55. Reinhart, G., & Krug, S. (2012). Automatic Configuration (Plug & Produce) of Robot Systems–Data-Interpretation and Exchange *Enabling Manufacturing Competitiveness and Economic Sustainability* (pp. 147-152): Springer.
56. Reinhart, G., Krug, S., Hüttner, S., Mari, Z., Riedelbauch, F., & Schlögel, M. (2010). Automatic configuration (plug & produce) of industrial ethernet networks. *Proceedings of the 2010 9th IEEE/IAS International Conference on Industry Applications*.
57. Rossmann, J., Schluse, M., Schlette, C., & Waspe, R. (2012). Control by 3d simulation–a new erobotics approach to control design in automation. *Proceedings of International Conference on Intelligent Robotics and Applications*.
58. Rubaai, A., Castro-Sitiriche, M. J., & Ofoli, A. R. (2008). Design and implementation of parallel fuzzy PID controller for high-performance brushless motor drives: an integrated environment for rapid control prototyping. *IEEE transactions on industry applications, 44*(4), 1090-1098.
59. Schmitt, M., Loskyll, M., & Zuehlke, D. (2014). Development of a Framework for Dynamic Function Deployment and Extension by Using Apps on Intelligent Field Devices. *IFAC Proceedings Volumes, 47*(3), 2611-2616.
60. Schuh, G., Kampker, A., & Wesch-Potente, C. (2011). Condition based factory planning. *Production Engineering, 5*(1), 89-94.
61. Shen, W., & Norrie, D. H. (1999). Agent-based systems for intelligent manufacturing: a state-of-the-art survey. *Knowledge and information systems, 1*(2), 129-156.
62. Sims, J. E., Chu, B. T. B., Long, J., Matthews, M., Barnes, J. G., Jones, C. H., Anderson, R. A., Lambert, R., Drake, D. C., & Hamilton, M. A. (1997). Framework for adaptive interoperability of manufacturing enterprises (FAIME): a case study. *Proceedings of International Society for Optics and Photonics*.
63. Steiger, C., Walder, H., & Platzner, M. (2004). Operating systems for reconfigurable embedded platforms: Online scheduling of real-time tasks. *IEEE Transactions on computers, 53*(11), 1393-1407.
64. Subramanian, R., Venhovens, P., & Keane, B. P. (2012). Accelerated design and optimization of battery management systems using HIL simulation and Rapid Control Prototyping. *Proceedings of the 2012 IEEE International Electric Vehicle Conference*.
65. Subramanian, S., Thangavel, P., MI, F. S., Sornam, K., Rambhaji, G. P., & Velusamy, R. (2015). Development and Testing of a Control Algorithm to Assist Drive-Off in the Gradient-A Rapid Control Prototyping Approach. *Proceedings of the 2012 IEEE International Electric Vehicle Conference*.
66. Tolio, T., Sacco, M., Terkaj, W., & Urgo, M. (2013). Virtual factory: An integrated framework for manufacturing systems design and analysis. *Procedia CIRP, 7*, 25-30.
67. Tursini, M., Di Leonardo, L., Olivieri, C., & Della Loggia, E. (2013). Rapid Control Prototyping of IPM Drives by Real Time Simulation. *Proceedings of 2013 8th EUROSIM Congress on Modelling and Simulation*.
68. Vermaak, H., & Niemann, J. (2015). Validating a reconfigurable assembly system utilizing virtual commissioning. *Proceedings of Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference*.
69. Wang, G. G. (2002). Definition and review of virtual prototyping. *Journal of Computing and Information Science in engineering, 2*(3), 232-236.
70. Wu, Y., Klimchik, A., Caro, S., Furet, B., & Pashkevich, A. (2015). Geometric calibration of industrial robots using enhanced partial pose measurements and design of experiments. *Robotics and Computer-Integrated Manufacturing, 35*, 151-168.
71. Yi, Y., Nousias, I., Milward, M., Khawam, S., Arslan, T., & Lindsay, I. (2006). System-level scheduling on instruction cell based reconfigurable systems. *Proceedings of the Design Automation & Test in Europe Conference*.
72. Zhang, J., Khalgui, M., Li, Z., Frey, G., Mosbahi, O., & Salah, H. B. (2015). Reconfigurable coordination of distributed discrete event control systems. *IEEE Transactions on Control Systems Technology, 23*(1), 323-330.
73. Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence, 22*(11), 1330-1334.
74. Zimmermann, U. E., Bischoff, R., Grunwald, G., Plank, G., & Reintsema, D. (2008). COMMUNICATION,

CONFIGURATION, APPLICATION. *International Conference on Informatics in Control, Automation, and Robotics*.

75. Zuehlke, D. (2010). SmartFactory—Towards a factory-of-things. *Annual Reviews in Control, 34*(1), 129-138.

Acknowledgement

I would like to express my deep gratitude to all those who have helped me to accomplish this work successfully. Professionally and personally, the people mentioned below made a great contribution to the achievement of this work but I do not come up with a word to say a bigger gratitude to them.

I owe big recognition to Professor Duck-Young Kim as an advisor, who supported and guided me. He gave me an interest in Smart Factory and my research, and supervised me as a mentor based on his academic and personal philosophy. Without his contribution, this work would never have been successful. I will become an outstanding student whom you will be proud of. Thank you.

I also would like to express my deep gratitude to two committees, Professor Daeil Kwon and Sang-Hoon Kang. Thank you for your consideration of the thesis. Thanks to professors' informative and valuable feedbacks, I can develop my thesis further and finally complete it.

I would like to express my gratitude to my colleagues who have been working together. I got a lot of valuable help from all of them both seniors and juniors. Especially, I would like to express my deep gratitude to Dr. Park who said he would let me graduate no matter what and gave me more sticks than carrots. And Won-il, I sometimes told you how I can finish my master without you. When I was frustrated while controlling the hardware, you sympathized with it and calmed me down. Thank you brother! And Sujeong as a group leader for a long time, she gave us a "sacrifice" in the lab. She has given the biggest helps and sharp comments as a senior in research life. Woon Sang, who is full of fun and mischief otherwise he gave me enriching discussions in my research. Jong-il, you kindly inform me of the tools I first deal with and you always bucked me up. Hye-Rim and Ki-Bum, you gave me a consideration to finish this work while taking most of my jobs. I was (and will be) happy to work with you during CSF project. Ha-young, you spared your kindness and knowledge when someone needs help including me. Finally, Hansol and Ki-Chang! Your brightness make my laboratory life enjoyable. Thank you all!

I owe a very special thanks to my best friends. Jeong-Hoon, a senior graduate student and a doctoral student in the US, I cannot forget your word that positive mind is important in research. Ji-Ho always encouraged me to study for the future. Jun-Young, who greeted me in my hometown when I was low.

Finally, infinite thanks to my parents for the belief and all support and my sister. Father, thank you for believing my choice and generous support. Mother, without the patience and education of you, this moment would never have come. Thank you for your unconditional love. And my sister, it was your number that I called first when I was hardest. I relied on you mentally even though I suppressed my feelings. I'm a bit shy, but I'll tell you to thank you for borrowing this moment. Thank you so much.

There are so many thankless people who have not mentioned here yet. I apologize for not being able to engrave their names and I would rather finish these words with my deepest gratitude.

"Thank you very much."