N° d'ordre 4892 Année 2013

Thèse

Contribution to engineering change management in product development projects :

Reference models and multi-agent-based simulation

Présentée devant :
L'Université Bordeaux 1
Pour obtenir :
Le grade de docteur
Spécialité :
PRODUCTIQUE
Formation doctorale :
PRODUCTIQUE
École doctorale :
ÉCOLE DOCTORALE
DES SCIENCES PHYSIQUES ET DE L'INGÉNIEUR
Par :
Xin ZHANG

Soutenue publiquement le 19 novembre 2013 devant le jury composé de :

Claude Baron, Professeur à l'INSA de Toulouse .... Rapporteur
Emmanuel Caillaud, Professeur à l'ENSAM-Metz .... Rapporteur
Claudia Eckert, Senior lecturer à l'Open University (UK) .... Examinateur
Patrice Leclaire, Maître de Conférences à SUPMECA Paris .... Examinateur
Marc Zolghadri, Professeur à SUPMECA Paris .... Directeur de thèse
Philippe Girard, Professeur à l'Université Montesquieu Bordeaux IV .. Co-Directeur de thèse

Laboratoire de l'Intégration du Matériau au Système

To My Wife, Yan.

Foreword

Acknowledgements

First and foremost, I would like to thank my thesis supervisor Professor Marc Zolghadri for his trust, encouragement, support and inspiration throughout this Ph.D. research. Without his consistent and illuminating instructions, the completion of the research would not have been possible. I am also sincerely grateful to my co-supervisor, Professor Philippe Girard, for the guidance he showed me throughout my study.

I am indebted to Dr. Claudia Eckert (The Open University) and Dr. Chris Earl (The Open University) for their constructive comments to my research work. I also appreciate very much for their helpful and patient guidance during my visiting stay in The Open University.

I am also sincerely grateful to Dr. Patrice Leclaire for the support and suggestions he provided during composing the thesis. I enjoyed having lively discussions and co-work with him.

I would like to extend my sincere gratitude to Professor Claude Baron, Professor Emmanuel Caillaud, the rapporteurs of my thesis committee for spending their precious effort to examine this dissertation.

Acknowledgements are also made to all the colleagues of Productique Group in IMS Lab for their social and academic support during my Master and Ph.D. study.

Last but not the least, I would like to thank my family, Yan and Ayra. It is their unconditioned love and unwavering support that keep me going.

Abstract

The overall goal of this Ph.D. research is to provide reference models, support methods and tools that simulate change propagations in a Product Development (PD) project to assist decision-makings. We establish a change analysis framework of modeling the context of change occurrence and propagation by taking into account the multiple knowledge areas of PD project simultaneously. Under this framework, we propose the conceptual models that provide a qualitative method to identify change and change propagation. We identify the main change propagation mechanisms. Relying on that, we suggest the procedures of building up the change propagation networks. Within this network, we propose the methodology to simulate change propagation mechanisms. We present then the process of implementing these concepts in a software prototype by using multi-agent based technology. A detailed illustrative case is provided to show change propagation mechanisms.

Résumé

L'objectif de cette thèse est de fournir des modèles de référence, méthodes et outils permettant de simuler loccurrence et la propagation de changements dans des projets de développement de produit (PD). Nous avons établi un cadre d'analyse des changements afin de modéliser le contexte de loccurrence de ceux-ci en prenant en compte les multiples domaines du projet PD, à savoir produit, projet, réseau de partenaires. Nous proposons alors des modèles conceptuels du changement et définissons des méthodes qui permettent leur détection suite à loccurrence des événements internes ou externes. Ces modèles permettent également de caractériser la propagation des changements au sein dun domaine ou entre les trois domaines d'un projet PD. Ces résultats nous permettent de modéliser les propagations de changements à travers dun réseau de constructs. Après une caractérisation précise de ces mécanismes de propagation, nous avons proposé la réalisation dun prototype de simulation basé sur la technologie multi-agents. Ces concepts sont alors illustrés à travers un exemple détaillé.

Table of Content

List of Figures

List of Tables

1 Introduction

▷

"For want of a nail the shoe was lost for want of a shoe the horse was lost ; and for want of a horse the rider was lost ; being overtaken and slain by the enemy, all for want of care about a horse-shoe nail."

(The Way to Wealth(1758) Benjamin Franklin)

Change and its propagation are key issues in product development projects, which can results in not only chance but also risk. In this chapter, we firstly introduce the background of our research on change and change propagation, which is followed by the problem statement. Then we present the research questions and research approach. This chapter ends with the presentation of the thesis structure.

◁

Plan of Chapter

1.1 Background

With the business context growing, companies are facing more and more challenges from product production management and supply chain management simultaneously during Product Development (PD) project. A PD project is defined as *an endeavor of activities beginning with the perception of a market opportunity and ending in the production, sale, and delivery of a product* (Ulrich & Eppinger, 2007). On one hand, products are expected to be designed and developed in a shorter period to satisfy the requirements as far as possible, whereas on the other hand the concerned partners should establish an effective and efficient communication to achieve their own objectives and take the profit through the Product Development project. Considering these aspects, one of the important issues is the management of changes. Generally, a change is an *alteration* in attributes or parameters of a product, activities, and adjunct resources during the various phases of the life-cycle of product and PD project, see Huang *et al.* (2000). For instance, "increasing the size of cylindre of an engine", "Changing the colour of the body of a hairdryer", "replacing the Supplier A by the Supplier B of the tires", or "prolonging the start date of the project/activity" are some examples of the changes we are dealing with. Therefore, the changes could refer to products, to partners of the project, or the project structure and attributes (time, delay, dependency, etc.). During a PD project, changes reveal multiple senses. Changes could bring the opportunity to a company for innovations. However, changes could also enhance the risk of failing to release the final product within the constraints (lead time, requirements, cost, etc.). Correspondingly, the influences from changes can be classified and described by the focal company as positive and negative. Here the focal company refers to a company from whose perspective the supply chain is analysed. The focal company plays a central role of information and material flows exchanges (Hanf & Pall, 2009). Therefore, the change management during the product development project generally aims at supporting the focal company to cope with alterations through the PD project in order to achieve the objectives.

Along the life-cycle of product, changes can occur at any time. Given the occurrence of a change, the later it is coped with, the more expensive the corrective actions would be, according to *Rule of Ten* (Clark & Fujimoto, 1991). In a stage-gateway process of product development, there exists a set of freezes along the phases (Eger *et al.*, 2005). Considering this, the concerned objects (parts, activities or capacities etc.) during the design phase are proposed to be frozen early to reduce the risk of rework (Zangwill, 1993). However, there also exists some argument that "early freezes" would lead to a poor design (Bhattacharya *et al.*, 1998). Therefore, the timing of freeze should be considered carefully to reduce and control changes during the process of product design and development. Therefore, coping with changes is under various constraints which implies a number of strategies and operations, such as preventing technical changes through employing better communication and discipline (Clark & Fujimoto, 1991), detecting emerging changes earlier, i.e., "front-loading" changes (Fricke *et al.*, 2000) (Nichols, 1990), designing system "easy to change", i.e., incorporating changeability into system architecture (Fricke & Schulz, 2005).

Moreover, one change can cause the occurrence of other changes, and it is to say that the change is propagated through the potential relations between the parts, roles, activities and any other element involved in the product development project. Eckert *et al.* (2004) pointed out that the link between the composition of a system (such as a complex product) can reflect the relations of changes. For example, the geometric links between a cylinder and a piston can be regarded as a change propagation channel in which the effect of change could be transferred (see Figure 1.1).

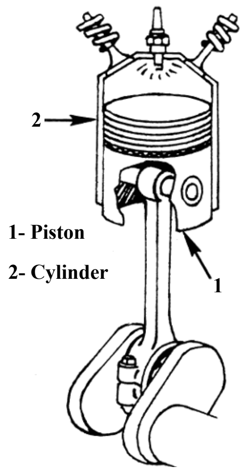


FIGURE 1.1 – An example (source: http://www.wpclipart.com)

Considering the complexity of the product development project and the involved aspects, the changes occurring on the involved elements and their inter-relations compose a network where the nodes represent elements and their relations are modelled by the edges. We specify this network as *change propagation network* and the edges as potential *change propagation channels*. When a change occurs on a given node, it could be coped with locally or transferred to other node(s) or both. Referring to the occurred changes and the potential propagations brought by it, the focal company has multiple choices, and we generally categorize them as *inner-company* and *outer-company* change handlings. The former, i.e., inner-company change handling, indicates the focal company would cope with the changes all by itself and eliminate the influences of the change, whereas the outer-company change handling means that the focal company would spread the changes to its partners and/or cooperate with them to cope with the change.

1.2 Problem Statement

Due to the motivations of upgrading product or satisfying requirements from the market, companies attempt to earn benefit through releasing a product with partial improvement or a totally new designed one. Changes could also lead in the occurrence of some unexpected or unforeseen events during the PD project. Being unable to properly manage changes can imperil entire projects.

Referring to the same occurred change, different companies would hold their different perspectives which may bring about different approaches of managing changes, strategies of product development. When looking into the life-cycle of product, changes can occur at any time in spite of they are foreseen or unforeseen. Furthermore, as mentioned before, a change can cause another one referring to the change propagation (cf. (Eckert *et al.*, 2004)). In a project, change could propagate, involving multiple aspects of the project : the product and constituent components, project activities, and involved partners. The impact of the original change could either be amplified or reduced through the propagation. Thus, changes are expected to be perceived, analyzed and coped with or even controlled as soon as possible.

Considering the above issues, the overall goal of this Ph.D. research is to provide support methods and tools that simulate change and change propagation in a PD project to assist decision-makings. They can be used to identify, analyse and cope with change and change propagation by taking the multiple domains of a PD project into account. We will make contributions to both academic and industrial sectors by achieving the following goals :

Academic and industrial

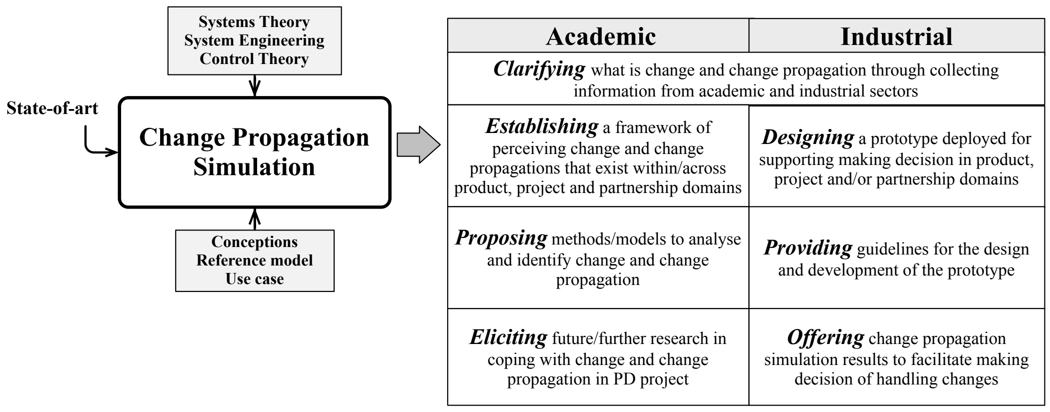
1. Clarifying what is change and change propagation through collecting information and knowledge from academic and industrial sectors ;

Academic

1. Establishing a framework of perceiving change and change propagation that exist either within or across product, project and partnership management knowledge areas ;
2. Proposing methods/models to analyse and identify change and change propagation ;
3. Eliciting the future/further research in coping with change and change propagation in Product Development (PD) project.

Industrial

1. Designing a prototype to support making decision in product, project and/or partnership domains ;
2. Providing guidelines for the design and development of the prototype ;
3. Offering change propagation simulation results to facilitate making decision of handling changes during product, project and partnership management.



Academic	Industrial
Clarifying what is change and change propagation through collecting information from academic and industrial sectors	
Establishing a framework of perceiving change and change propagations that exist within/across product, project and partnership domains	***Designing*** a prototype deployed for supporting making decision in product, project and/or partnership domains
Proposing methods/models to analyse and identify change and change propagation	***Providing*** guidelines for the design and development of the prototype
Eliciting future/further research in coping with change and change propagation in PD project	***Offering*** change propagation simulation results to facilitate making decision of handling changes

FIGURE 1.2 – Goals of Change Propagation Simulation

1.3 Research Questions

After defining, in sections 1.1 and 1.2, why do we need to take account of changes in the most efficient way, in this section we will describe precisely the questions we tried to answer.

1. *What exactly is meant by change and change propagation ?*

Generally, change is considered as an act or process through which something becomes different while change propagation is considered as a phenomena that a change leads to another change. But under different subjects and scenarios, they have more specific definitions and meanings. Therefore, we turn to a literature exploration to clarify what is change and change propagation in our research, and further to position our research in the existing related literature work.

2. *What is the context of change occurrence and propagation ?*

A PD project refers simultaneously to several management domains that are named as "*knowledge areas*" to underline the fact that the systematic influence of decisions made in product development could be brought into the supply chain design/development and vice versa. The considered knowledge areas in this thesis are :
– *Product management knowledge area ;*
– *Project management knowledge area ;*
– *Partnership management knowledge area.*

To analyze the context where change occurs and propagates, it is necessary to understand the knowledge areas of PD project. To achieve this purpose, we propose a framework that models a PD project considering product management, project management, and partnership management knowledge areas simultaneously as well as the interrelations between them.

3. *How to identify change and change propagation ?*

To identify change and its propagation, it is important to clarify what are the profound causes of changes, under what conditions change propagates, and how change propagates. To answer these questions, we propose a conceptual model of change and change propagation under the framework we mentioned in question 2. It also provides a qualitative method of perceiving and specifying change and change propagation. Furthermore, it enables us to establish a change propagation network in which the propagation patterns are studied.

4. *How to simulate change propagation in an efficient way ?*

Simulating change propagations involves two aspects : the methodology of simulation and the techniques to implement the simulation. The methodology of simulating change propagations states the procedures of identifying changes, determining actions of coping with changes, planning change propagations as well as representing the outcome. With the methodology, the behaviour of changes and the process of change propagation during PD project are simulated to present the various scenarios in which change propagates in multiple possibilities. To do so, we select the agent-based techniques. This technique enables us to abstract and model a complex system effectively and efficiently. Meanwhile, with integrating the graphical user interface, the simulation procedures could be displayed.

5. *How to represent and verify the research results ?*

In order to present our research results, we design and develop an agent-based software prototype that implements the entire methodology and provides graphical representations to demonstrate the research results. To assure the correctness of our methods and prototype, on one hand, we use simple cases throughout the entire thesis to explain and verify the methods. On the other hand, we apply a more complex case to the method and achieve the final verification. Meanwhile, we compare the outputs of the prototype with our expectations to make sure it implements the method correctly and produces the right results by introducing the case into the prototype.

1.4 Research Approach

Eckert & Stacey (2003) advocated "the spiral model of applied research" for a research methodology, which consists of an eight-fold path of research, illustrated in Figure 1.3. The spiral model is a framework useful for outlining different aspects of research. All of the activities in the eight segments generate information and insights that can be used to formulate the requirements and hypotheses that guide the design of research within any other step. A research can start with any segment or *a priori* analysis of the problem at hand, and should carefully select the segments that fit the research context. Considering the research routine of this thesis that is from the theory to methods to implementation, the thesis follows the four segments of the spiral research methodology starting from “development of theory and integrated understanding” to “evaluation of tools” with an additional fold “evidence collection”. Those segments considered in this thesis are slightly adjusted to fit the specific research aims. The main efforts are made in development of theory and development of tools which will be introduced in chapter 3, chapter 4, chapter 5 and chapter 6. The followings describe the segments considered in our research :

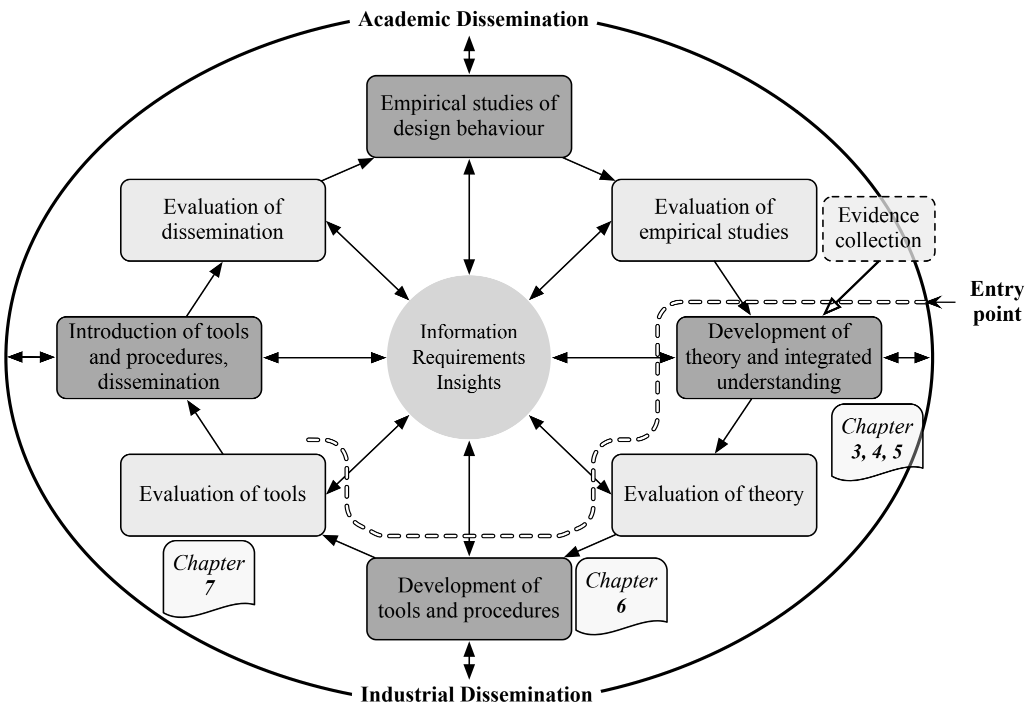

Academic Dissemination
Empirical studies of design behaviour
Evaluation of dissemination
Evaluation of empirical studies
Evidence collection
Entry point
Introduction of tools and procedures, dissemination
Information Requirements Insights
Development of theory and integrated understanding
Chapter 3, 4, 5
Evaluation of tools
Evaluation of theory
Chapter 7
Development of tools and procedures
Chapter 6
Industrial Dissemination

FIGURE 1.3 – The Spiral Research Methodology (adapted from (Eckert & Stacey, 2003))

Evidence collection

The segment “Evidence collection” elaborates the initial understanding of change and change propagation. It includes literature and existing theories exploration employing a range of analysis, as well as related research subjects comparison (see chapter 2). This segment is an additional segment we add to the spiral model to show our priori analysis of the research problems.

Development of theory

The segment “Development of theory” proposes methods to analyze the PD project, builds models to identify change and change propagation, and develops the methodology of simulation change propagations (see chapter 3, 4 and 5), based on the information and analysis from the evidence collection.

Evaluation of theory

In the segment "Evaluation of theory", the developed models and methods are evaluated through applying them into simple cases (see chapter 4 and 5) and the illustrative case (see chapter 7).

Development of tools The segment "Development of tools" makes effort to design and implementation of Agent-based solution to simulate change propagation (see chapter 6). As providing tools for supporting the theories developed in the research is essential, the thesis studies the development environment, finds out the proper software techniques and carefully designs the prototype architecture.

Evaluation of tools

The segment "Evaluation of tools" is performed through an illustrative case (see also chapter 7). The prototype with graphical user interfaces is tested by data from the case and compared with expected results.

1.5 Research Roadmap and Thesis Structure

This Ph.D. thesis is organized as Figure 1.4.

Chapter 1 introduces the research background explaining why to manage change is essential for focal company during the product development process. This chapter points out the motivations of this thesis and briefly discusses the problems in previous existing literature. The contributions of the thesis is also presented in this chapter. Finally, the research methodology is presented.

Chapter 2 discusses the relevant literature from various fields. Firstly, the main concepts and contents of change management are explained, as well as the comparison with risk. Then it introduces the existing PD project modeling framework. it also introduces the agent-based modeling paradigm including the concepts of agents, the application fields and the advantages of this modeling praradigm. Finally, it discusses the main existing computer-aided technique solutions.

Chapter 3 proposes an analysis methodology of change and change propagation in PD project, named Co-evolution Oriented Change Analysis (COCA) framework. Three involved knowledge areas that specified the analysis background are introduced and their interactions are explained. Then, it discusses how to structure the PD project in order to find out the basic elements that are called "constructs" for the analysis methodology.

Chapter 4 focuses on how to identify change occurrence and propagation. It establishes the conceptual models of change and change propagation. With the conceptual models, the definitions and the key factors to identify and analyze changes and change propagations are presented. Then it analyzes the phenomena of change occurrence and

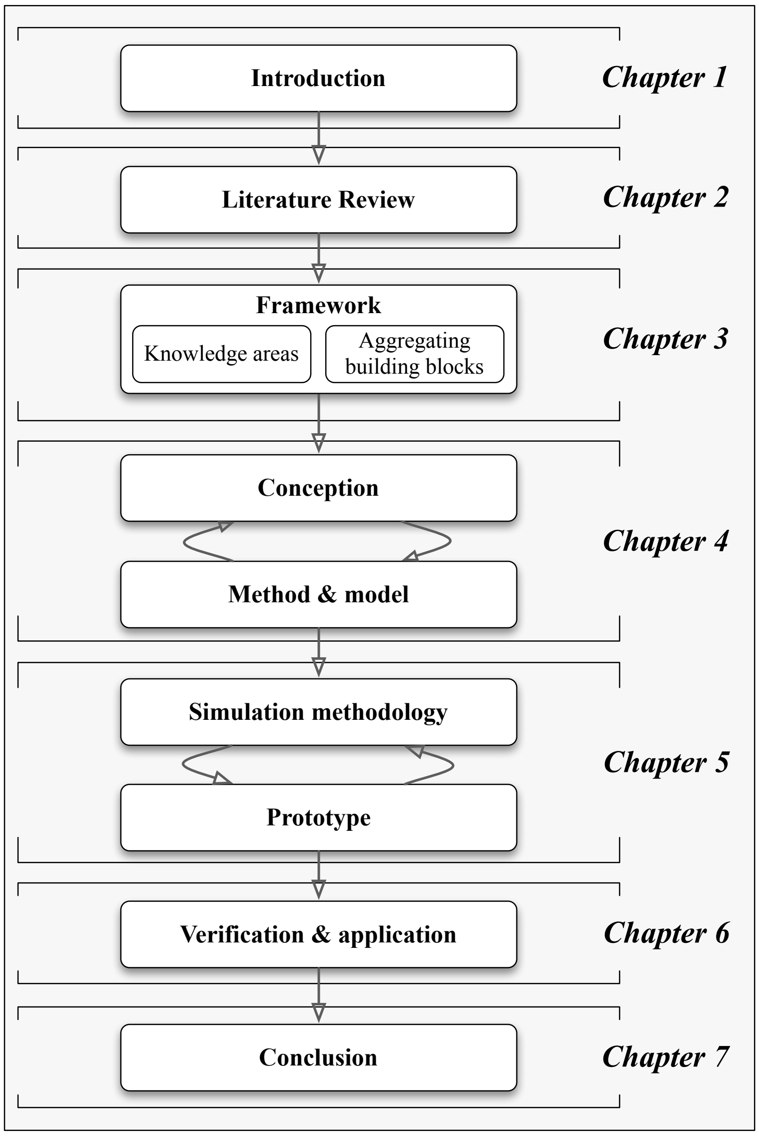


FIGURE 1.4 – Roadmap of the thesis

highlight a set of scenarios of change occurrence with investigating the direct parameters which are led in the alterations.

Chapter 5 proposes a methodology of simulating change propagations and illustrates how to design and develop the agent-based simulation prototype. The prototyped development environment is introduced, named Java Agent Development Framework. Then it explains the basic items of the agent-based simulation system including agent life cycle, agent configuration, the communication between agents, the management of agents and their external interactions. To transform the theoretical methods into the Agent-based implementation, a methodology to design the multi-agent system is proposed, which starts from the analysis of modeling process to the comparison with other methodologies. How to design the multi-agent system for change and change propagation is explained, which discusses the specification, architecture and the algorithms of the systems. At the end of this chapter, the prototype is presented which provides graphical user interfaces.

Chapter 6 suggests an illustrative case to verify this research and presents the application of the methodologies. The simulation tool is applied through this illustrative case.

Chapter 7 concludes this thesis, stating the main contributions. It ends with a discussion of future work that can be based on the contributions made by this thesis.

2 Literature Review

▷ *In this chapter, we mainly identify and summarize the existing literature of change management in Product Development (PD) project. We firstly introduce the objectives of Change Management. Then the literature is sorted out according to a set of dimensions in terms of the characteristics of change management, followed by a detailed discussion on the concepts of change and change propagation. We also introduce a modeling method that provides a general view of PD projects. In addition, the agent-based modeling techniques and computer-aided technical solution are introduced and discussed, which provide the possible techniques for us to implement the suggested methods and models.* ◁

Plan of Chapter

2.1 Objectives of Change Management

In this Ph.D. research, we introduce a set of attributes proposed by Fricke *et al.* (2000) that indicates the tendency of better change management. The attributes are :

- *Less* : to have less changes ;
- *Earlier* : to front-load changes ;
- *Effective* : to select necessary changes more effectively ;
- *Efficient* : to perform the changes efficiently in terms of time, cost, and resources ;
- *Better* : to learn continuously from changes to do it better in the next project.

Then the corresponding strategies of coping with or improving the handling of changes were also proposed within the same article as :

1. Prevention : This strategy aims to reduce and/or eliminate the avoidable changes especially in the later phase of project.

   During a project, some of requirements and specifications are too ambitious, and they would lead to cost and time slips. Therefore, some changes could be prevented by eliminating such unrealistic and/or undesirable requirements and specification. According to this strategy, early freeze is introduced to prevent these kinds of changes (Eger *et al.*, 2005). However, freezing has poor performances in responding dynamic needs. This is specially the case when the initial specifications are moving, unstable or vague (Smith & Reinertsen, 1997). Another way of preventing change is improving the discipline in decision-making (Clark & Fujimoto, 1991) and employing mature an robust technology during project (Clausing, 1994; Schulz *et al.*, 1999). According to the study made by Clark & Fujimoto (1991), up to two-thirds of all technical changes could be prevented by better communication and discipline.

2. Front-loading : This strategy derives from the consideration in "*Rule of Ten*" (Clark & Fujimoto, 1991) and aims to detect emerging changes earlier.

   With this strategy, the overall influence brought in by the detected change and the cost of coping with the change could be limited due to the early detection. Based on this strategy, "Design for X" provides a set of practices of front-loading changes through verifying and validating concepts and designs in the early phases of project. For example, Design for Manufacture (DfM) suggests a way to reduce costs required to manufacture a product and improve the ease with which that product can be made (Bralla, 1999; Korngold & Luscher, 2000). Especially, Fricke *et al.* (2000) also proposed "Design for Changeability (DfC)" as a means to design systems that can be changed without major impacts. Martin & Ishii (2002)proposed "Design for Variety (DfV)" aiming at reducing design effort and decrease time-to-market. In accordance with "Design for Variety", some methods were proposed to identify common and unique elements for designing product family/platform toward maximum performance and/or profit.Then flexible designs are generated resulting in lowering future redesign cost (cf. (Li & Azarm, 2002), (Gonzalez-Zugasti *et al.*, 2000)). In further, De Weck & Suh (2006) proposed a normative flexible product platform design process considering exogenous uncertainties and incorporating the concept of flexible elements which were not mentioned by the former two contributions.

3. Effectiveness : This strategy provides analysis of effective efforts and benefits a change could bring.

   Given that not all the changes are necessary and urgent to be coped with, it is critical for focal company to assess the possible effects of changes and then determine a proper response to the change request. However, according to the research proposed by Fricke *et al.* (2000), the quality of the assessment is closely related to the experience and knowledge owned by the focal company. Eckert *et al.* (2004) proposed a pair of factors to determine whether a change can be absorbed : the initial specification of the part or system, and the margins included in the design of the part or system.

4. Efficiency : This strategy states that the resources (such as time, cost) employed for coping with changes should be used efficiently.

   Riviere *et al.* (2002) proposed a set of indicators based on the experience from the past engineering change management for measuring the performances or variables of a process so as to control their discrepancies with some targeted objectives and to plot their evolutions in time. The indicators concern measuring the effectiveness and the efficiency of engineering change management respectively.

5. Learning : This strategy emphasises the importance of improving the maturity of managing changes through learning from the past change process.

   Lawrence & David (1999) proposed a methodology for management of the change process, especial innovative change process, in terms of group management, strategic planning, empowerment, systems engineering and lifelong learning.Those five factors are considered as key factors in developing a successful management process for Innovative change, which can guide organizations towards achieving performance goals.

Based on the above mentioned attributes, we propose the dimensions used to sort out the reviewed literature. Then we will introduce the highlighted issues in details.

2.2 Dimensions of Reviewing Literature and Reference Tables

In our research, we sort out the literature with a set of dimensions in considering the characteristics of change management (see Table A.1). As focusing on studying change propagations in PD project, we firstly suggest *"Involved knowledge areas"* as one of the dimensions (i.e., the **second** column of Table A.1 that is denoted as "IKA"). Corresponding to our research scope, the involved knowledge areas includes : *project management*, *product management* and *partnership management*.

Given the contributions from the reviewed literature, we then pay attention to the *moment* when those contributions focus during the change management as the second dimension for our review (corresponding to the **third** column of Table A.1 that is denoted as "FM"). Corresponding to the time moment of change occurrence, we classify the focus moment of the contributions as : *"pre-occurrence"*, *"proceeding-occurrence"* and *"post-occurrence"*.

Through studying the contributed results from the reviewed literature, we mainly highlight three aspects of them, i.e., *"contributed research issue"* (corresponding to the **fourth** column of Table A.1), *"key technology"* (corresponding to the **fifth** column of Table A.1) and *"dissemination"* (corresponding to the **sixth** column of Table A.1 that is denoted as "Diss"). The contributed research issues refer to the prescribed change management issues, such as changes handling, change propagation prediction. For each contributed research work, some key technology(s) is/are employed in order to resolve the problem, in which the technology could refer to some framework (e.g., COCA framework in (Zhang *et al.*, 2013)), approach (e.g., Design for Changeability : DfC in (Fricke & Schulz, 2005)), algorithm (e.g., a change propagation algorithm in (Ma *et al.*, 2008)). Among the contributed research result, they are presented and disseminated in various ways, for example academic articles, technical solutions, prototypes.

In appendix A, we will show these contributions in detail. By the end of this section, these contributions are summed up to provide a general view of existing results and possible niches for further research works.

2.3 Analysis of Reviewed Literature

Through our literature review work, we mainly focus on the research contributions involved in product and project management knowledge areas (i.e., 56.5% and 26.2%). Referring to the partnership management knowledge area, we prefer to investigate the changes which are caused by the partnership related issues and then brought in the other two knowledge areas. So we select the ones involved in multiple knowledge areas instead of only the partnership management knowledge area though there is only few ones (i.e., 8.7%). With the literature, we then highlight the moment when the change management solutions are deployed. More than half of the research contributions are focusing on the pre-occurrence moment (i.e., 56.5%), whereas nearly one third of the contributions are focusing on the proceeding-occurrence moment (i.e., 30.4%). Moreover, several contributions covers both the moments (i.e., 6.5%). In the reviewed literature, we do not cover the research contribution that only focuses on the post-occurrence moment. There are only few ones mentioning learning from the occurred changes when coping with them (i.e., 6.5%). Along with the reviewed research contributions, only one quarter of them are implemented by technical solutions (i.e., 26.1%), such as software prototypes (see Figure 2.1).

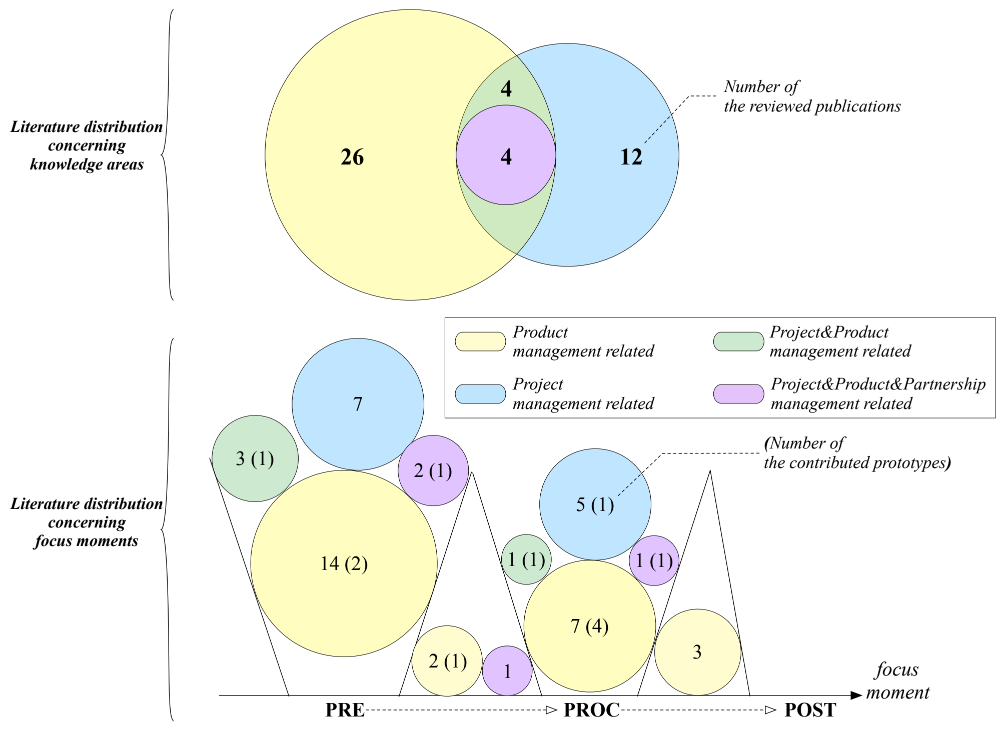

Literature distribution concerning knowledge areas
26
4
4
12
Number of the reviewed publications
Product management related
Project&Product management related
Project management related
Project&Product&Partnership management related
Literature distribution concerning focus moments
7
3 (1)
2 (1)
14 (2)
2 (1)
1
5 (1)
1 (1)
1 (1)
7 (4)
3
(Number of the contributed prototypes)
focus moment
PRE
PROC
POST

FIGURE 2.1 – Global analysis of the reviewed literature

2.3.1 Identified research entries

Given the literature reviewed in Table A.1, we categorize the reviewed literature according to the involved knowledge areas and the focused moment when the research work is introduced. In this way, we are enabled to discover several research entries to change management exposed in the review literature.

As the global literature analysis illustrating in Figure 2.1, more than half of the research works keep focusing on the product management knowledge area, and the results from them are mainly deployed during the pre-occurrence and the proceeding moment. Along with that, a part of research offered the contributions within the project management knowledge area and deployed their work during the pre-occurrence and the proceeding moment. There also exist some other research works being proposed as contribution to the inter-knowledge areas between/among the product, the project and the partner management. Based on the above the scope division onto the gather research works, we discover and identify the following research issues :

1. Handling and controlling changes : the research works are proposed to improve the performance before and/or after accepting changes in consideration of va-

rious constraints (such as time, budget, etc.). The research works, such as (Shiau & Wee, 2008), (Huang *et al.*, 2003), proposed their work to this research issue.

2. Modeling and improving change management process : the research works support to build up the model of change management process in order to deploy change management solution effectively and efficiently. Meanwhile, these research works also facilitate to discover the potential issues for incorporating further improvement and handle the complexity of change management process. For example, (Riviere *et al.*, 2002), (Kocar & Akgunduz, 2010) proposed their considerations in this issue.
3. Identifying and modeling change propagations : the research works focus on proposing the idea/knowledge/expertise in describing and prescribing the characteristics of change propagations. Based on their works, the mechanism of change propagations could be also identified and analyzed. (Eckert *et al.*, 2004), (Fei *et al.*, 2011), (Tang *et al.*, 2010), (Keller *et al.*, 2005), (Hamraz *et al.*, 2012) presented their contribution to this issue.
4. Predicting, analyzing and evaluating change propagations : the research works provide the methodologies of suggesting the qualitative and the quantitative estimations to prescribe the probability of change propagations. Moreover, referring to the perceived change propagations, some research works introduce their solutions to analyze and evaluate change propagation for the further change management work. (Clarkson *et al.*, 2001), (Eckert *et al.*, 2006), (Fricke & Schulz, 2005), (Reddi & Moon, 2009), (Hamraz *et al.*, 2013) proposed the contribution.
5. Designing and developing technical solutions to change management : the research works mainly concrete in design and development computer-aid technical solution to support change management process. The representative works can be found in (Huang *et al.*, 2001), (Huang *et al.*, 2003), (Huang & Mak, 1998), (Eckert *et al.*, 2006).

In the following sections, we present a detailed statement of the highlighted research works involved in our research scope.

2.3.2 Occurrence of change and change propagation

Generally, change is defined as *an act or process through which something becomes different*. In the Ph.D. research, change is further described as an alteration led in an object that causes some outputs shifting or varying from the expectation in which *object* refers to a conceptual term. It can be a thing, being or basic concept that is presented as objective reality. For example, cylinder, piston, activity duration, partner's delivery delay can be considered as objects. In the later chapter, the details of *object* will be presented. Changes could be generated by different sources such as a customer demand, an engineering constraint (maximum thickness of a plate), etc. These changes can propagate from one object to another through the propagation channel (geometric dependency for instance). Finally, during their propagations, changes could be amplified or not. Changing the requirement regarding the thickness of the plate could cause the change of the material to use (switch from Aluminium to Composite) and finally to the change of the supplier. Therefore, when considering the occurrence of changes some relevant concepts should also be introduced. No matter whether changes are expected or not, we prefer to perceive and control them to push the project evolving towards success under the constraints of time, cost and scope.

Referring to the definition of change, most of the reviewed research works prefer to quote the content from standards (e.g. US Military Standard 480B (Department of Defense, 1988) and previous research (e.g. (Ibrahim, 2003; Riviere *et al.*, 2002)). They are used to emphasize the distinct scenarios where changes are considered. According to the field or the background where changes occur, there are more specified definitions about change such as *design change* and *engineering change*.

Eckert *et al.* (2005) indicated design change can be interpreted as the alterations to design process for modifying to an existing design or recognising shortcomings in a partially completed design. Lying at the heart of almost all the design processes, change is identified as the improvement or even the innovation of design processes. Furthermore, Eckert *et al.* (2004) presented the characteristics of change, and analyzed the sources, the causes and the predictability of change. Two different sources of change are differentiated as *emergent change* caused by the state of the design and *initiated change* arising from an outside sources. Then through the interview to an industrial company, the causes for change are summarized into different representation, insufficient communication, no decisions or wrong decisions, insufficient clarification of the task and inadequate processes.

Referring to engineering changes, they are highlighted as the alterations occurring after an initial engineering decision has been made, which play an important role in product development and contribute to improving products. Rouibah & Caskey (2003) specified three kinds of engineering changes based on the design process : (1) engineering changes during initial design, (2) engineering changes after initial design, and (3) the ones during major reconstruction of product. The authors also defined the strategies used to minimise the impact of changes. While Riviere *et al.* (2002) discuss engineering change from a business perspective instead of addressing a particular phase of product life cycle. So the causes of engineering change in the automotive and aeronautic industries are considered and summarized as changes in needs and requirements, program or project interactions, need to fix deficiencies, technological changes, legislation changes and changes in project scheduling. Then authors attempted to qualify the consequences of an engineering change. They specified the consequences into a series of impacts : near term cost impacts, impact on program schedule, impacts on product performances with respect to expectations, primary impacts on particular stakeholders (suppliers, sub-contractors and work partners), impacts on other programs or projects, additional changes resulting from the same issue and life cycle phases impacts. To decide whether an engineering change should be touched or studied further, Riviere *et al.* (2002) quoted the concept of interchangeability, which is defined as the ability of an entity to be used in place of another to fulfil the same requirements without modification. Correspondingly, engineering changes are characterized into modification, amendment and correction.

Based on the understanding of change, change propagation means that *a change causes another one*. In fact, a change rarely occurs alone and multiple changes can have interacting effects with each other in the real business context. Eckert *et al.* (2004) suggested that the network of changes with various connections is a more practical and veracious representation of change propagation. They indicated that individual change chains is not sufficient to illustrate the practical multiple changes having interacting effects on other (sub-)systems, therefore the concept of complex change networks was introduced to display the connections between (sub-)systems or parts.

In (Rouibah & Caskey, 2003), authors proposed a parameter-based methodology to analyse change propagation in a collaborative scenario. A list of changes with short descriptions, is prepared. Afterwards the relevant persons (or roles) are authorized to maintain the content of the list. During the evolution of the parameters, the co-ordinators and collaborators would discuss the change list to specify the states of the parameters when arriving an agreement on the changes, and the states of parameters will be set as either "in change" or "revised" consequently. According to the product structure, the parameters linked to elements of product as well as to persons can compose a network, which can also inform the relevant persons (or roles) whenever the status of the value of a particular parameter changes. With the change approved on a parameter, a serial of adjacent parameters might also be affected by the change, through which the parameters subjected to change can be identified. In this way, the possible change propagation between the changed and the neighbouring parameters in the parameter network can be discovered and notified to the co-ordinators and collaborators. During the propagation, the parameter list is maintained and updated with the latest status of the changes. Depending on the parameter network, the change propagation can be tracked to its conclusion.

Distinguishing change from risk

During investigating changes in PD project, it becomes necessary to distinguish risk management and change management because they are often interrelated with close implications, and have some shared knowledge areas.

In general, risk is perceived as a negative concept, and it is often defined as a measure of the probability and consequence of not achieving a defined project goal in project management (Kerzner, 2009). Thus, risks could be described as some unfavorble events that is probable to occur in the future. Referring to the future events that are favorable, they are usually described as opportunities. Compared with risks, changes imply a wider scope of describing events that might bring either positive or negative consequence.

Even if risk management and change management could share and do share some common concepts and methods, a risk refers mainly to an uncertain event that could have an impact on the project objectives such as the deviation of a function or a service. The risk of a valve failure, the risk of an empty tank, etc. are some of the examples of risk management issues. Risks are characterised by their gravity and probability. However, the change is dealing with a kind of "what-if" scenario in which it is necessary to think of the consequences on the PD project. For example, if a new customer requirement comes, what will happen on the functions or structure of a product. Once change happens, the main issue is not only to assess the consequences (in terms of likelihood and probability) but also to understand how the consequences can be propagated, whether the change can be coped with locally or should be transferred. Therefore, the main goal of change management is to ensure the system resilience and robustness (for future generation of products for instance) while in risk management, the analysts look for assessing the potential impact of a failure in a component on the target service.

2.3.3 Change propagation analysis and prediction

Change propagation is one of the most important issues in change management, because it reflects the phenomenon that other changes could be invoked from the initial change and that the change could be passed from one knowledge area to another. Therefore, in this subsection, we will explore change propagation deeper in terms of its analysis and prediction.

Chu & Trethewey (1998) proposed a dynamic structural design model for evaluation based on Finite element analysis (FEA), which can be developed in a rapid fashion. The model examines the effects of potential design changes, but it sacrifices a certain level of detail in order to obtain rapid development time. In (Williams, 2000), the author demonstrated the project dynamics, modeled the inter-relationships between factors to quantify their combined effect based on System Dynamics. In the contributed work, a conceptual framework is just only provided but no real calculation formulas or methods of change are presented. Majeske *et al.* (1997) suggested a method for evaluating product and process design changes by analyzing hazard plots of the random variable time to first warranty claim. They mentioned that “this allows focusing on product quality at time of sale by eliminating any possible changes due to replaced or repaired components”. In the paper proposed by Rutka *et al.* (2006), a method for change propagation analysis based on a dependency model considering 3 main aspects is suggested : information that represents several viewpoints or domains of the engineering system, dependency information that describes the links or relationships between two items, the evolvement of overall design representation and the corresponding engineering organisation along the product lifecycle.

Eckert *et al.* (2005) discussed the problems involved in predicting how changes propagate. They introduced two different change prediction strategies that originates from (Jarratt & Clarkson, 2005) : Depth-first search and Experience based heuristic search. Clarkson *et al.* (2001) proposed a change prediction method based on Design Structure Matrix (DSM) that is composed with aggregated risk value according to treating impact and likelihood. The method consists of three steps : creating the product model ; completing the dependency matrices and computing the predictive matrices. In (Keller *et al.*, 2007), Keller et al. proposed two combined strategies for predicating changes in complex products design, taking use of the Change Prediction Method (CPM) and the Contact & Channel Model (C&CM). CPM is a technique for predicting change propagation risks based on product connectivity models (Clarkson *et al.*, 2004) while C&CM is a design model describing products on different levels of abstraction regarding functions as well as shapes and connecting these levels of abstraction (Albers *et al.*, 2003). The two strategies proposed by René et al. are "Considering the Functional Information" (strategy 1) and "Translation of C&CM Models into CPM models" (strategy 2). Strategy 1 starts from building CPM model to identifying high-risk connections and finally to building C&CM model of high-risk connections. Strategy 2 firstly builds C&CM model, then translate C&CM model to CPM model and finally use CPM algorithms to identify high-risk connections. According to (Fricke & Schulz, 2005), "a reduction of unnecessary specifications and requirements also leads to a reduction of changes".

Lawrence & David (1999) proposed a methodology for management of the change process, especial innovative change process, in terms of group management, strategic planning, empowerment, systems engineering and lifelong learning. Those five factors are considered as key factors in developing a successful management process for Innovative change, which can guide organizations towards achieving performance goals. Contributing to reduce lead time, Christian *et al.* (**?**) outlined a process-based view of Engineering Change Order (ECO) management. Five contributors to long ECO lead times are identified. In this paper, the authors classified the strategies a development organization can adopt to reduce the negative consequences of ECOs into four groups in the form of "Four Principles of ECO Management". Ibbs *et al.* (2001) introduced a systematic approach for project change management, which is founded on five principles : (1) promote a balanced change culture ; (2) recognize change ; (3) evaluate change ; (4) implement change ; and (5) continuously improve from lessons learned. However, the detailed methods such as how to recognize change or evaluate change were not provided in their work.

As an executive summary about change and change propagation, we can learn the followings from these past sections :

1. Change management issues are regarded from several points of view : less, earlier, effective, efficient and better ;
2. Changes of various kinds are considered in a huge accumulated literature ;
3. Methods and tools are available to model and analyze changes ;

4. Changes are propagated and there are propagation channels discussed in several papers, for example, (Clarkson *et al.*, 2001; Eckert *et al.*, 2004, 2005; Jarratt & Clarkson, 2005).

However, there are still some shortcomings that our research aims to address :

1. As far as we know, a formal model of change and change propagation is not provided ;
2. Changes are often considered and contained into one knowledge area, and the propagations channels are often intra-domain ;
3. There are few computerized prototype or software provided to simulate change and change propagation ;
4. There are very few literature that mention how to translate their methods or models to computerized prototype or software.

2.3.4 Implemented technical solution

The research from IT provides strong supports in improving performance of methodologies, resolving conflicts in executing process, treating complexity, visualising data and integrating solutions. In this section, we introduce the possible aspects that the computer-aided technical solutions can help during the change management.

Assistant treatment

Keller *et al.* (2007) proposed a set of visualisation methodologies for representing change propagation data to assist designers to predict change in product design process, in which the graphical interface displayed two kinds of networks of component connectivity. They were illustrating the propagation path and the risk path respectively, which are invoked from one component.

Enhancement function

Referring to the collaborative management environment, Huang *et al.* (2000) proposed a web-based system developed for managing engineering changes, which reduces the paper-based data exchange to a minimum level, and improve the effectiveness and efficiency of communication and co-operations through allowing simultaneous access.

Solutions integration

A virtual environment named ADVICE integrating the approach for processing the changes within a virtual collaborative design environment was proposed by Kocar & Akgunduz (2010) to provide users with iterative support for predicting engineering changes. Besides the methodology to model the engineering change management process, a compact solution to combine parametric and graphical information was proposed integrateing the functionalities of maintaining database, tracking engineering change data and providing real-time manipulation of the shared three-dimension product structure.

A commercial software suite, CATIA, developed by the French company Dassault Systemes also provides a software solution “CATIA Program Change Control” for companies to define, plan, track and implement complex product changes [1]. With CATIA Program Change Control, the total impact, scope and cost of a proposed change to the overall product development can be captured by establishing a change project plan. It also provides complete visibility of all changes occurring throughout the product development process.

2.3.5 Research gap to be filled

From the reviewed literature, we are enabled to identify the research gap and approve the research idea.

1. Only one of the knowledge areas (i.e., product, project, partnership management knowledge areas) is highlighted in most of the research contributions (i.e., 82.6%), and the change management research concerning multiple knowledge areas is still in its infancy.
2. Most of the research contributions do not provide the softwares or prototypes that implement their methods or models (i.e., 73.9%).
3. There seldom exists some simulation methodology to support analyzing the mechanism of change propagation in order to enhance the robustness of PD project when facing an introduced change.
4. Multi-agent based technique has not been mentioned nor used though it has quite good performance in decision making support.

Meanwhile, we discover that more research contributions focus on the product management knowledge area. This inspires us to take this knowledge area as the main clue to analyze the scope of change and change propagation, and then extend to the other two knowledge areas.

Based on the above understanding, our research work will make contribution to fill the research gap as :

1. Propose a framework of modeling and analyzing PD project by considering its product, project and partnership management knowledge areas simultaneously.
2. Create a methodology of simulating change propagations in PD project within the change propagation network built up by an innovative model approach.
3. Incorporating multi-agent based technique in designing and developing the technical solution in our research.

1. http ://www.3ds.com/fr/products/catia/portfolio/catia-version-6/overview/

2.4 PD Project Modeling

Intuitively, one can understand that product development is firmly connected and coupled with partners. As we want to show these links and use them as potential change propagation channels, we focused on the methods and frameworks that consider these couplings and look for modelling them. In this section we go through some of the previous works performed with this target in mind.

Early Supplier Involvement (ESI) is the first research work in this area. ESI means collaborating with the supplier in the design phase of the product development process to create better designed components (Hölttä *et al.*, 2009). It is a tactic used to assist the company in design of products and to obtain more value from the supplier base. The benefits of ESI are to reduce costs, improve quality, faster development, and shorter time-to-market (Clark & Fujimoto, 1991; Dowlatshahi, 1997; Morgan & Liker, 2006). Hartley *et al.* (1997) proposed a model of suppliers' involvement in product development (see Figure 2.2) and emphasized that early supplier involvement is greatly related to success of new product development process in his model.

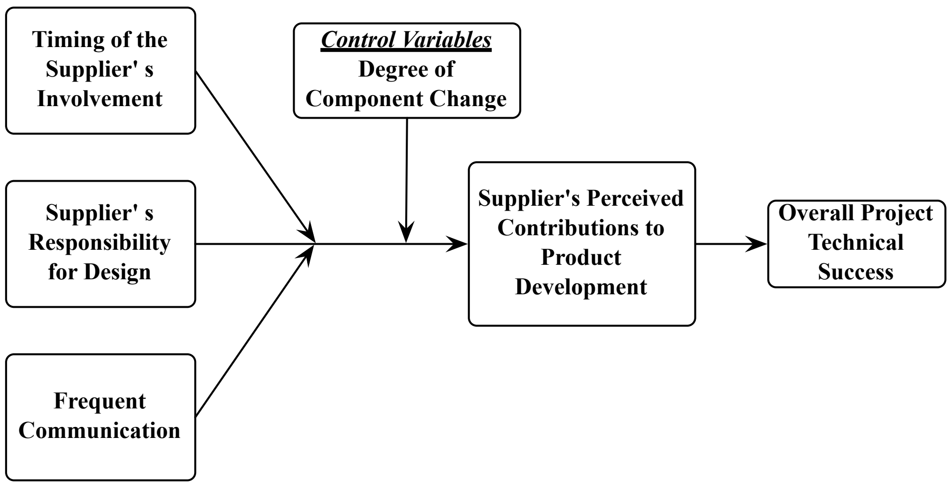

Timing of the Supplier' s Involvement
Control Variables
Degree of Component Change
Supplier' s Responsibility for Design
Supplier's Perceived Contributions to Product Development
Overall Project Technical Success
Frequent Communication

FIGURE 2.2 – Research model of suppliers' involvement in product development (Hartley *et al.*, 1997)

Zolghadri *et al.* (2009) proposed a framework named Co-Engineering of Product design and Supply Chain (hereafter CEPS) considering simultaneously the product design/development process and the associated network of partners along the process of project, and meanwhile with the evolvement of project the systematic influence of decisions made in product development on the supply chain design/development is underlined and vice versa. In CEPS, both end products and enabling products are considered based on the observation that the product design and supply chain management are rarely regarded together. To introduce a final product into market, the process according to CEPS is illustrated as the product in-house life cycle of four main phases : development, production and sale, usage and recycling. Furthermore, the framework highlights that not all partners have the same influence on the product. Supply chain partners can be classified into four categories according to their activities during the product life-cycle : risk and revenue sharing partners, design partners, manufacturing partners and Standard part sellers.

This CEPS framework provides the clues to analyze the mutual relations between product, partners and PD project. The product can be examined from three dimensions : structure, linkages of constituent components, and the requirements and specifications.The network of partners can also be viewed from three aspects : the structure of the network, the dependencies from demand-supply, and management of the network including coordinating resources and maintaining relationship. These three aspects of products and networks create couplings between product and partners. For instance, the product structure could determine the dependency of the contributions of the partners. Meanwhile, the product life-cycle and development processes are closely related to the PD project phases. The PD project management activities monitor and control the product development. On the other hand, the involvement and the roles of partners are organized through project management but also set up the management activities. In this case, by analyzing the couplings between product, partners and PD project, the relation network between these there domains can be established, and as a result, the propagation routine can be derived.

In general, to analyze change and change propagation in PD project, it is necessary to understand the relations between PD project, product and partners. Further, based on this idea, we can deduce the relations between the three knowledge areas (i.e., project management knowledge area, product management knowledge area and partnership management knowledge area).

2.5 Agent-based Modeling

One of the main target of the Ph.D. research is to be able to simulate the change propagation within a network of interconnected objects. Later in chapter 3, the reader will see that these objects or nodes are clearly identified by components, partners and activities. One of the most promising techniques that could allow us to simulate the change propagation is the Agent-Based Modelling (ABM). In fact, if a node is considered as an agent, once receiving an alteration, it will be justified whether a change occurs or not (i.e., not all alternations are change) on this node. Further, once the node is impacted by a change, it would be interesting to show how these changes are coped with internally or transferred to the neighbour nodes. This means that each agent should have enough "intelligence" to be able to assess if a change occurs and if it can be managed internally or not. This intelligence refers to methods that we put inside each agent. These points will be clarified in chapter 5 and 6 when the Agent-based models are proposed. Considering this potential powerful way of modeling and simulating change propagation, we will go through some of the main issues in the Agent-based modelling. This section does not have any pretension to be complete. The main idea is to provide a kind of sufficient-enough package of knowledge required for understanding the further development of the Ph.D.

Agent-based simulation is a modeling paradigm that abstracts and models a complex system comprised of interacting autonomous agents (Kuhn Jr *et al.*, 2010). It also relates to the term "agent-based modeling", "agent-based modeling and simulation", or "multi-agent simulation". This modeling paradigm can simulate the simultaneous operations and interactions of multiple agents, in an attempt to recreate and predict the occurrence of complex phenomena. Generally speaking, an agent-based simulation model, at least, consists of numerous agents and the relationships between them. In the model, an agent is an autonomous entity, capable of independent actions in the environments that are typically dynamic and unpredictable (Jennings & Sycara, 1998; Kuhn Jr *et al.*, 2001). It fulfills a specific role in order to achieve particular objectives and solve problems. Each agent has a knowledge base that contains the essential data and knowledge required by the agent to perform its planning activities (Huang *et al.*, 2000). The actions of an agent can be specified into two types (Bauer *et al.*, 2001) : pro-active and re-active. Pro-active actions are triggered by the agent itself, i.e., it is tested on state changes of the agent if the pre-condition of the action evaluates to true. Re-active actions are triggered by another agent, i.e., receiving some message from another agent. However, agents do not only act in isolation but in cooperation with other agents. The communication between two neighbours agents is to transform a behaviour of one neighbouring agent to another.

Agent-based simulation method can be applied to many fields, see table 2.1 (adapt from (Macal & North, 2001)). The benefits of using agent-based simulation method over other modeling techniques can be concluded in three aspects (Bonabeau, 2002) :

– *The capture of emergent phenomena*. Emergent phenomena result from the interactions of individual entities. In the agent-based simulation model, emergent phenomena is generated from a bottom-up way. In this case, one can model and simulate the behavior of the agents and their interactions, capturing the emergence from bottom up when the simulation is run.
– *A natural description of a system*. Because of the characteristics of agent-based simulation method, it is quite natural to describe and simulate the behavior of the components in a system.
– *Flexibility*. It is easy to add, delete and modify an agent.

TABLE 2.1 – Agent-based Simulation Applications (Macal & North, 2001)

Subjects	*Application Field*
Business and Organizations	Supply chains Manufacturing Operations Consumer markets Insurance industry
Economics	Artificial financial markets Trade networks
Infrastructure	Electric power markets Transportation Hydrogen infrastructure
Crowds	Pedestrian movement Evacuation modeling
Society and Culture	Ancient civilizations Civil disobedience Social determinants of terrorism Organizational networks Military Command and control Force-on-force
Biology	Population dynamics Ecological networks Animal group behavior Cell behavior and sub cellular processes

2.6 Chapter Summary

This chapter presented a literature review on change and change propagation.

We firstly sort out the literature with a set of dimensions in considering the characteristics of change management and also introduced basic concepts. Change and risk are often apt to be confused, thus, the differences between these two terms was explained. Based on the analysis of the literature on change management, we found that few research cover the three knowledge areas (project management, product management and partnership management) at the same time. Meanwhile, there are some tools or prototypes for change management in terms of data analysis, predication and impact calculation, but few research provided simulation solutions for change management. Therefore, our research aims to fill the gap on the existing literature, i.e., to provide a simulation solution and prototype on change and change propagation considering the three knowledge areas simultaneously.

Then in this chapter we introduced the Co-Evolution of Products and Partner Network in Project that provided the basic framework for understanding product, partners and project. After, we catagorized the usage of computer-aided technical solution of change management. The last part of this chapter presented the basic theories and techniques for methodologies development and prototype design and implementation. The agent-based simulation method was the basis for our research to develop the agent-based system and prototype for change and change propagation. In the agent-based system, each individual agent is responsible for different behaviors such as determining whether a change occurs, whether a change will be propagated and where to be propagated. Those behaviors on individual agent result in collective behaviors that present all the possible change propagation channels and impacted nodes. In the next chapter, we will introduce how we understand and connect the three knowledge areas as well as how to identify basic elements for constructing the simulation network.

3 Co-evolution Oriented Change Analysis Methodology

▷

In this chapter, we will introduce our Co-evolution Oriented Change Analysis (COCA) framework that indicates this Ph.D. research scope. The framework provides a simultaneous modeling consideration in product management, project management, and partnership management knowledge areas as well as the interrelations between them. Under the framework, we propose hierarchical systems models to analyse a Product Development project in its different knowledge areas and at different granularity levels. In the framework, we also propose a product evolution model facilitating to recognize how the information/data is aggregated to reflect the efforts contributed for obtaining the final product design solution along the project progress. This enable us to identify the interactions between the three knowledge areas and further to identify the change propagation channels.

◁

Plan of Chapter

3.1 Key Concepts of Research Work

There is a number of key concepts mentioned and used in our research work. Therefore, before presenting our work, we use a figure to state all the key concepts and the relationships between them (see Figure 3.1). From the figure, the readers could also discover and get to know the research process that is started from analyzing a PD project to specify our model of change propagation for the simulation. Moreover, each of the key concepts can be looked up from Annex C for detailed explanation.

3.2 Co-evolution Oriented Change Analysis (COCA) Framework

According to (Browning *et al.*, 2006), Product Development (PD) project is defined as "*an endeavor comprised of the myriad, multi-functional activities done between defining a technology or market opportunity and starting production*". During the endeavor, the concerned partners are engaged to create the final product, under various constraints. A PD project is different from the conventional manufacturing project. Compared with the latter, the former is with higher ambiguity, uncertainty and risk (Pich *et al.*, 2002) since it is to produce a novel product or with some innovative parts.

To manage a PD project efficiently, it is important to manage changes efficiently. In last section it has been already noticed that : firstly, during the life-cycle of product, changes can occur at any time no matter that they are foreseen or unforeseen. Secondly, a change in a system would cause another change(s) occurring. Let us consider the example of engine (firstly mentioned in the end of section 1.1). If the diameter of the piston is changed, then the internal diameter of the cylinder has also to be changed. The impact of the original change could either be amplified or reduced or kept unchanged through being transferred to somewhere. Thirdly, change propagation involves multiple knowledge areas of project, i.e., the product and constituent components, project activities, and involved partners. For instance, a modification to the design parameter of a product component would force the concerned supplier to re-produce the components, and then the project could be delayed because of the extra spent time in the re-production. Thus, changes and change propagations are expected to be perceived, analyzed and coped with or even controlled as soon as possible.

In consideration of the above issues, we propose a Co-evolution Oriented Change Analysis (COCA) framework (see Figure 3.2) that simultaneously models (1) product management, (2) project management, (3) partnership management knowledge areas of a PD project and the interrelations between these areas. The COCA framework depicts our research scope and implies the approach of modeling a PD project in aim of identifying change and change propagation.

The COCA framework enables to :

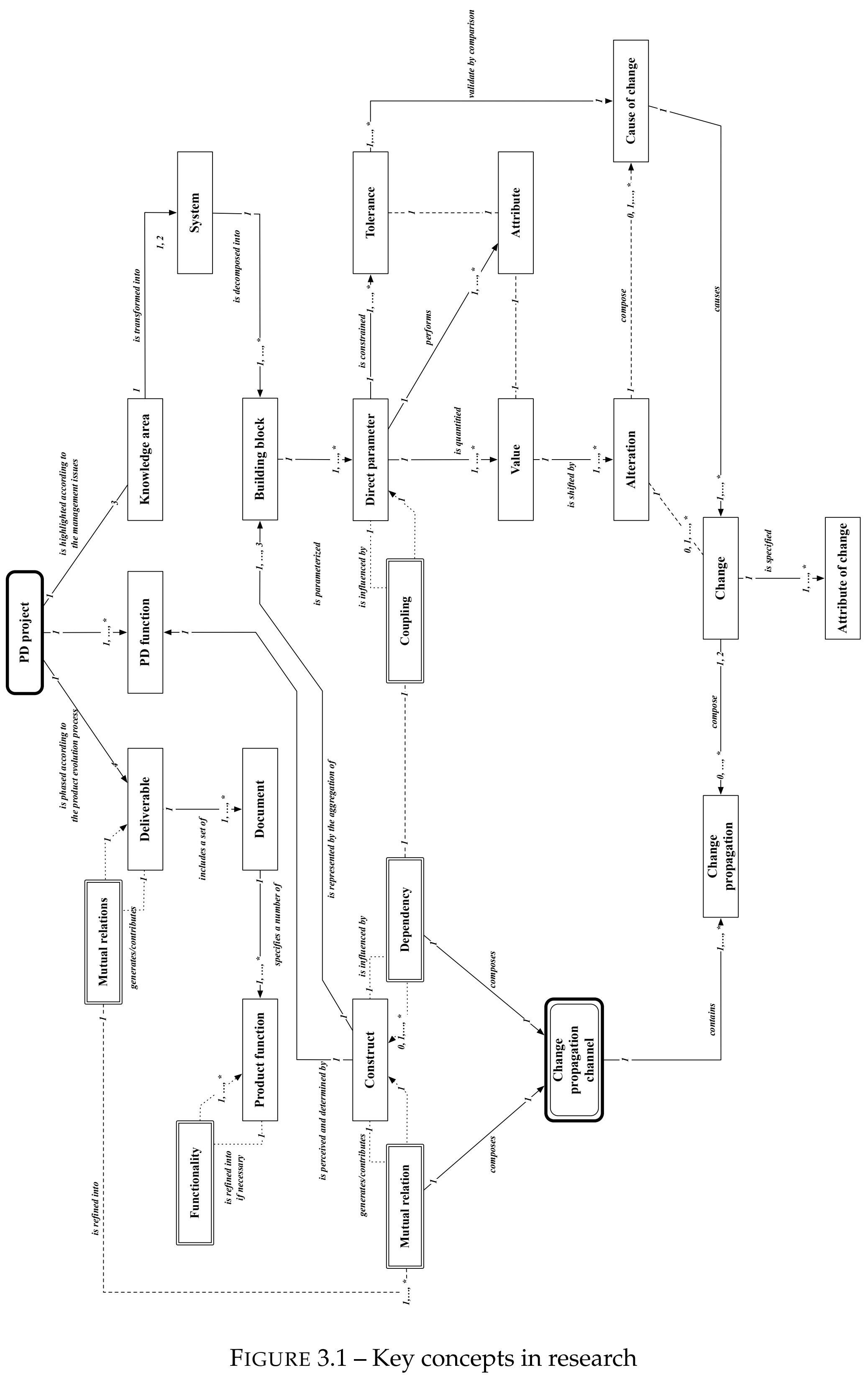


FIGURE 3.1 – Key concepts in research

– Model product design and development process by indicating how the product evolves considering those three knowledge areas simultaneously ;
– Acquire the knowledge/information/data derived from the multiple knowledge areas. The knowledge/information/data can be aggregated to reflect the product functions ;
– Identify the dependencies between the knowledge/information/data belonging to the same/different knowledge area(s) ;
– Identify the potential change propagation channels, and analyze the mechanism of change propagations.

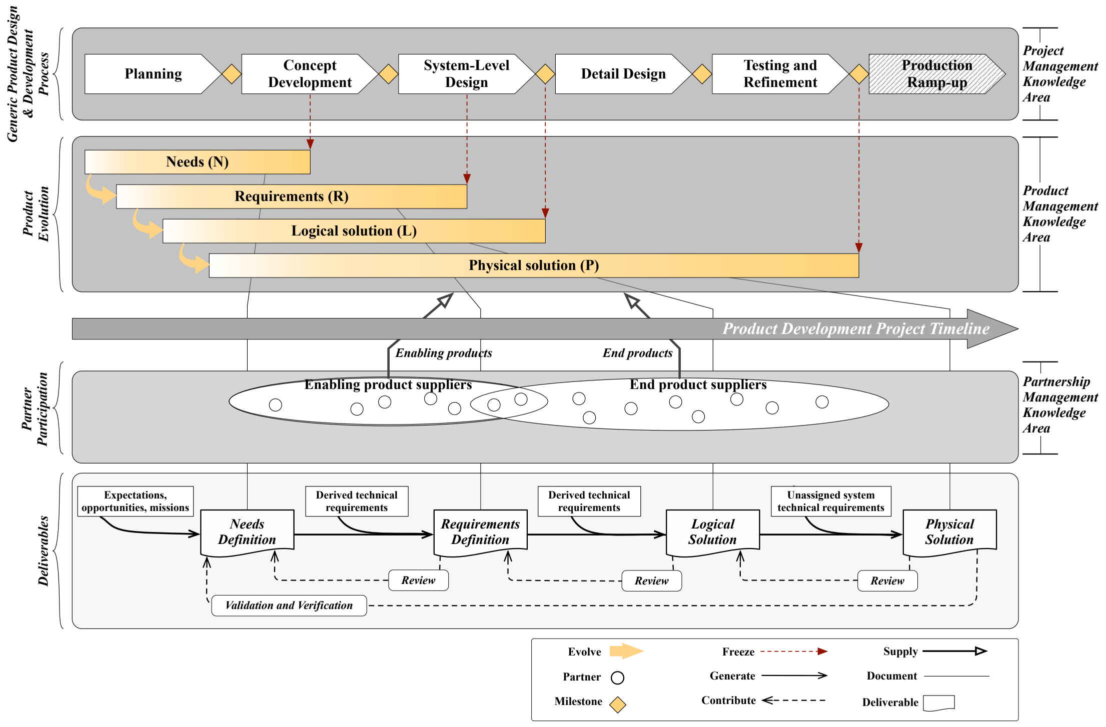


FIGURE 3.2 – Co-evolution Oriented Change Analysis (COCA) framework

3.3 Highlighted Knowledge Areas of COCA Framework

In the COCA framework, we observe and analyze a PD project from three aspects. First, we adopt the six-phase generic product design and development process (we also use “*the generic process*” for short in the following context) proposed by Ulrich & Eppinger (2007) as the starting point to model the evolution of a PD project. Along the phases, a serial of milestones are assigned between each two successive phases to indicate that the upstream phase should have been completed and the downstream phase could be invoked. In this way, the generic process presents a "stage-gate" model and assembles our consideration in project management issue. Then we propose a product evolution model reflecting the evolving *states* of product along the PD project. The states capture a set of **K**nowledge-**I**nformation-**D**ata, *KID* in short, that define the product model at a given level of maturity. The product evolution model presents our understanding in how the KID is aggregated to reflect the efforts of generating the product design solution. Finally, we identify the partners from supply chains who offer their efforts during the product evolution and categorize them according to their contributions.

Through considering the multiple aspects of the PD project simultaneously, we highlight three knowledge areas in the COCA framework (also see Figure 3.2) :

1. Project management : this knowledge area manages product design and development activities with a set of milestones and makes sure they are under the time, quality and cost constraints ;
2. Product management : this knowledge area covers the process during which product model evolves from customer needs to design solution, and then the product solution is released to achieve the expected performance and expected needs ;
3. Partnership management : this knowledge area considers the activities and roles of the partners participating in the PD project along the evolution of the product.

3.3.1 Project management knowledge area

As proposed by Ulrich et al. (2007), the generic product design and development process consists of six phases between which there are milestones indicating a phase has completed and another phase is upcoming :

1. Planning : this phase is started with studying the objectives from the market analysis and preparing the answer to the requests from some customers. Through this phase, the whole project is approved to launch.
2. Concept development : during this phase, the needs from market and/or particular customized buyers are identified. Some alternative product concepts are generated and evaluated, and some (or one) of them will be selected for further development. Meanwhile, the industrial design concepts are developed as well as the production feasibility is assessed, i.e., the product manufacturing begin to be considered.
3. System level design : in this phase, the product architecture is defined and the modules/subsystems composing the product are also recognized. The functional specification of each module/subsystem is generated. The supply chain development enables to identify the critical suppliers and perform make-buy analysis.
4. Detail design : in this phase, the specification of all the components are composed concerning the parameters of geometric, materials, tolerances, etc. With the speci-

fication, the standard components to be purchased from suppliers are identified, and the process plan for further assembly is released.

5. Test and refinement : in this phase, the multiple pre-production versions of product are assembled and evaluated.
6. Production ramp-up : in this phase, the products are made in aim of training work force and working out any production process problem. The produced products are evaluated to identify any remaining issue. The production transits gradually from product ramp-up to ongoing production during this phase, during which the product is formally launched.

The generic product design and development process stresses an endeavor from the viewpoint of the project evolution, during which the six phases imply a serial of critical activities that are executed to achieve the prescribed milestones. So the generic process is modeled as the composition of stages and gates. Each stage (i.e., a phase) is a set of activities performed either in sequential or parallel according to the project schedule, while each gate (i.e. a milestone) indicates a decision point at which the previous stage is reviewed to determine whether the stage should be completed through validation and verification as well as the following stage is invoked. In our research, "activities" are regarded as the higher level of work and they could be performed in various scheduling ways. In consideration of the complexity of activities, we use "tasks" to describe the assignments which are assigned to some executors to do during the scheduled span time. In this way, an activity is treated as a set of tasks.

Based on the above consideration, we propose a hierarchical model of project corresponding to the project management knowledge area. According to the hierarchical model, a project is modeled as a set of phases, and each phase is modeled as a set of tasks (see Figure 3.3).

During the above modeling procedures, the most elementary objects (i.e., the tasks) are referred to as end elements. End elements are gathered or assembled to together in order to form more complex objects called *building blocks*. The composition of these building blocks can be performed iteratively. To be noticed, the building blocks obtained during our analyzing the project management knowledge area are named as "*project building block*", and they are also called "building blocks" in this section. In the further sections, we will also introduce other types of building blocks, such as product building blocks when analyzing the product management knowledge area. In this Ph.D. research, end elements are considered to be the smallest bricks in knowledge areas ; they are of minimum granularity.

As Figure 3.3 illustrated, the hierarchical system model of project implies three levels of systems and they are :

– Project : it is the endeavor during which the product is designed and developed with the constraints from time, cost, quality aspects. It is divided into several phases.

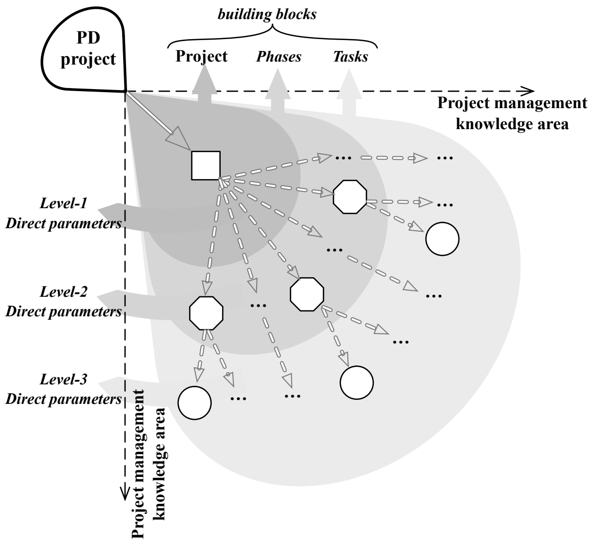

building blocks
PD project
Project
Phases
Tasks
Project management knowledge area
Level-1 Direct parameters
Level-2 Direct parameters
Level-3 Direct parameters
Project management knowledge area

FIGURE 3.3 – Hierarchical model of systems in project

– Phase : it is a distinct period in the project process and contains a group of tasks to be executed. At the end of a phase, one or more milestones are predefined to validate and verify the phase.
– Task : it is the work arranged according to project definition (such as Work Breakdown Structure) and describes what needs to be completed by the participants in the project. Each task has time, cost and quality constraints.

We identify a number of direct parameters characterizing the building blocks. Direct parameter is a concept named by Andreasen & Hein (1987) and initially used to define and express a product. In our research, we extend the definition of direct parameter (based on the definitions proposed in (Andreasen & Hein, 1987)) as *a measure item determining the response, the characteristics and/or the behaviour of a system obtained through decomposing the knowledge areas*, seen Table 3.1, Table 3.2 and Table 3.3 for the concreted examples. To be noticed, the direct parameters are regarded as the entry through which alterations are introduced into the PD project. In chapter 4, we will explain it in details. Corresponding to the hierarchy of the above building blocks, we categorize the direct parameters characterizing the building blocks into three groups : project, phase, task.

Each direct parameter is formulated by a pair of data items that are *attribute name* and *value*. The former indicates the brief description of a feature regarded as the property of a system (i.e., the involved building block), whereas the latter indicates the value belonging to a prescribed *domain* (such as integer, boolean, etc.) as the measurement and/or the presentation of the property.

In the COCA framework, we mainly analyze a project through investigating the constraints concerning *time*, *cost* and *quality/scope* (Kerzner, 2009). These constraints are inherited through identifying the direct parameters in the multiple levels and specified in various forms corresponding to the granularity of the building blocks. In Table 3.1, we present an illustrative list of project direct parameters characterizing a project. In the further illustrative case, we will extend this direct parameter list and then illustrate the procedure of how to identify and collect parameters.

TABLE 3.1 – Direct parameters of project building block

Property	*Attribute name*	*Description*
Time	Start date	The start date
	End date	The end date
	Duration	The duration implying margin time
	Completed rate	The general progress of project
Cost	Budget	The amount of money spent during the project
	Resource	The list of required human, software, hardware resources
Quality	Final product	The index of product specification
	Supply chain	The list of potential suppliers, customers
	Department	The list of internal participants
	Document	The list of project documents

Table 3.2 presents some examples of the direct parameters associated with phases.

TABLE 3.2 – Direct parameters of phase building block

Property	*Attribute name*	*Description*
Time	Planned start date	The planned start date of phase
	Deadline	The planned end date of phase
	Time limit	The planned duration of phase that implies margin time
	Actual start date	The actual start date of phase
	Actual end date	The actual end date of phase
	Duration	The actual duration spent by the phase
	Upstream phase	The index of upstream phase
	Downstream phase	The index of upstream phase
	Completed rate	The progress of the current phase
	Iteration	Whether the current phase is executed iteratively
Cost	Hardware	The list of hardware resource
	Software	The list of software resource
Quality	Supply chain	The list of concerned suppliers, customers
	Team	The list of internal participants
	Milestone	The index of decision point specifications
	Achivement	The index of the result obtained through the current phase
	Document	The list of document involved in the current phase

The direct parameters characterizing the task are presented in Table 3.3.

TABLE 3.3 – Level-3 direct parameters of task building block

Property	*Attribute name*	*Description*
Time	Planned start date	The planned start date of task
	Deadline	The planned end date of task
	Time limit	The planned duration of task that implies margin time
	Actual start date	The actual start date of task
	Actual end date	The actual end date of task
	Duration	The actual duration spent by the task
	Upstream task	The index of upstream task
	Link to upstream task	The relations with the upstream task
	Downstream task	The index of upstream task
	Link to downstream task	The relations with the downstream task
	Completed rate	The progress of the current task
	Iteration	Whether the current task is executed iteratively
	Item	The index of the product item operated during the current task
Cost	Hardware	The list of hardware resource
	Software	The list of software resource
Quality	Supply chain	The list of concerned suppliers, customers
	Document	The list of document involved in the current task

In the above tables of the illustrative direct parameters, we classify the attributes according to time, cost and quality/scope constraints. Corresponding to the time constraint, the attributes concerning the temporal aspects of project are collected, such as the time moments of project/phases/tasks, the deadline of the phases/tasks. Meanwhile, the attributes concerning the arrangement of phases/tasks and the progress of project are also collected, for example, the relations between the phases/tasks. Corresponding to the cost constraint, the attributes concerning the money cost, the resource utilization and consumption during the project are collected. Referring to the quality/scope constraint, the attributes concerning the objectives/achievements, the participants, the workload distribution are collected, such as the partners, the team. At the same time, the direct parameters used to manage quality of project should also be collected. So we suggest to collect and maintain the various documents during the project. The classification of the direct parameters according to the above three constraints are not restricted, and some of the attributes could be categorized according to the specific project practices. For example, the "completed rate of the project" could be mentioned as either time-related attribute by highlighting the temporal aspect or quality-related attribute by considering the risk of project risk. The similar condition could also be found when classifying the "milestone" attribute.

The procedure of identifying the direct parameters implies that the "values" of direct parameters are of many possible data types corresponding to the various properties (described by the "attribute name") they are measuring. For example, "integer" data type can be used to measure "the number of employees in project"; "float" data type can be used to measure "the budget to execute a task", etc.

In the above content, we elaborate our perspective of analyzing and modeling the project management knowledge area with the COCA framework, and suggest the hierarchical system model of project with the illustrative direct parameter lists organized in three levels correspondingly. As following, we continue to introduce our perspective of analyzing and modeling the product management knowledge area, in which we propose a product evolution model reflecting the evolving states of product along the PD project.

3.3.2 Product management knowledge area

Considering that the product model emerges in various forms along the process of PD project, we propose a four-state product evolution model that reflects the procedures of efforts to generate the product design solution (see the yellow coloured boxes in Figure 3.2).

In the product evolution model, the four states refer to *Needs* ***(N)***, *Requirements* ***(R)***, *Logical solution* ***(L)*** and *Physical solution* ***(P)***. These states capture a set of milestones in the evolution of the product model. In what follows we will try to make correspond these four states to the product development process made by those aforementioned six phases. However, readers should keep in mind that the exact correspondence and temporal synchronized are not the most important issue here. In fact, in our model, the most important concept is the four-state model of N, R, L, P.

There would exist lots of back and forth between each couple of the adjacent states, which reflects the possible iterations during the PD project and implies the mutual relations between the states. In section 3.5.1, we will elaborate this issue in details.

3.3.2.1 Product evolution

During the period of achieving the states, four *deliverables* are composed and maintained one by one to document the KID generated during the product evolution (see the bottom part of Figure 3.2). The deliverables are *Needs Definition*, *Requirement Definition*, *Logical Solution Representation* and *Physical Solution Representation*. Each deliverable is a set of generated documents and indicates :

– The obtained results from aggregating the contributed efforts during achieving one of the product evolution states, i.e., what the product model currently consists of and how the product model is made up given the collected and generated KID.

– The specified objectives of designing and developing the product, and the means used to determine whether the objectives are achieved.

During the product evolution process, the product firstly emerges as the "***needs***" that is forming from the beginning of the PD project. Along the project timeline indicated by the generic product design and development process, the "needs" state will evolve into the further state (i.e., the "requirements" state) during the "concept development". Before evolving into the further state, there could exist a *freezing period* of the "needs" state lasting an amount of time, and a *freeze point* (Eger *et al.*, 2005) as the end mark of the freeze period is prescribed. The freeze point describes the latest end point before which the KID aggregated by the contributed effort must be generated and collected completely. During the freezing period, the aggregated KID is, on one hand, still allowed to be modified internally, and meanwhile the further state would, on the other hand, bring in some iteration effect causing the KID to be changed externally (see Figure 3.4). During the product in "needs" state, the expectations, needs from acquirer and stakeholders are transformed to specifications. These information as well as some other derived technical requirements are contained in the deliverable "needs definition".

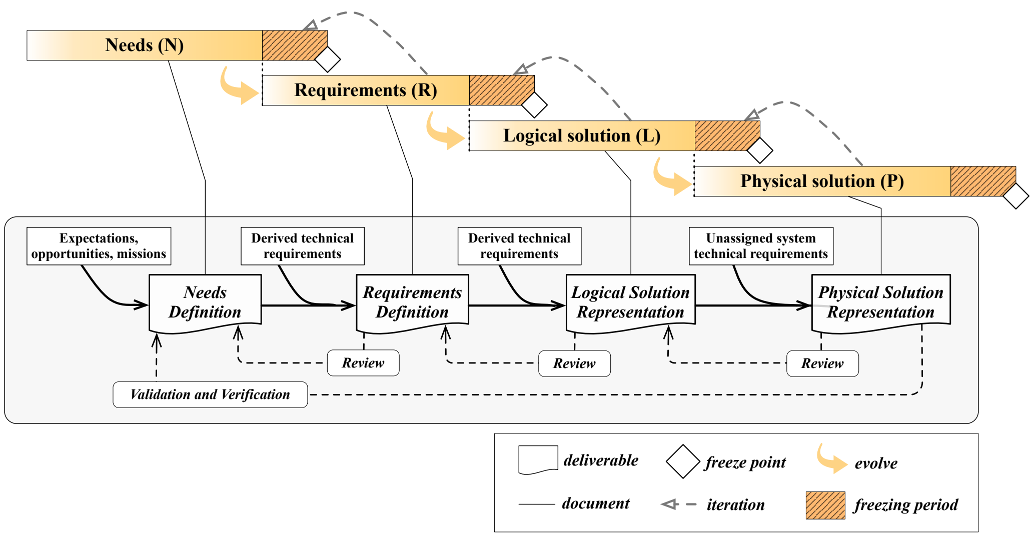

Needs (N)
Requirements (R)
Logical solution (L)
Physical solution (P)
Expectations, opportunities, missions
Derived technical requirements
Derived technical requirements
Unassigned system technical requirements
Needs Definition
Requirements Definition
Logical Solution Representation
Physical Solution Representation
Review
Review
Review
Validation and Verification
deliverable
freeze point
evolve
document
iteration
freezing period

FIGURE 3.4 – State relations

The "***requirements***" state is forming from the "needs" state. The freeze point of "requirements" state is deployed during the "system-level design". During the product emerging as "requirements" state, the feasibility of the potential product concepts are investigated and the industrial design concepts are developed. The deliverable documenting the "requirements" state contains some system technical requirements, and the design constraints concerning the performance and function of design solution. The constraints concerning production and the supply chain strategy as well as the requirements of enabling products could also be included in the deliverable.

The "***logical solution***" state is forming after the "requirements" state, and its freeze point is deployed at the end of "system-level design". During this state, the capability, behaviour, and structure of the product are defined. The product functional architectures, some derived technical requirements and a list of key suppliers or suppliers' classes are generated and recorded in the logical solution representation deliverable.

The "***physical solution***" state is forming after the "logical solution" state, and the freeze point is deployed at the end of "testing and refinement". During this state, the component is defined, and the tolerances of all the concerned parameters are identified and assigned. Also the plans of validating and verifying the design solution are composed. The make-buy analysis is executed according to the determined supply chain strategy. The physical solution representation deliverable records the specification of the subsystems and the design parameters. The design solution of the end product(s) as well as the alternative ones are also included in this deliverable.

3.3.2.2 Modeling dimensions

The product evolution process enables to model the product data with investigating the relations between the states and the relations between the building blocks belonging to the same state. In other words, the COCA framework models the product management knowledge area by solving two issues :

1. Reflecting the procedures of achieving one state from its previous one.
2. Presenting the interconnection model of the systems corresponding to the same state as well as the relations between the systems corresponding to the different states.

In this Ph.D. research, we model the product data with the four states emerging from collecting the needs derived from identifying, analyzing the expectations at the beginning of the PD project to releasing the design solution for the further manufacturing phase(s) (see Figure 3.5).

As illustrated in Figure 3.5, the product model emerges the four states through the iterative procedures of "need collection", "requirement discovery", "logical solution definition", and "physical solution definition". In other words, the above procedures would be operated iteratively according to the specific product design methodology (i.e., demonstrated by the cyclic arrows in Figure 3.5). Along the procedures, there exists a number of methodologies enabling to drive the product model evolving from one state to another one. Some methodologies are adopted during one of the procedures. For example, from the first state, i.e., "needs", achieved through collecting needs, expectations from stakeholders, the procedure of requirement discovery can employ requirement prototyping, reverse engineering, requirement reusing methods

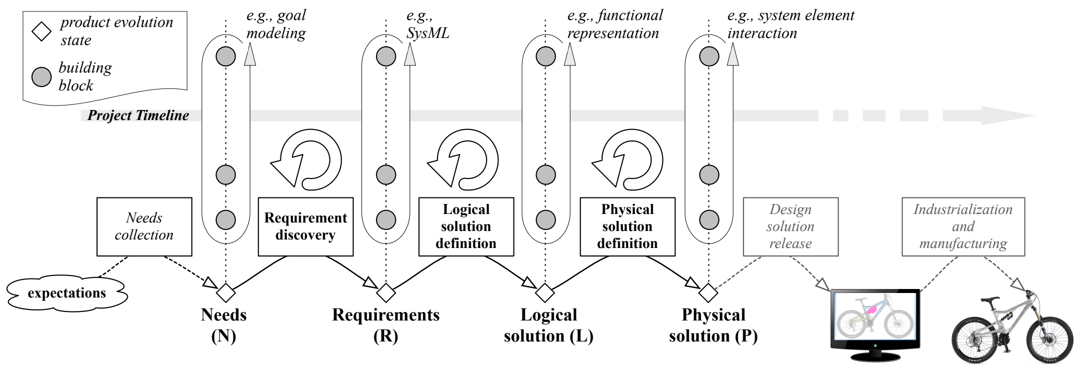


FIGURE 3.5 – Four-state product evolution model

to drive the product model evolving to "requirements" state (Alexander & Beus-Dukic, 2009). There are also some other methods being used during more than one procedures, such as the axiomatic design methodology, as a system design methodology, guides the designers to analyze the transformation of needs into the functional requirements, design parameters (Suh, 1990). Moreover, the functional modeling, such as the Function−Behaviour−Structure (FBS) scheme proposed by Gero & Kannengiesser (2004), could also help to analyze requirements, build up product architecture or even define physical model (Hirtz *et al.*, 2002).

With the states specifying, the product data is modeled systematically, i.e., the states prescribe a set of systems equipped with the respective objectives correspondingly. In this way, the product evolution model also exposes the interconnection model of these systems. At the same time, the systems could be analyzed and modeled through being decomposed into the subsystems in order to handle the complexity of the product data. In further, the obtained subsystems could be decomposed iteratively in further if necessary. In regarding that, we also use the concept of "*product building block*" to designate the obtained subsystems during the iterations as we did when analyzing project management knowledge area (see section 3.3.1). As illustrated in Figure 3.5, a set of building blocks could be identified and modeled corresponding to each of the states, and the building blocks are classified as *need building blocks*, *requirement building blocks*, *functional building blocks* and *physical building blocks* accordingly. Given each set of building blocks, there is also a number of techniques/methods enabling to model their interconnections. We mention several representative ones of them for each set of building blocks in Figure 3.5 briefly.

Among the product need building blocks, the building blocks are obtained through the needs collection procedure from the expectations. The need building blocks and their relations presents the preliminary description of the intentions made onto the final product (Alexander & Beus-Dukic, 2009). As we mentioned, *goal modeling* technique could be used to model the need building blocks and their relations. According to the methods based on goal modeling technique, a goal is explained as something to be achieved. During the needs expectation procedure, the expectations are identified and modeled as goals which are not restricted to be completely achievable and neither are measurable. In the further requirement discovery procedure, the goals would be analyzed into the targets that are measurable, and the possible conflicts between goals would be resolved.

To assist the designers/engineers in understanding enough of the product system, there are several models of systems being often used to present the different aspects of requirements. For example, the *OMG System Modeling Language* (denoted as SysML) [1] could be used to capture the what is required during the product evolution process. SysML provides a set of graphical objects to represent the requirements and reflect their relations. At the same time, it also provides the linkage between the requirements management tools and the system models.

Given the product functional building blocks, there are some methods based on functional representation techniques to reflect them, such as IDEF-0 [2], Functional-Behaviour-Structure (FBS) scheme (Gero & Kannengiesser, 2004). The functional building blocks could be organized by hierarchical way or others. Referring to the product physical building blocks, they could be modeled and analyzed by the system element interactions. Bill-Of-Materials (i.e., BOM) is one of the method to reflect the elements comprising the final product (Jiao *et al.*, 2000).

During the product evolution process, we focus on highlighting the functional and physical terms of the product based on the Function−BehaviourStructure (FBS) scheme proposed by Gero & Kannengiesser (2004) and identifying the product functional and the physical building blocks for building up the base where the change propagations will be identified and simulated in further.

The *product functional building blocks* (denoted as "functional building blocks" for short) are "the individual operations and transformations that contribute to the overall performance of the product", and the *product physical building blocks* (denoted as "physical building blocks" for short) are "the parts, components, and subassemblies that ultimately implement the product's functions" (Ulrich & Eppinger, 2007). During the product design, the functional building blocks describe the effects exposed by the physical building blocks (Wie *et al.*, 2005).

We mainly study two design methodologies to implement the functional building blocks with the physical ones (see Figure 3.6). Along the product evolution process, the overall functional system is identified through investigating and analyzing the customer expectations, market opportunity and/or project mission as the overall product

1. Version 1.3 of SysML has now been released as a formal specification by OMG, and it is accessed by the website : http ://www.omgsysml.org
2. i.e., Icam DEFinition for Function Modeling, where 'ICAM' is an acronym for Integrated Computer Aided Manufacturing

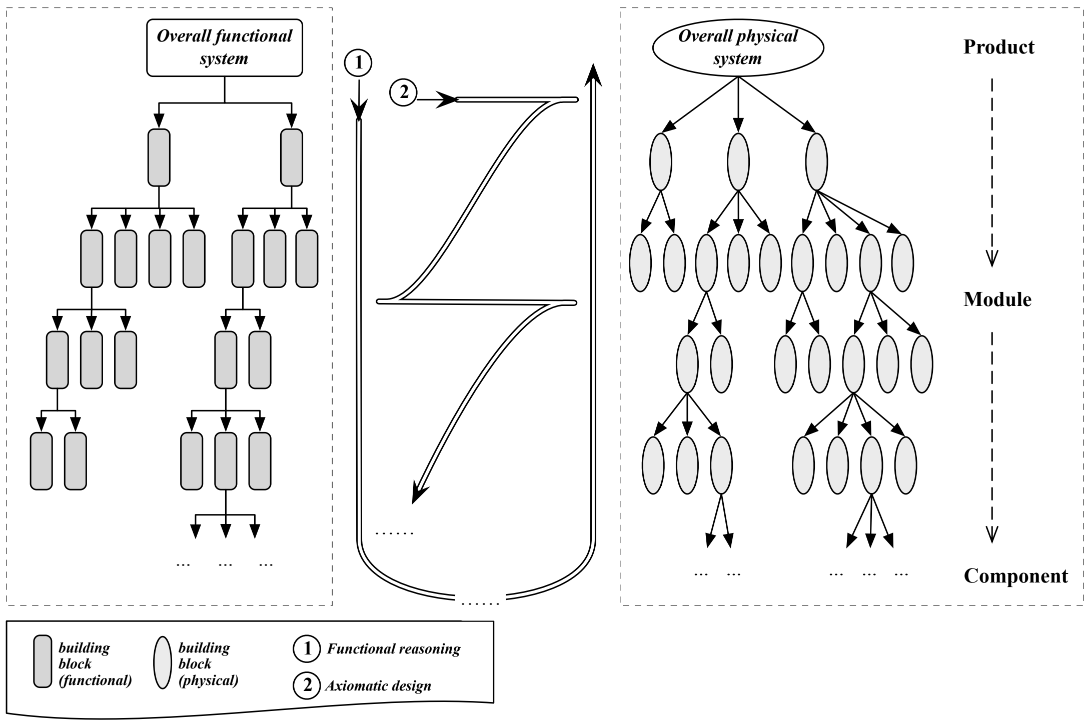


FIGURE 3.6 – Evolutions of functional and physical elements

design problem (i.e., the need building blocks, the requirement building blocks), and it is then more and more specified, branching into the functional building blocks. Along with the functional system evolving into the smaller functional building blocks, the overall physical system of the product (i.e., the designed artefact) is designed to be complete by either of the highlighted design methodologies (Chakrabarti & Bligh, 2001; Suh, 1990; Umeda & Tomiyama, 1997). Throughout the procedures of acquiring the complete overall physical system, the physical building blocks also emerge a hierarchical structure as that of the functional system evolution though they could be treated in either the top-down scheme (illustrated by ① according to (Chakrabarti & Bligh, 2001; Umeda & Tomiyama, 1997)) or the bottom-up scheme (illustrated by ② according to (Suh, 1990)). Meanwhile, as the hierarchical structure of the physical product suggesting, the overall physical product is decomposed into a set of modules, and each module is decomposed into the smaller physical elements. Then all the product building blocks are correspondingly classified into the product-related, the module-related and the component-related ones (see Figure 3.6). Although there is no clear restriction of determining the minimum granularity of the physical building blocks, the physical building blocks should not be decomposed any further when all the functional building blocks have been fully implemented and no more functional building blocks are elicited. We identify these final physical building blocks as the components of product.

With the building blocks modeled and classified, the corresponding direct parameters are also identified corresponding to the hierarchical structures of the overall physical system and the overall functional system.

We provide an illustrative list of the three levels of direct parameters in the current section to demonstrate how the physical system of product is parameterized with the FBS scheme (see Table 3.4). In the further illustrative case, we will provide a completed illustrative lists corresponding to the specific final product.

TABLE 3.4 – Direct parameters (product, module, component) of product management knowledge area

Property	*Attribute name*	*Description*
Function	Product/Module/Component function	The overall performance of product/module/component
	Architecture	Emerged with the informal representations
Behaviour	Expected behaviour	The expected property of entire product
Structure	Geometric specifications	The geometric layout, interfaces, and any other parameters for characterizing product dimensions
	Outlook	The appearance of product
	Materials	What the product is made in/from
	Cost	the cost spent to the product design

Referring to the detailed procedures of defining functional and physical building blocks, we will not cover this issue in this Ph.D. thesis considered as out of scope.

In the product evolution process, both the functional and the physical systems are documented in the deliverables.

As follows, we introduce our perspective of analyzing and modeling the partnership management knowledge area.

3.3.3 Partnership management knowledge area

During the product evolution process, the supply chain partners participate in the project, and contribute their efforts to the product evolution. According to the intentions of their supplied products, we categorize the partners into two classes, i.e., *end product suppliers* and *enabling product suppliers*.

- End product suppliers are *the companies or strategic business units supplying the end product(s) to the focal company*;
- Enabling product suppliers are *the companies simply providing enabling products to the focal company*.

One focal company is *a company from whose perspective the product evolution and the network of partners are to be analyzed, and it is a central role of information and material* *flows* (Hanf, 2009). The supplied products mentioned in the partner types, i.e., *end products* and *enabling products*, originate from the EIA-632 standard (1999). According to the standard :

– End product refers to *the portion of a system (i.e., final product) that performs the operational functions and is delivered to the focal company* ;
– Enabling product is *the item that provides the means of enabling the end product(s) to get into service, keep in service or terminate from service*.

The above categorized partners participate in the project at different time moments and are involved in different activities according to their supplied products. The time and activities are the two aspects we care about in the partnership in our research.

When considering the partnership management knowledge area, all the partners are firstly divided into the classes (i.e., end product suppliers, enabling product suppliers) by the criterion that what objectives of the products (i.e., end product, enabling product) they provide to the focal company are. We then identify the partner individuals (the companies or business units) in each class according to their own specific business scopes. During the product evolution process, the partners participate in the project with performing their responsibilities, such as delivering the components to the focal company when a particular task is started. In our research, we identify those responsibilities as the elementary building blocks in partnership management knowledge area. One partner can play multiple responsibilities according to the objective of the activity it is involved and the items (i.e., the product) it provides (see Figure 3.7).

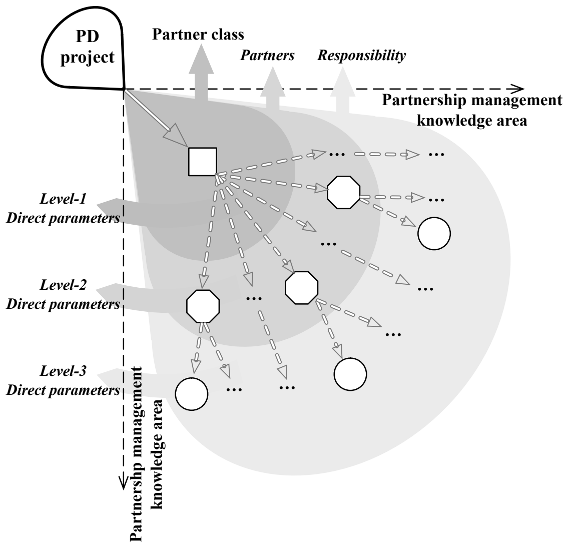

PD project
Partner class
Partners
Responsibility
Partnership management knowledge area
Level-1 Direct parameters
Level-2 Direct parameters
Level-3 Direct parameters
Partnership management knowledge area

FIGURE 3.7 – Hierarchical model of partnership

With the specified building blocks in Figure 3.7, we identify the corresponding direct parameters from two aspects, i.e., the *time of involvement*, and the *performance*. The former enables us to consider the behaviour of partners during the collaboration with the focal company (such as joining in, quit project), and the latter allows to identify the potential influences led by the partners. For example, a partner would bring some delay into the project and cause the focal company re-arrange activities. We suggest an illustrative list of the three levels of direct parameters to demonstrate characterizing the building blocks in the consideration in the time of involvement and the performance (see Table 3.5).

TABLE 3.5 – Direct parameters (partner class, partner individual, responsibility) of partnership management knowledge area

Property	*Attribute name*	*Description*
Time of involvement	History of collaboration (partner individual)	The list of past collaboration(s)
	Involved phase/task (responsibility)	The index of the phase/task participated in by the partner class/partner
Performance	Due time	The constrained time at which a partner should execute some task
	Specifications of supplied product (partner class)	The specification document of the supplied product
	Geographic information	The location of partner's company/plant/any other business unit
	Description	The general statement of partner class/partner/responsibility

Through the above three sections, we introduced the three highlighted knowledge areas in the COCA framework, and presented the hierarchical system models correspondingly that supports to acquire the information/data for the further change analysis. During each hierarchical modeling procedure, we suggest a set of modeling dimensions as the guideline to identify the direct parameters characterizing the building blocks. In the following section, we introduce the simultaneous consideration in the three knowledge areas and analyze the interactions between them.

3.4 A Simple Example

In this section, we create a simple example and will keep mentioning it in throughout this Ph.D. thesis to facilitate the readers to understand the further developments. The final product in this example is an assembly of three parts illustrated in Figure 3.8, and it is named as "ASE" (i.e., the abbreviation of "A Simple Example").

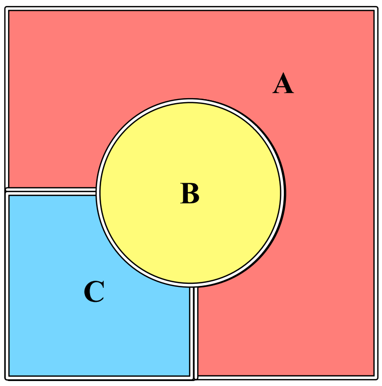


FIGURE 3.8 – An example final product

The product is specified based on our own expectation for the demonstration. The general function is described as “a square-shape artefact in which a circular part is mounted in the center”. The product is divided into two modules as a circular part, i.e., a component named as “B”, and a square part composed with two components, i.e., “A” and “C”. The square part is with a circular hole in the center which is mounted by Component B. The Bill-Of-Materials (BOM) of the product is illustrated as Figure 3.9.

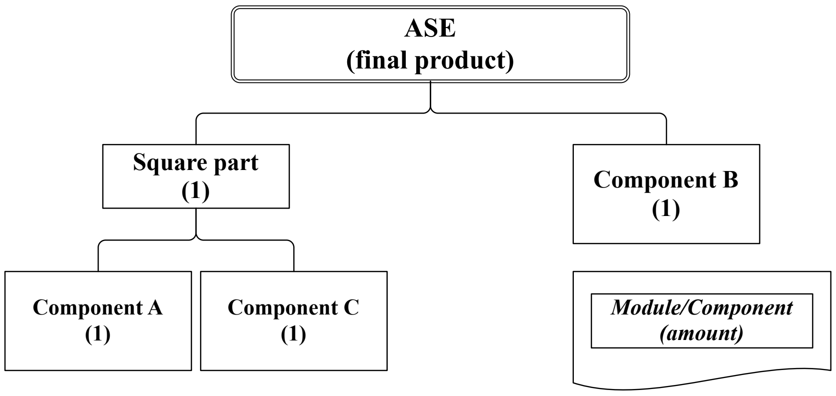


FIGURE 3.9 – Bill-Of-Materials (BOM) of final product

The geometric specifications of the components are illustrated in Figure 3.10.

In Figure 3.10, we label each of vertexes with a letter in order to specify all the geometric parameters. The parameters in Figure 3.10 only display an idealized parameter value, but they are allowed to vary within a small interval.

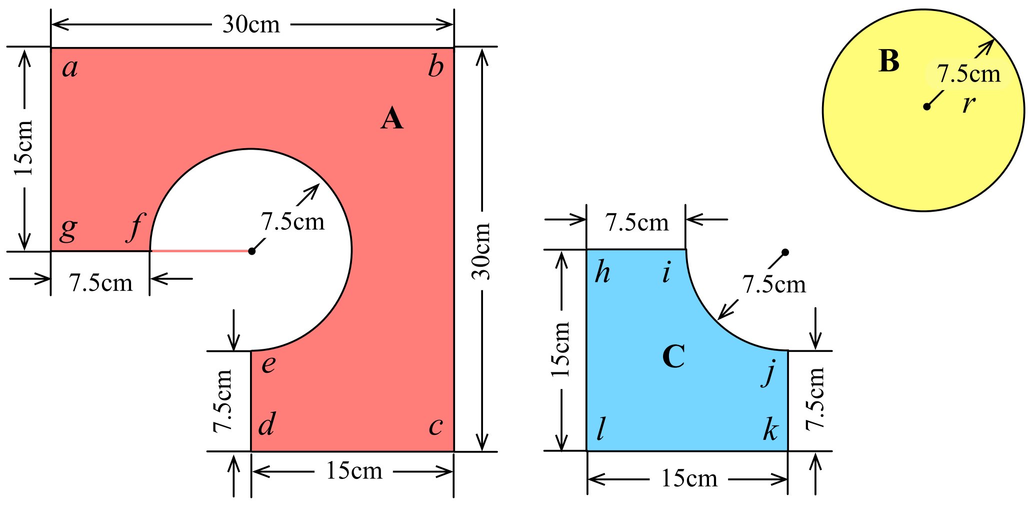


FIGURE 3.10 – Geometric specifications

Concerning the project management issues, we arrange a set of tasks to demonstrate the procedures of acquiring the product design solution (see Figure 3.11).

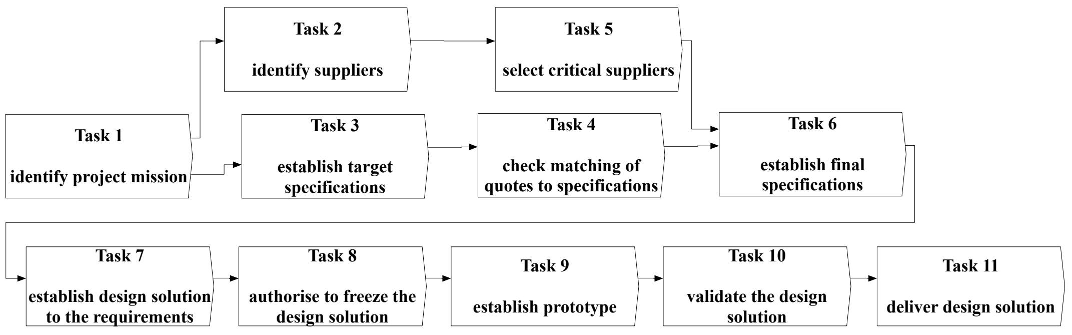

Task 1
identify project mission
Task 2
identify suppliers
Task 3
establish target specifications
Task 4
check matching of quotes to specifications
Task 5
select critical suppliers
Task 6
establish final specifications
Task 7
establish design solution to the requirements
Task 8
authorise to freeze the design solution
Task 9
establish prototype
Task 10
validate the design solution
Task 11
deliver design solution

FIGURE 3.11 – Task arrangement

The arranged tasks are described as following :

– **Task 1** : identify project mission. The focal company (FC)[3] begins a PD project under the preset objectives. In this example, the project mission is described as "design a square-shape artefact in which a circular part is mounted in the center". After the project mission is identified, the further tasks of identifying suppliers and establishing target specifications are started.

3. Although it is referring us, we will still keep using "focal company" in the context.

– **Task 2** : identify the potential suppliers offering the services to test and evaluate the product performance. The focal company identifies some of potential suppliers for building up the further collaboration in the project.
– **Task 3** : establish target specifications. With the project mission, the focal company creates the target specifications as the preliminary answer to the product objective in the project mission. However, the target specifications might be too arbitrary to be technically feasible.
– **Task 4** : check matching of quotes to specifications. Through communication with the suppliers who could offering testing and evaluating services, the focal company checks matching of quotes by the suppliers by considering the time cost and the price of the service. The obtained result would support the focal company to select the critical suppliers.
– **Task 5** : select critical suppliers. From the identified suppliers, the focal company selects one or more critical suppliers. The critical suppliers refers the ones participate in the project since an early time and share risk/revenue with the focal company, and meanwhile their effort would affect the progress of project
  In the example, we only have one critical supplier named as "ASP company". This supplier offers the evaluation and test service since Task 6, and it shares the market risk and revenue with the focal company.
– **Task 6** : establish final specifications. After approving the project plan and determining the critical supplier, the focal company re-considers the specifications according to the various trade-offs (such as the technological constraints, production costs, and the evaluation from the supplier) and then finalize the target specifications into the final specifications.
  In the example, we specify the direct parameters of this task and the supplier participating in the task. The direct parameters are :
  (1) Planned task start date : 15th of August ;
  (2) Earliest task start date : 10th of August ;
  (3) Latest task start date : 17th of August ;
  (4) Freeze of task start date : 21st of August ;
  (5) Planned task finish date : 25th of August ;
  (6) Latest task finish date : 29th of August ;
  (7) Freeze of task finish date : 31st of August ;
  (8) Planned due time of delivering service : 23th of August ;
  (9) Latest due time of delivery : 24th of August ;
  (10) Freeze of due time of delivery : 24th of August.
– **Task 7** : establish design solution to the requirements. Referring to the approved requirements, the focal company establishes the corresponding design solutions that is illustrated in Figure 3.10.

Through executing this task, the geometric parameters are all specified. During this task, the supplier provides the service of computing the gap width between all the components.

– **Task 8** : authorise that the design solution is frozen. Through reviewing the design solution in the cooperation with the supplier, the focal company makes the decision to declare whether the design solution is accepted or not. If accepted, the focal company authorizes the design solution to freeze which implies that any change to the design would be rejected. If the authorization could not be made, then the project plan would be adjusted correspondingly.
  The focal company designs all the three components, and the supplier (i.e., ASP company) provides the service of testing the gap width between the components which enables the focal company to evaluate the performance of the design solution.
– **Task 9** : establish prototype. Based on the test tool and the test plan, the focal company establishes the prototype in advanced which would be used to test, validate and verify the design solution.
– **Task 10** : validate the design solution. Through adopting the prototype(s), the focal company validates the design solution of the product.
– **Task 11** : deliver design solution. After validating the design solution, it is delivered to the next process for manufacturing.

3.5 Interactions between Knowledge Areas

When analyzing the three knowledge areas simultaneously, we take use of product evolution process (product management knowledge area) as the main clue to connect the other two knowledge areas. This is because the product evolution process demonstrates the main target of a PD project, i.e. to provide the final product. All the other activities and participants are organized around this target. Therefore, we follow a guideline to connect the three knowledge areas :

*To make the product evolves towards the design solution under the determined design methodology (**product**), what activities/tasks will be executed under what constraints (**project**), and is there any partner involved in and what are their responsibilities (**partner**) ?*

If putting this guideline to the whole process of product evolution, we can find out the interactions between the three knowledge areas throughout the whole duration of the PD project. In this case, it is necessary to understand how different states of the product relate with each other, because the relations between the product states also reveals the potential relations of activities/tasks and the indirect relations of partners. This section will introduce the relations between deliverables which reflect the relations between product states as well as the building blocks. The relations are categorized into the *effort-based* and the *non-effort-based* ones.

The effort-based relation refers to the relationship between two objects reflecting that the KID is aggregated by the exertion from human and/or nonhuman resource. One of the two objects is taken as the basis, whereas the other one as the objective and achievement. The non-effort-based relation refers to the rest relationship between two objects.

3.5.1 Effort-based relations : generation and contribution

As we mentioned in section 3.3.2, the product manifests itself with a serial of states during the evolution process. Corresponding to the product evolution states, four deliverables document the results obtained during the product being emerging each of states, and each two adjacent deliverables (denoted as upstream deliverable and downstream deliverable respectively) are interrelated through effort-based relations. The upstream deliverable indicates the goal to be achieved and therefore generates the downstream deliverable, whereas the downstream deliverable indicates the plan/solution contributing to achieve the goal. The mutual effort-based relation between the deliverables refers to the following two relationships (see Figure 3.12).

- Generation : Given the goal in the upstream deliverable, the corresponding plan(s)/solution(s) is/are created or produced in the downstream deliverable.
- Contribution : The plan/solution in the downstream deliverable contributes in achieving the corresponding goal created in the upstream deliverable through supplying the produced effort.

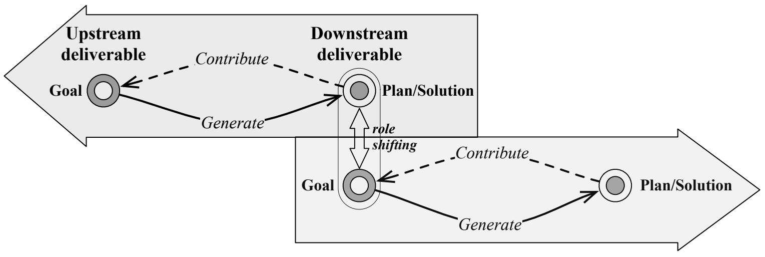


FIGURE 3.12 – Mutual relations between deliverables in a chain

Based on the above statement, the four deliverables in the product evolution process and the mutual relations between them compose a chain. In this chain, one deliverable is either the goal of its following deliverable or the plan/solution of its previous deliverable (see Figure3.12). To be noticed that, the freezing periods of the product evolution state reserve the possibility in which the iterations are allowed through the contribution relations (see Figure 3.4).

Focusing on the four states (i.e., Needs denoted as ***N***, Requirements denoted as ***R***, Logical solution denoted as ***L*** and Physical solution denoted as ***P***) reflected by the deliverables, there are techniques to go from one state to the other (see section 3.3.2.2).

By acting in this way, the states are interrelated in terms of the generation relations. The generation relations reflect the effort of making the product evolve from one state to another. Along the generation relations, the KID is produced, aggregated and developed and the project is progressing. Associating with each of the generation relations, a contribution relation is identified and it traces the effort contributed during the evolution procedures. Therefore, the mutual relation (generation and contribution) is classified as the effort-based relation.

In considering the systems exposed and modeled from the deliverables during the product evolution process, i.e., the product need building blocks, the requirement building blocks, the functional building blocks and the physical building blocks corresponding to their belonging states, the mutual effort-based relations between the deliverables are embodied by the relationships between their eliciting building blocks (see Figure 3.4 and Figure 3.5). In further, a number of effort-based relation chains of building blocks could be identified through the four deliverables (see Figure 3.13).

Through analyzing the characteristics of the building blocks belonging to the different deliverables as well as the evolution along the corresponding states (see Figure 3.5), we suggest a set of generation situations between each two related building blocks along the effort-based relation chains(see Figure 3.13).

1. **One-To-Multiple mapping** : Along the generation relation, one building block as the goal in the upstream deliverable maps to more than one building blocks as the plan/solution in the downstream deliverable. For instance, within the highlighted chain in Figure 3.13, l_1 has two generation relations with p_1 and p_2.
2. **Multiple-To-One mapping** : Along the generation relation, more than one building blocks as the goal in the upstream deliverable map to one building block as the plan/solution in the downstream deliverable. For example, within the highlighted chain in Figure 3.13, a Many-To-One mapping can be discovered from n_3, n_4 to r_3.
3. **One-To-One mapping** : Along the generation relation, one building blocks as the goal in the upstream deliverable maps to one building block as the plan/solution in the downstream deliverable.

The effort-based relations between the deliverables are not only representing the states transformation during the product evolution but also could transfer the impact of occurred changes as change propagation channels. In other words, the change occurring in the goal deliverable could cause another change in the adjacent plan/solution deliverable through their mutual relations. The four deliverables and their mutual relations contribute to compose a change propagation network, in which change pro-

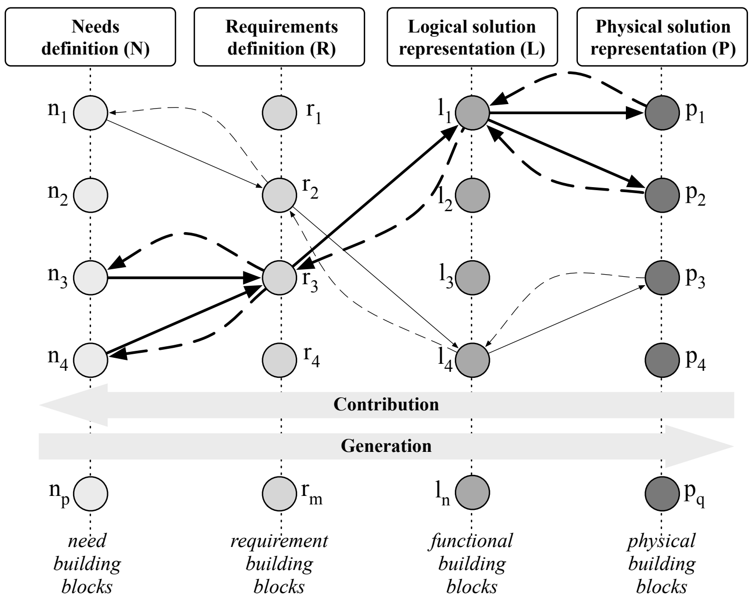

Needs definition (N)
Requirements definition (R)
Logical solution representation (L)
Physical solution representation (P)
n1
n2
n3
n4
np
r1
r2
r3
r4
rm
l1
l2
l3
l4
ln
p1
p2
p3
p4
pq
Contribution
Generation
need building blocks
requirement building blocks
functional building blocks
physical building blocks

FIGURE 3.13 – Relation chains of building blocks through deliverables

pagations are along either generation or contribution direction between the adjacent deliverables.

In the simple example (cf. section 3.4), the project mission as the goal deliverable could generate the target specification as the plan/solution deliverable. We can find that the specification of "the shape of Component B" is generated by the statement that "a circular part is mounted in the center" in the mission. If there comes a change that "a square-shape component is mounted in the center", then "the shape of Component B" would be also changed accordingly. Thus, a change could be propagated from a goal deliverable to the adjacent plan/solution deliverable.

3.5.2 Non-effort-based relations : Dependencies between Direct Parameters

Besides the effort-based relations, there are other relations between the building blocks, i.e., the non-effort-based relations. Compared with the building blocks interrelated with the effort-based relation, both of the building blocks with the non-effort-based relation can not be qualified neither goal nor plan/solution. Depending on that, we then identify and model non-effort-based relations by studying the dependencies between the direct parameters characterizing the building blocks.

Referring to the direct parameters, we discover there exist various *dependencies* between them. In our research, a dependency between two direct parameters is defined as *the effect of the change in one direct parameter's value on another* according to the definition proposed in (Andrew & Juite, 1995). These direct parameters could belong to either the same knowledge area or the different one. Referring to the former condition, the dependencies between the direct parameters could be identified and modeled corresponding to the characteristics of the knowledge area, such as the activity-based matrix for the project management knowledge area and the component-based matrix for the product management knowledge area (Browning, 2001). Referring to the latter condition, the dependencies could be identified and modeled by the guideline of identifying the interactions between the knowledge areas (cf. section 3.5), and the Domain Mapping Matrix (DMM) could be adopted (Mike & Browning, 2007) to reflect, such as, the components are treated during executing the particular tasks.

In Table 3.6, we present six cases of dependencies between direct parameters corresponding to the combination in Figure 3.14[4]. With the dependencies, we also present some representative research contributions. These contributions proposed the methods of modeling the dependencies under the different cases.

4. use the data from the simple example in section 3.4

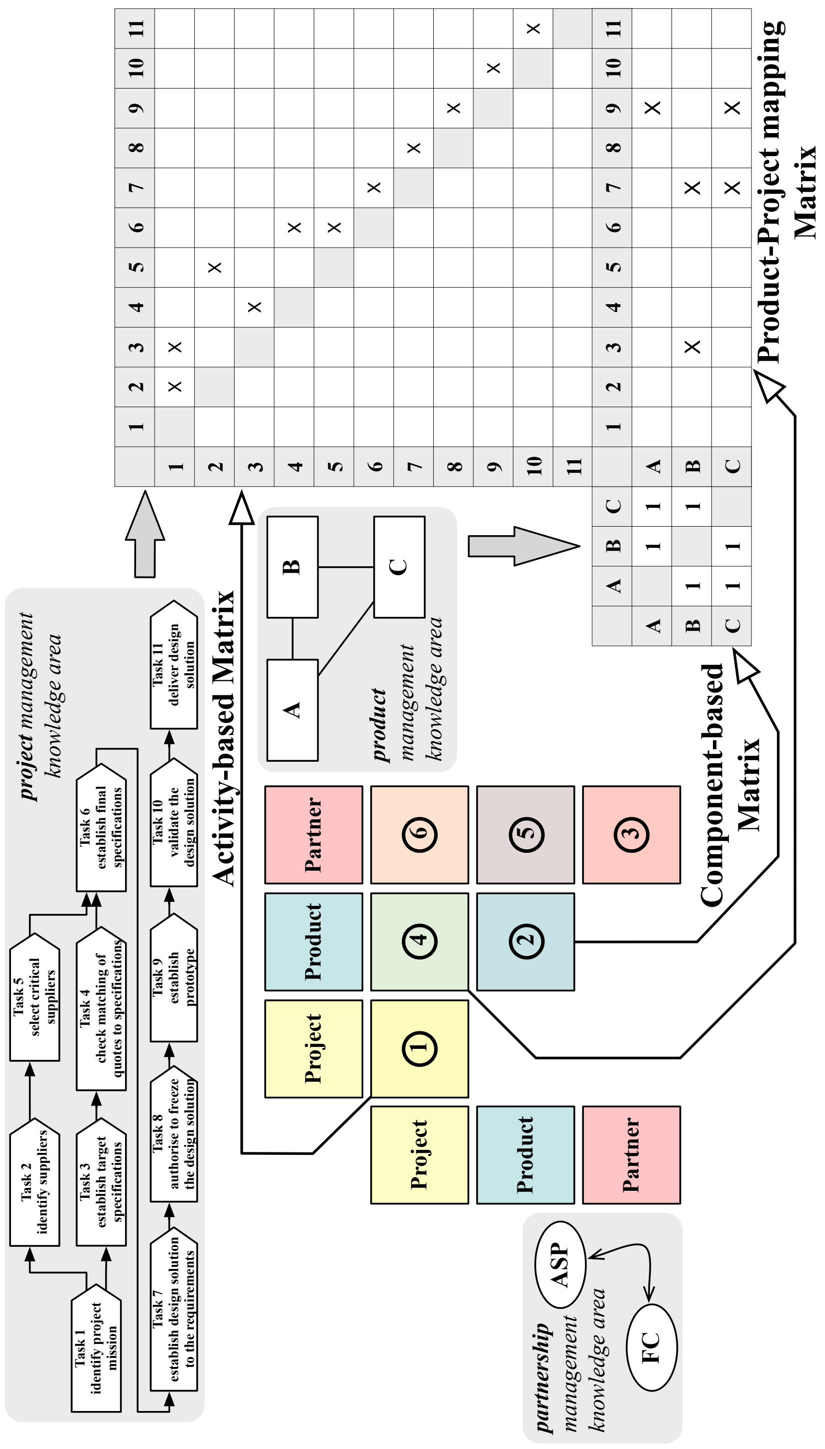

project management knowledge area
Task 1 identify project mission
Task 2 identify suppliers
Task 3 establish target specifications
Task 4 check matching of quotes to specifications
Task 5 select critical suppliers
Task 6 establish final specifications
Task 7 establish design solution to the requirements
Task 8 authorise to freeze the design solution
Task 9 establish prototype
Task 10 validate the design solution
Task 11 deliver design solution
Activity-based Matrix
product management knowledge area
A
B
C
Component-based Matrix
Product-Project mapping Matrix
partnership management knowledge area
ASP
FC
Project
Product
Partner
1
2
3
4
5
6

FIGURE 3.14 – Dependency situations

TABLE 3.6 – Classification of dependencies

Mapping	*Case N°*	*Methods and example references*
Project∼Project	N°1	– Six modelling "primitives" (Carley & Krackhardt, 1999) ; – Activity-Based Design Structure Matrix*(Browning, 2001) ; – Activity dependencies (Browning *et al.*, 2006).
Product∼Product	N°2	– Dependency structure matrix, Component-Based Design Structure Matrix*(Browning, 2001) ; – Dual-domain analyses of product issues (Mike & Browning, 2007) ; – Dependency at creation/modification, consistency, redundancy (Ouertani & Gzara, 2008).
Partner∼Partner	N°3	– Team-Based Design Structure Matrix*(Browning, 2001) ; – Partners dependency modeling (Zouggar *et al.*, 2009) ; – Mutual dependencies between partners (Zolghadri *et al.*, 2010).
Product∼Project	N°4	– Domain Mapping Matrix*(Mike & Browning, 2007) – Generalized Bill-Of-Materials and Operations (Zolghadri *et al.*, 2010) ;
Product∼Partner	N°5	– Dependencies between multiple domains (Danilovic & Börjesson, 2001) ; – Incidence matrix mapping activity to design parameters* (Zouggar *et al.*, 2009).
Project∼Partner	N°6	– Hidden dependency between partners (Ulrich & Eppinger, 2007) ; – Incidence matrix mapping activity to design parameters* (Zouggar *et al.*, 2009) ; – Generalized Bill-Of-Materials and Operations (Zolghadri *et al.*, 2010).

* used to model dependencies in the Ph.D. research

In the Ph.D. research, the dependencies between the direct parameters are mainly measured from two aspects : the direction and the dependent rate, which are proposed by Andrew & Juite (1995). The former implies the direction of the modification of one parameter that is affected by another ; the latter implies the rate of modification of a parameter that is affected by another.

Moreover, according to the definition of dependency, the dependencies between two building blocks could also transfer the effect of modification from one block to another. In our research, the dependencies also imply the change propagation channels. The details of how dependencies work as propagation channels are explained in Chapter 4 (readers can refer to section 4.1.2).

Relying on the deliverables corresponding to the product evolution states, we mainly modeled the mutual relations between the building blocks in accordance with the project timeline. Meanwhile, within each of the deliverables, the dependencies reflected by the coupled direct parameters within the same knowledge area and from the different knowledge areas were considered. Let us consider the simple example (cf. section 3.4). In the "logical solution representation" deliverable, there is a dependency between the diameter parameter of Component B and the arc length of inner surface of Component A which belong to the product management knowledge area. The effect of modifying the former parameter (such as increasing the parameter value) can be identified in the latter parameter, i.e., causing the latter parameter value to be increased. Moreover, there also exists another dependency between the geometric parameter of Component B and the responsibility of the partner (i.e., ASP Company), and this dependency involves in two different knowledge areas.

As taking the product evolution evolution process as the main clue to analyze the knowledge areas simultaneously, we are enabled to model the relationships between the building blocks within one of the knowledge areas and from different ones.

3.6 Structuring Product Development (PD) Project

Through highlighting the multiple knowledge areas and analyzing the interactions between them as well as between the building blocks, we are enabled to model the PD project progress and the product evolution process with considering the participation of the partners at the same time (see Figure 3.15).

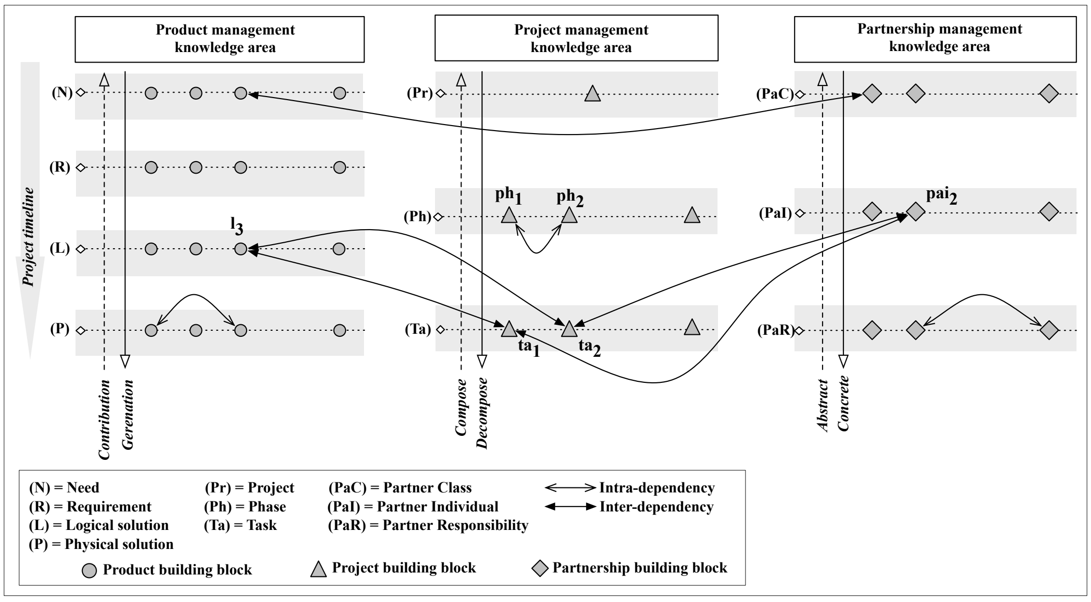


FIGURE 3.15 – Relations among building blocks

In Figure 3.15, the building blocks belonging to the three knowledge areas are modeled in the hierarchical models (see section 3.3). Corresponding to the building blocks belonging to the same knowledge area or the different knowledge areas, their relations are classified as *intra-knowledge area* and *inter-knowledge area relations*.

As Figure 3.15 illustrating, we combine the considerations in modeling the three knowledge areas together with analyzing the inter-knowledge area relations between them. For the product management knowledge area, the product evolution model is proposed and it enables us to obtain the product building blocks which are related through the generation and contribution relations in terms of the project timeline. These building blocks are categorized into four deliverables that reflect the product evolution states along the project progress. So the intra-knowledge area relations can be categorized into *intra-knowledge area effort-based* and *intra-knowledge area non-effort-based relations* in the product management knowledge area. In other words, the intra-knowledge area effort-based relations can only identified between the building blocks belonging to the product management knowledge area. Referring to the project management knowledge area, the project building blocks are obtained through a hierarchical modeling method, by which a project is decomposed into a set of phases and each phase is decomposed into a set of tasks. For the partnership management knowledge area, we mainly consider the contributed effort and the participations of partners to identify the partnership building blocks in the knowledge area. In the high level, we consider partners based on the effort they would contributed to the PD project and proposed two partner classes, i.e., end product suppliers and enabling product suppliers. We then concrete to identify the partner individuals corresponding to the two classes. The partner individuals refer to the specific organizations. Afterwards, we concrete to model the responsibilities that the partner individuals could take during their participation in the project (See Table 3.7).

TABLE 3.7 – Relations between building blocks

	Effort-based	*Non-effort-based*
Intra-knowledge area	intra-knowledge area effort-based relation (only belonging to product management knowledge area)	intra-knowledge area non-effort-based relation (reflecting dependencies between direct parameters)
Inter-knowledge area	-	inter-knowledge area non-effort-based relation (used to create constructs)

With the building blocks obtained through the above procedures as well as the relations between them, we create an entity named as *construct* through aggregating every three building blocks according to the guideline of modeling the interactions between the knowledge areas (cf. section 3.5). These three building blocks are from the three knowledge areas respectively, and they are related through inter-knowledge area relations. In this way, we are enabled to analyze change propagations within and between the three knowledge area simultaneously.

3.6.1 Constructs : structured representations in PD project

A construct is specified as the "artefact" through aggregating the building blocks from the three knowledge areas respectively. Each construct is identified and determined by the following procedures.

– Each product building block is firstly selected as the identity of the construct.
– Given the selected product building block, each of its dependent project building block is then selected as the secondary identification of the construct.

This means we use product building block first to differentiate constructs. If the two constructs contains the same product building block, we will turn to project building block.

Let us consider the simple example (cf. section 3.4) in Figure 3.15. l_3 represents a product functional building block documented in the logical solution representation deliverable, i.e., the functional specification of Component B in the simple example. ta_1 represents a task, i.e., Task 6, and ta_2 represents Task 7. pai_2 represents the partner, i.e., ASP company. Accordingly, two constructs can be identified as follows.

– Construct 1 : it is aggregated by "l_3", "ta_1" and "pai_2" and indicates that the final functional specification of Component B is established in Task 6 with cooperating with ASP company. The company supports to determine the specification with the knowledge and know-how.
– Construct 2 : it is aggregated by "l_3", "ta_2" and "pai_2" and indicates that the geometric parameter of Component B answering its functional specification is established in Task 7 with cooperating with ASP company. The company provides the service of computing the gap width between Component B with the others.

When distinguishing these two constructs, we find both having the same product building block "l_3", therefore, we continue to look up their project building blocks and learn that the project building blocks are different by which the two constructs are differentiated.

Particularly, there might exist a situation that two constructs have the same product building block, the same project building but different partner building blocks. In this case, we combine the two constructs into one construct that has two partner building blocks.

Here we continue with using the simple example created in section 3.4 to demonstrate the procedures of acquiring the constructs in details. At the earlier phase of the project (see Figure 3.11), the project mission is identified (i.e., Task 1) and the tasks of identifying suppliers and establish target specifications are to be launched. At this moment, since the information/data is still little, and it can only describe the product as a "block box", i.e., a square-shape artefact in which a circular part is mounted in the center. From the statement, we can identified two physical building blocks as "a square-shape artefact with a circular-shape hole in the center" and "a circular -shape artefact". Therefore, two constructs are identified as illustrated in Figure 3.16.

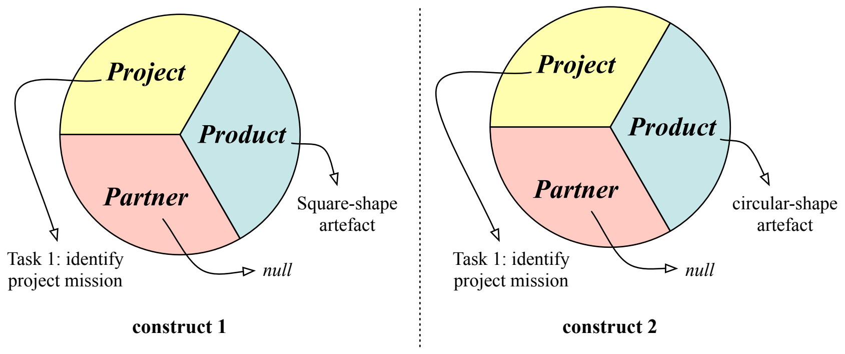


FIGURE 3.16 – Examples of construct

The above identified constructs as well as the mutual relations between the building blocks and the dependencies between the direct parameters enable us to build up a network, in which the phenomena of change occurrence and change propagation can be analyzed. In the following chapter, we begin to depict our research in identifying change occurring within a construct and change propagation.

3.7 Chapter Summary

This chapter introduced the Co-evolution Oriented Change Analysis (COCA) framework. The main contributions were :

- Considering the multiple aspects of a PD project simultaneously, i.e. modeling the product management, project management and partnership management knowledge areas simultaneously.
- Revealing the change propagation channels by analyzing the interactions between the three knowledge areas.
- Proposing a way of aggregating information and data from different knowledge areas and further to help the analysis on change occurrence and change propagation.

It also described a simple example that will be used to explain the complex models and ideas in the further chapters.

4
Identifying Change Occurrence and Propagation

▷ *In general understanding, change is described as "an act or process through which something becomes different" according to The Oxford Dictionary. To describe the "act/process" in a concreted manner for computing and simulating change propagation, we firstly propose a conceptual model of change occurrence and change propagation based on the structuring model of PD project in the previous chapter. The conceptual model describes the procedure of perceiving the occurrence of change and change propagation. Then we turn to the analysis of the direct parameters that are used to improve the conceptual model. At the end of this chapter, we present a hierarchical method to identify and analyze the change propagations patterns.* ◁

Plan of Chapter

4.1 Conceptual Specification of Change Occurrence and Propagation

In our research, we introduce an innovative concept called "*alteration*". An alteration generally specifies the shifting effect that "*makes something different*" and it is led into the parameters involved in the PD project as the precondition of identifying change occurrence. Through analyzing the consequence due to introducing the alteration, we prescribe a procedure of perceiving the occurrence of change. As elaborated in the previous chapter, the Knowledge/Information/Data (KID) of the PD project is aggregated as the structured representations, i.e., the constructs. Corresponding to the highlighted knowledge areas in the COCA framework, each construct is composed of three respective building blocks which are the data entities generated and educed according to the design methodology along the product evolution process. A building block emerges a set of attributes which are characterized by the direct parameters. Alterations will be introduced into these direct parameters. Therefore, we present the conceptual model through investigating the direct parameters.

4.1.1 Change occurrence conceptual model

Referring to a direct parameter, it measures the response, the characteristics and/or the behaviour of a system (or mentioned as a building block) by its exposed value (i.e., the parameter value) and emerges a prescribed attribute. In practice, the direct parameter is assigned and restricted with not a particular fixed value but an interval within which the parameter value is always valid and acceptable during the product design. The mentioned interval is denoted as "*tolerance domain*" and the specific values located at the upper bound and the lower bound are denoted as "*maximum boundary value*" and "*minimum boundary value*" respectively. When the parameter value varies within the tolerance, the performance of the direct parameter attribute is correspondingly acceptable. We denote the interval composed with all the acceptable attribute emergences as "*reference domain*". We are then enabled to investigate the potential consequence due to introducing alterations into the parameter value. (see Figure 4.1)

In the conceptual model illustrated in Figure 4.1, the horizontal axis of the coordinate system refers to the parameter value of the involved direct parameter, whereas the vertical axis refers to the attribute charcterized by the direct parameter. In the coordinate system, the curve indicates all the possible performance given the valid assigned parameters. Then two alterations are introduced to the direct parameter value respectively (denoted as "$alteration_1$" and "$alteration_2$"). Due to the shifting effect brought by the alterations, one current valid direct parameter (illustrated as the node of white colour in the curve) is assigned with a new value in two conditions. In the condition where $alteration_1$ is introduced, the current valid parameter is assigned with another value that is still within the tolerance domain and the educed attribute still emerges

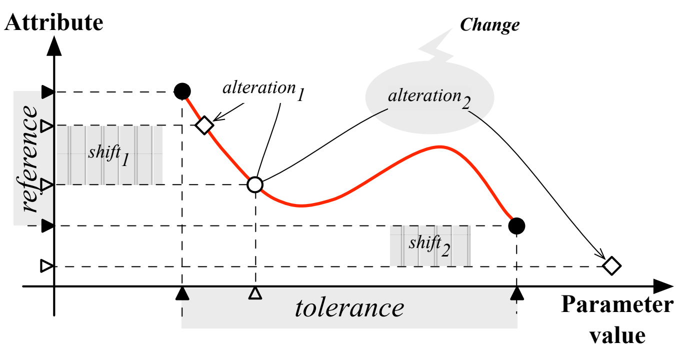


FIGURE 4.1 – Change conceptual model

within the reference domain. In a second condition where $alteration_2$ is introduced, the current valid parameter is assigned with a value that is out of the tolerance and makes the attribute emerge out of the reference domain. With the above two conditions, we identify the change occurrence from the second condition (in which $alteration_2$ is identified as a change) while there is no change being perceived due to $alteration_1$.

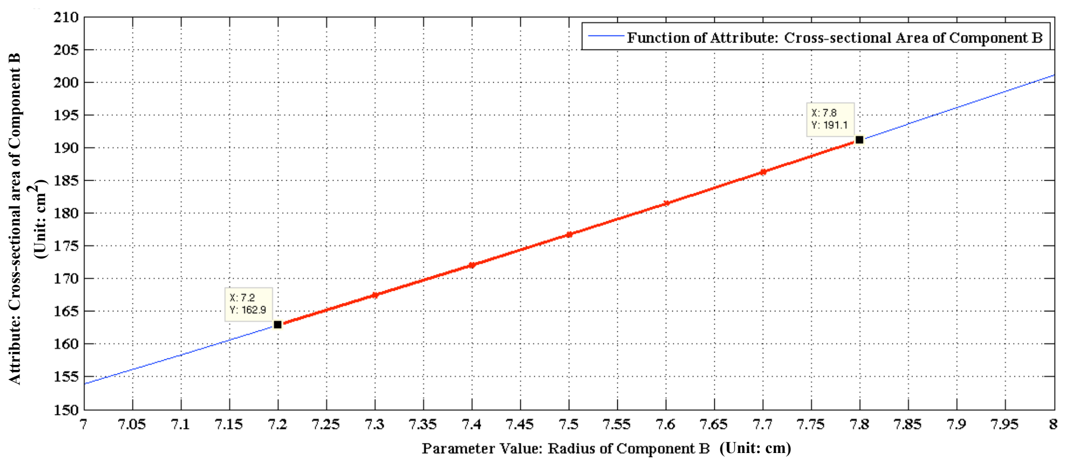


FIGURE 4.2 – Direct parameter : Radius of Component B

To make the above conceptual model more understandable, we turn to the simple example suggested in the previous chapter and select one of direct parameters (i.e., "Radius of Component B" formalized by variable "R", also see Figure 4.5) to investigate change occurrence (see Figure 4.2). The selected direct parameter is used to characterize the attribute of "Cross-sectional area of Component B" (formalized by variable "S_B", i.e., the shadow area of Component B in Figure 4.5), which is expressed by a function as :

$$S_B = \pi R^2$$

In the example, the value of the selected direct parameter is with the tolerance domain $[7.2cm, 7.8cm]$ and the reference domain of the attribute is $[162.9cm^2, 191.1cm^2]$ correspondingly. With introducing an alteration, the valid value of the parameter is shifted to $7.025cm$ that make the attribute with the value of $155cm^2$. According to the conceptual model of change occurrence, we identify that a change occurs in the direct parameter (see Figure 4.3).

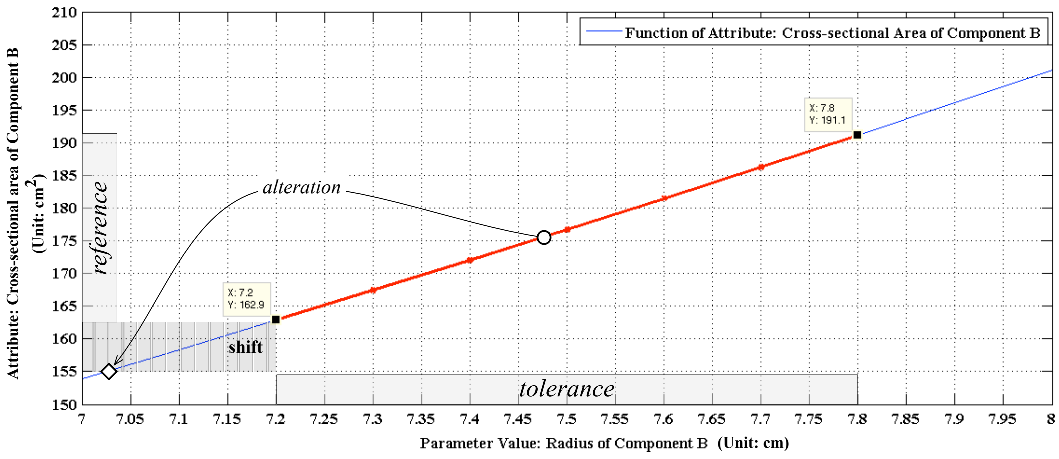


FIGURE 4.3 – Introducing alteration into direct parameter of Component B

Still in the simple example, Task 6 is planned to start on 15th of August and finish on 25th of August (see section 3.4), and the task is also arranged with the earliest/latest start and the latest finish date as 10th/17th and 29th of August. With the the earliest and the latest start dates, the start time margin of task is present. During the margin, the task could be started on any one day without causing any change occurrence. If the task is started later than the latest start date (i.e., 17th of August) but still earlier than the freeze of start date (i.e., 21st of August) for some reason, then a change occurrence would be identified. Referring to the phenomenon that the task is started later than the freeze of start date, we will explain it in further content.

4.1.2 Change propagation conceptual model

Based on the model of change occurrence, we extend to observe on change propagation that is a phenomenon that a change cause another one. We take two building blocks whose owned parameters are with some relation, such as dependency between the direct parameters (the building blocks are denoted as *"Building Block_1"* and *"Building Block_2"* respectively).

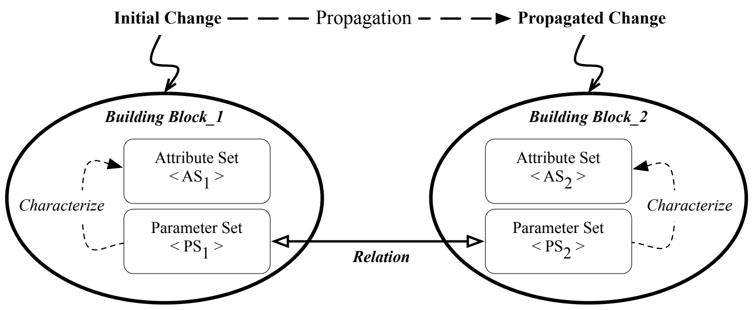


FIGURE 4.4 – Change propagation conceptual model

In Figure4.4, each build block has a set of direct parameters (i.e., parameter sets denoted as "PS_1" and "PS_2" respectively) characterizing its attributes (i.e., attribute sets denoted as "AS_1" and "AS_2" respectively). Between the two sets of direct parameters owned by the building blocks, there is some relations, such as the dependency between the direct parameters characterizing the geometric attributes. Through the relation between the involved direct parameters, some alteration led in the direct parameter owned by Building Block_1 could cause another alteration to the corresponding direct parameter owned by Building Block_2 (cf. the statement of dependency mentioned in section 3.5.2). Then the phenomenon of change propagation is specified as that an alteration introduced into Building Block_1 causes a change to occur (i.e., *initial change*) and leads a second alteration in Building Block_2 through the relation. The second alteration, if it is out of the tolerance domain, will cause another change (i.e., *propagated change*) in Building Block_2.

In the above change propagation conceptual model, we select two related building blocks to analyze and present the phenomenon. In the same principle, change propagations can also be modeled between two related direct parameters, constructs (cf. section 3.6.1). In the further section, we will extend this point in details.

We still use the simple example to explain the change propagation conceptual model. From the example, we select Component A and Component B as the two building blocks where to investigate change propagation (see Figure 4.5).

For Component A, we highlight its attribute of "Cross-sectional area of Component A" (formalized as "S_A", i.e., the shadow area of Component A in Figure 4.5) characterized by the parameter value of "Arc length of inner surface" (fomalized as "L", i.e., the bold dash line of Component A in Figure 4.5) with the function as :

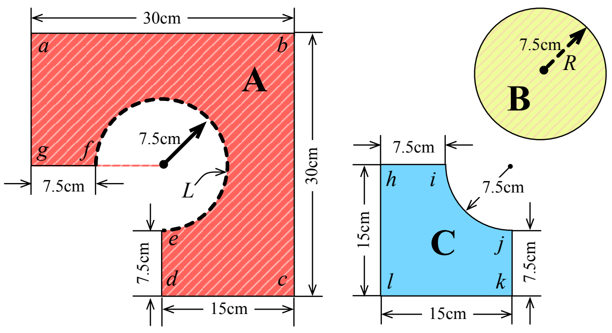


FIGURE 4.5 – Reminder : the simple example

$$S_A = 675 - \frac{L^2}{3\pi}$$

The above function is illustrated by the coordinate system in Figure 4.6 :

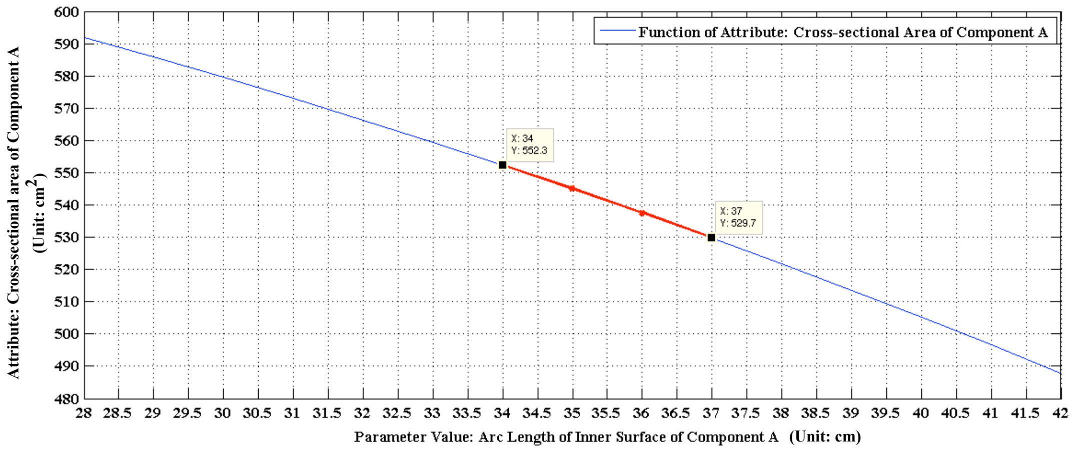


FIGURE 4.6 – Direct parameter : Arc length of inner surface of Component A

As illustrated in Figure 4.6, the parameter value is with the tolerance domain $[34cm, 37cm]$, where as the reference domain of the attribute emergence is $[552.3cm^2, 529.7cm^2]$ corresponding to the tolerance domain.

For Component B, we still highlight its "Cross-sectional area of Component B" (formalized as "S_B", i.e., the shadow area of Component B in Figure 4.5) by the parameter value of "Radius" with the functional as (formalized as "R", i.e., the bold dash arrow line of Component B in Figure 4.5, also see Figure 4.2) :

$$S_B = \pi R^2$$

Between the above two direct parameters (i.e., R and L), one dependency is identified as :

$$L = \frac{3\pi R}{2} \quad \text{or} \quad R = \frac{2L}{3\pi}$$

The above dependency is illustrated in Figure 4.7 :

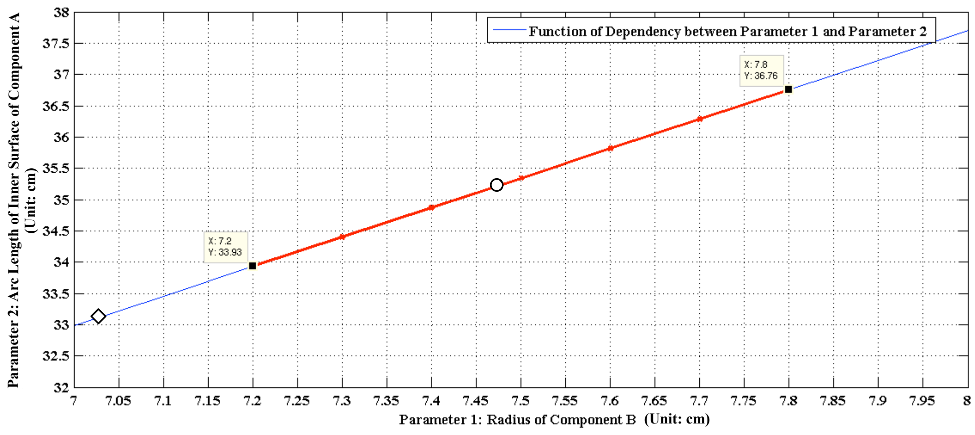


FIGURE 4.7 – Dependency between Radius of Component B (i.e., Parameter 1) and Arc length of inner surface of Component A (i.e., Parameter 2)

With introducing the alteration shifting the parameter value to $7.025cm$ into the above direct parameter of Radius of Component B, we identify a change (i.e., the initial change) occurs in Component B (also see Figure 4.3). Meanwhile, due to the dependency between the parameters, another alteration is introduced to the direct parameter of Arc length of inner surface of Component A and shifts the parameter value to $33.1cm$ (see Figure 4.8).

Given the above alteration into the direct parameter of Arc length of inner surface, a change (i.e., the propagated change) is identified in Component A (see Figure 4.9).

Through the above procedures, we identify a phenomenon of change propagation between the direct parameters belong to Component A and Component B.

In the following section, we extend to investigate the characteristics of the direct parameters involved in analyzing change occurrence and propagation.

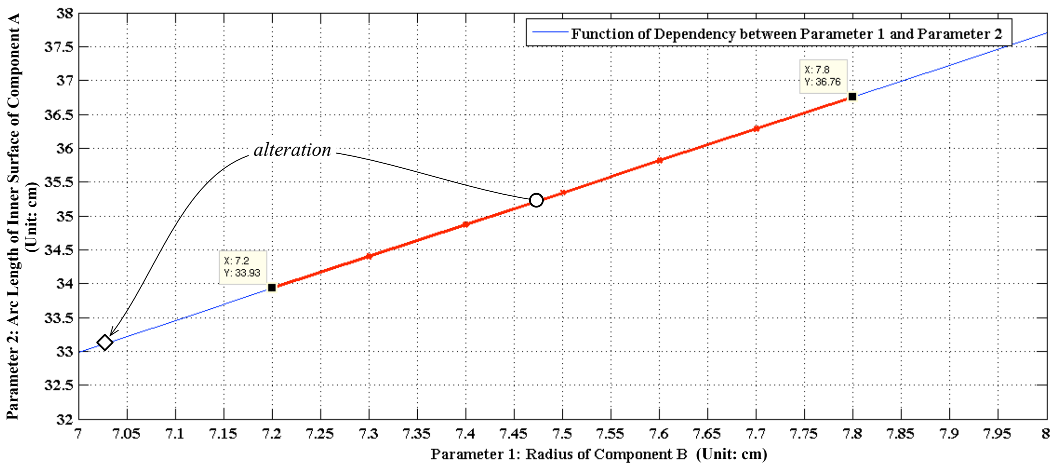


FIGURE 4.8 – Alterations to direct parameters

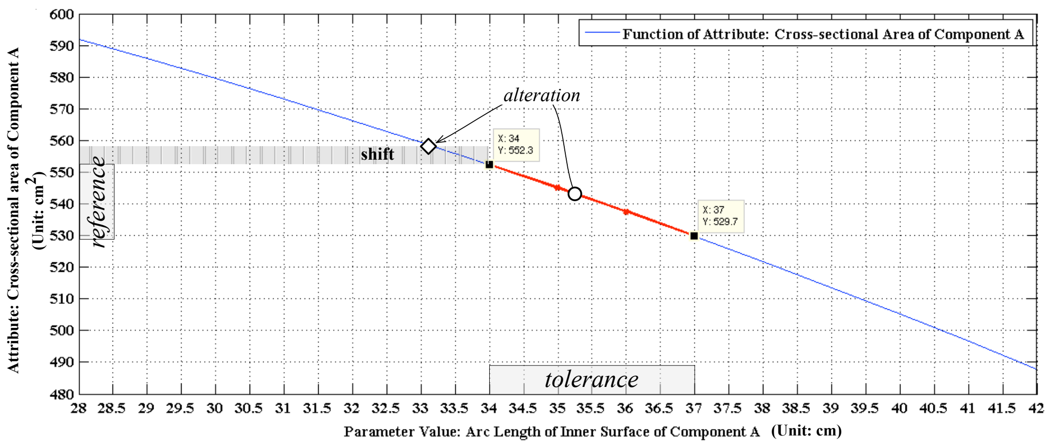


FIGURE 4.9 – Introducing alteration into direct parameter of Component A

4.2 Involved Direct Parameters

As the change conceptual model specifies, a change can be identified and perceived when the two following qualifications are met simultaneously :

– Alteration is introduced into a direct parameter that is characterizing a building block ;
– Due to the alteration, the emergence of the involved attribute of the building block is out of its prescribed reference domain.

To identify the characteristics of the direct parameters, we turn to identify and analyze the real data used by the focal company to analyze the data format and type of the direct parameters which are operated during the business. The perceived data are analyzed and specified in a number of parameter types in Table 4.1 according to the following consideration.

1. The information exchanged internally and externally during the focal companies' business is collected.
2. Through analyzing the data presented in all kinds of forms, the functionalities of various types of data are identified.
3. According to the functionalities of data, the formats of the data are investigated.

TABLE 4.1 – Parameter Types

Name	**Data Format**	**Function**
plain text	string	stating content of plain text
financial data	float with two decimals	stating numeric values for financial usage
geometry representation	graphical shapes	stating figures for graphical demonstration
quantity value	integer	stating amount
precise numerical value	real number	stating precise value
date value	numeric char	stating date
time value	numeric char	stating timing
terminology	string	stating terms in particular subjects
symbol	mark and/or char	stating conventional representations
code	mark and/or char	representing other implications
unit	mark and/or string	stating a standard for measurement

4.2.1 Characteristic value of direct parameters

Through investigating the parameters of the above types illustrated in Table 4.1, we suggest six parameter characteristic values during the PD project inspired by the contribution by Kerzner (2009).

– PERFECTION : It refers to the optimal value of the involved parameter according to the focal objective of design irrespective of any trade-off and/or obstacle.

– BASE : It refers to the particular value of the involved parameter within the tolerance domain and the value is the furthest one away from the optimal value (i.e., PERFECTION value) of the parameter.
– TARGET : It refers to the preset target value of the involved parameter established in an arbitrary manner.
– ACHIEVABLE : It refers to the extreme boundary value of the parameter educed based on the past experience, and the value and is the closest one to the PERFECTION value.
– PLANNED : It refers to the final specified value of the involved parameter in considering the various constraints/rules, the trade-off, etc.
– ACTUAL : It refers to the latest value of the involved parameter.

According to the above description of the characteristic values, the following statement is implied (cf. Table 4.2 as example) :

1. BASE and PLANNED value specify the tolerance domain of the involved direct parameter, and any alteration only shifting the parameter value within this domain would not cause any change.
2. The offset between TARGET and PLANNED value implies some possible trade-off and/or reserved margin maintained during product design.
3. If an ACTUAL value is out of the domain specified by BASE and PLANNED values, then the phenomenon of change occurrence would be perceived.

Based on the above specified direct parameter characteristic values, we educe five expectations made onto the parameters during the PD project.

1. *Monotonic increasing expectation : "The larger, the better"*
2. *Monotonic decreasing expectation : "The smaller, the better"*
3. *Approaching extreme expectation : "The closer, the better"*
4. *Leaving extreme expectation : "The farther, the better"*
5. *Enumeration expectation : "Membership or not"*

With respect to the conceptual model of change occurrence, the ACTUAL chararcteristic value is probable to be found within any one of the domains specified by the other characteristic values due to some alterations. Then we extend to consider the above mentioned expectations in the conceptual model and suggest five corresponding scenarios (see Figure 4.10).

In Figure 4.10, we describe each scenario with a separated coordinate system, in which the horizontal axis refers to the parameter value as the same as that in the conceptual model of change occurrence and the vertical axis refers to the expectation degree to the parameter value. In the horizontal axis, the parameter value turns higher and higher along the direct of the axis, and so does the expectation degree in the vertical axis. Comparing with the coordination system of the conceptual model of change occurrence, the coordination system illustrated in Figure 4.10 indicates the relation between

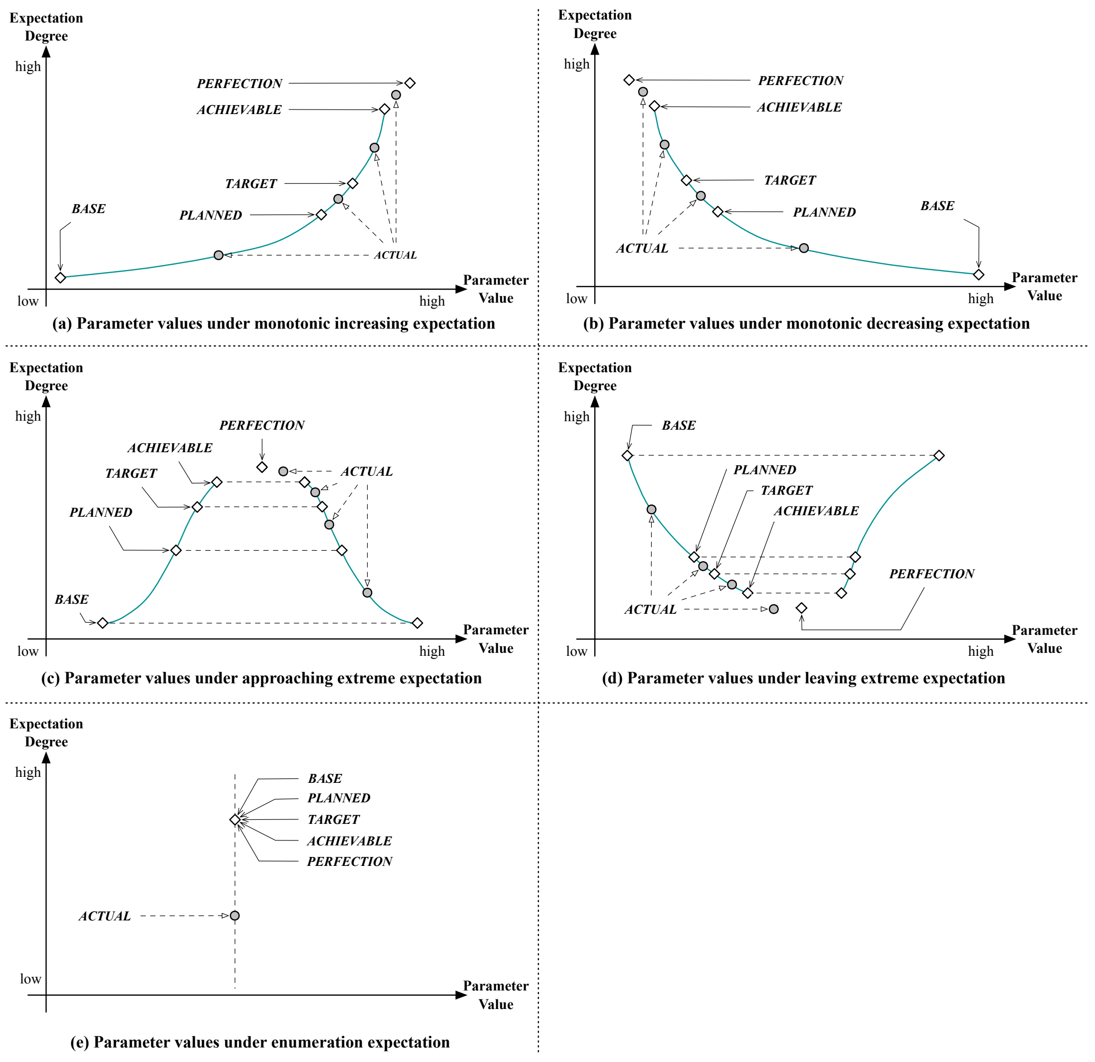

Expectation Degree
high
low
Parameter Value
PERFECTION
ACHIEVABLE
TARGET
PLANNED
BASE
ACTUAL
(a) Parameter values under monotonic increasing expectation
(b) Parameter values under monotonic decreasing expectation
(c) Parameter values under approaching extreme expectation
(d) Parameter values under leaving extreme expectation
(e) Parameter values under enumeration expectation

FIGURE 4.10 – Parameter states under five expectations

the product design behaviours and the direct parameters whereas the former reflects the mechanism in which the direct parameter values characterize the attributes. These two types of coordination systems are compatible.

In Figure 4.10, the trends of each curve represents the expected evolution of the actual parameter value, which plays an important role in prescribing change occurrence. Although the forms of the curves also have some influence, we decided to not tackle this issue.

Readers should keep in mind that the curves in Figure 4.10, representing expected evolution of the actual parameter value, have an important role to play in prescribing change occurrence.

Under the scenarios illustrated in Figure 4.10, some complement could be made through studying the characteristic values :

1. The various expectations of parameter values imply the ways of affecting the performance of the involved attribute (i.e., the higher expectation degree). For example, under the monotonic increasing expectation (see Figure 4.10(a)), a larger parameter value (i.e., evolving closer to PERFECTION value) is regarded as better performance of the corresponding attribute.
2. Under the various expectations, there exist several solutions to shift the parameter value in order to affect the performance of the involved attribute. For example, under the monotonic increasing and deceasing expectations (see Figure 4.10(a) and Figure 4.10(b)), the better performance of the involved attribute could be perceived through deploying a larger and a smaller parameter values respectively. However, under the approaching extreme and the leaving extreme expectations (see Figure 4.10(c) and Figure 4.10(d)), either increasing or decreasing the parameter value to get closer to PERFECTION value could be the solutions to improve the performance of the involved attribute.
3. Given the various expectations, the data format of parameter value could also restrict the way of affecting the performance of the involved attribute. For example, under the enumeration expectation (see Figure 4.10(e)), the data of the parameter value are of enumeration. In this case, if ACTUAL value is not equal to any other characteristic values, then it is hardly possible to identify that ACTUAL value calls forth better performance of the corresponding attribute.

The characteristic values of parameter values and the educed statement do not only present the basic situations of assigning values to the direct parameters, but they also elicit some practices of predicting and controlling changes. In further chapters, we will adopt these practices to facilitate the decision making procedures during simulating change propagations.

Referring to the scenarios, we suggest a set of examples with the specific parameter values and especially highlight the ACTUAL values due to introducing some alterations of causing change occurrence (see Table 4.2).

4.2.2 Domains of direct parameter value

The above characteristics of direct parameters imply that an encountered parameter value could not only qualified as either valid or not in practice by comparing the

TABLE 4.2 – Examples of parameter values under the expectations

	Scenario 1	**Scenario 2**	**Scenario 3**	**Scenario 4**	**Scenario 5**
	Budget margin	**Delivery delay**	**Weight, size**	**Sympathetic vibration**	**color**
PERFECTION	200 euros	zero day	1 kg	6Hz	dark blue
PLANNED	70 euros	four day	1$\pm$ 0.25 kg	15Hz	dark blue
ACHIEVABLE	150 euros	two day	1 $\pm$ 0.05 kg	18Hz	dark blue
TARGET	80 euros	three day	1 $\pm$ 0.2 kg	17Hz	dark blue
ACTUAL	90 euros	five day	1 $\pm$ 0.3 kg	14Hz	light blue
Expectations	**The larger, the better**	**The smaller, the better**	**The closer, the better**	**The farther, the better**	**Membership or not**

value with the boundary ones of the tolerance but it should be also evaluated in considering the various constraints, trade-off(s) and any other requirements from the design process. Therefore, we propose to refine the tolerance in further. Based on the identified characteristic values of direct parameters, the whole parameter value definition domain is divided into three parts (see Figure 4.11).

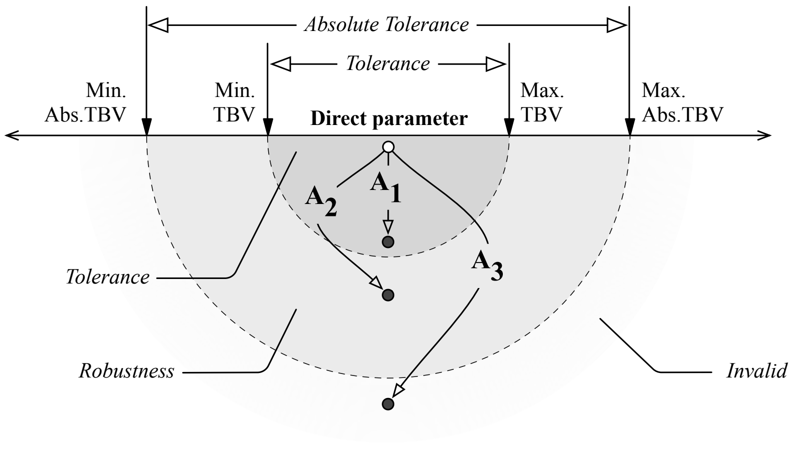


FIGURE 4.11 – Domains of direct parameter value

As illustrated in Figure 4.11, two pairs of characteristic boundary values are prescribed.

1. *Minimum/Maximum Tolerance Boundary Values* (i.e., denoted as *Min.TBV* and *Max.TBV*) : they refer to the minimum and maximum values the direct parameters could be assigned with. These values as well as the ones between them is designated in advance.

2. *Absolute Minimum/Maximum Tolerance Boundary Values* (i.e., denoted as *Abs.Min.-TBV* and *Abs.Max.TBV*) : they refer to the specific minimum and maximum values the direct parameter could be assigned with though some values between them might not be designated or expected. However, these unexpected values are still valid and are able to make the corresponding attributes emerge correctly.

With the above pairs of characteristic boundary values, the direct parameter definition domain is divided into three parts as :

– *Tolerance* : it refers to the designated domain defined by the Minimum/Maximum Tolerance Boundary Values (i.e., $Min.TBV$ and $Max.TBV$) in which the direct parameter values are allowed to vary without causing any change occurring. This domain is specified with respect to the various constraints, trade-off and any other requirements designated from the design process.
– *Robustness* : it refers to the domain defined by the Absolute Minimum/Maximum Tolerance Boundary Values (i.e., $Abs.Min.TBV$ and $Abs.Max.TBV$) and it is either equal to or larger than the tolerance domain. All the parameter values from this domain are valid to perform the corresponding attributes correctly as designed. However, in the case that this domain is larger than the tolerance domain, the offset between this domain and the tolerance domain represents some trade-off and/or extra capacity and/or any other reserved margin though the contained direct parameter values are still valid. Thus, the domain between Absolute Tolerance and Tolerance is specified as *robustness domain*.
– *Invalid* : it refers to the domain that is out of the robustness domain and specified as *invalid domain*. The direct parameter values from this domain are invalid with regard to the design process. The corresponding attributes by the direct parameter values from this domain will never be guaranteed as correct or even are not existent.

When introducing alterations into the direct parameter values, the parameter value is probable to be shifted into any one of the above domains. Corresponding to the arrived domain after shifting the parameter value by the alteration, we identify three consequences (see Figure 4.11 and Figure 4.12).

1. *Regular alteration* (illustrated by arrow A_1) : the alteration only causes the direct parameter value shifted within the tolerance which indicates the preset variation allowance of the involved direct parameter value, and no change occurs. We describe this consequence as a regular alteration.

   For example, the length parameter value[1] of a pencil is specified as $190mm$ by standard, but the acceptable length parameter value might be varied around the above standard value. So the allowed variation domain, i.e., the tolerance, set as $[-0.5mm, 0.5mm]$ is reasonable. In this case, if an alteration shifting the length parameter value shifted within the domain as $[189.5mm, 190.5mm]$, then the conse-

1. learned from http ://en.wikipedia.org/wiki/Pencil

quence is perceived as “regular alteration”. The regular alteration indicates that the introduced alteration only causes the preset variation, and it could not make the involved direct parameter become different. So the regular alteration can be understood as *regular variation* preserved for the involved direct parameter.

2. *Change occurrence* (illustrated by arrow A_2) : the alteration causes the direct parameter value shifted out of the tolerance but still within the robustness domain, i.e., the direct parameter value is still valid. In this condition, the consequence is described as a change occurrence.

   For example, the start date of a task along a PD project could be taken as a direct parameter (mentioned in Table 3.3), which is assign with a date type value. If the time arrangement of the task preserves some margin, then the start date could be associated with the “earliest start date” and the “latest start date” which enable the start date to be varied with causing some regular alteration occurrence, i.e., the tolerance domain. Due to an alteration, the start date could be shifted out of the tolerance domain, such as the task would be started only one day later than the latest start date. In this condition, the consequence from the alteration could be identified as a change occurrence if the one-day delay could still be handled and accepted under the capability of time management.

3. *Dysfunctional change occurrence* (illustrated by arrow A_3) : the alteration causes the direct parameter value shifted out of the robustness domain, i.e., the direct parameter value is invalid and then can not be treated by any further treatment. The consequence in this condition is described as some dysfunctional change occurrence.

   For example, a partner individual is required to deliver the supplied items at the due time (mentioned in Table 3.5). After an alteration is introduced, the due time would be delayed by quite a long time, such as the nature disease, due to which the project would never be completed. So a dysfunctional change occurrence is identified since the alteration would make the direct parameter not valid any further.

In Figure 4.12, we use the simple example mentioned in section 4.1.2 to explain the consequences of introducing alterations into the different domains of the parameter value. As the simple example specifying, the tolerance domain of the parameter value (i.e., the arc length of inner surface of Component A) is $[34cm, 37cm]$, and the robustness domain is specified as $[33cm, 34cm]$ and $[37cm, 37.5cm]$. The rest is the invalid domain. In the figure, three alterations are introduced and labeled with ①, ② and ③ respectively. Then three consequences are identified according to the conceptual model of change occurrence. The alteration labeled with ① only shifts the default value to another one that is still within the tolerance domain, and the consequence is identified as regular alteration. The alteration with ② causes the default value out of the tolerance domain and still within the robustness domain, and the consequence is identified as change

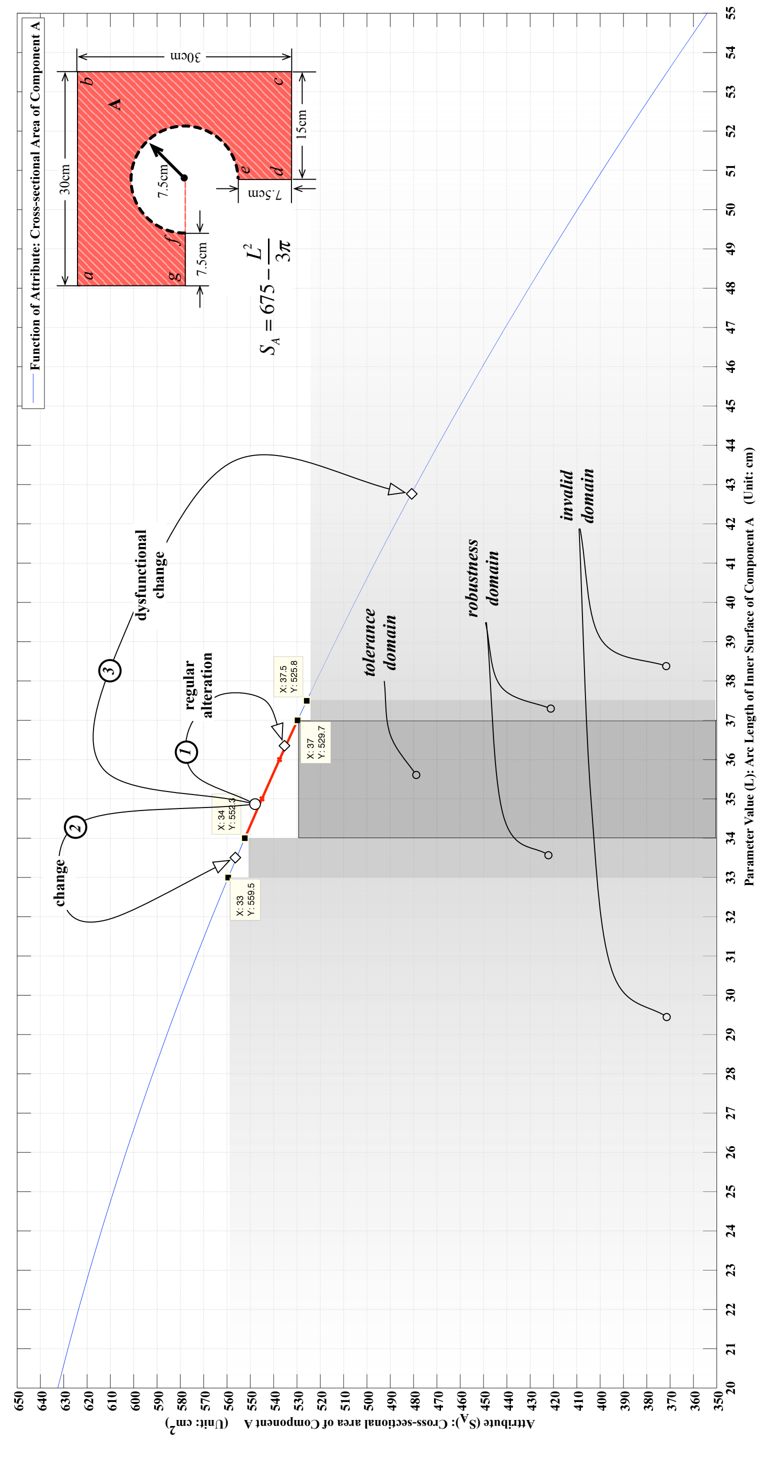


FIGURE 4.12 – Consequences of introducing alterations into the domains

occurrence. The alteration with ③ causes a dysfunctional change occurred due to the default value is shifted into the invalid domain.

Based on the above identified consequences from introducing the alterations shifting the direct parameter values to different domains, we then integrate them into the change propagation conceptual model in order to facilitate identifying and analyzing the corresponding change propagation patterns.

4.3 Change Propagation Patterns

According to the change propagation conceptual model, the phenomena of change propagation can be perceived between two related direct parameters, building blocks, and constructs. In accordance with the characteristics of the focused objects (i.e., direct parameters, building blocks, constructs) in change propagations, we propose a hierarchical method of identifying and analyzing change propagations patterns.

1. *Data* : When identifying and analyzing change propagations between two related direct parameters, the alterations is probable to shift the parameter value to any one of the prescribed domains (i.e., tolerance, robustness, invalid domains) and then cause several possible consequences (i.e., regular alteration, change and dysfunction occurrences). Meanwhile, the relations between the direct parameters are embodied with the data dependencies which could all be formalized as "the effect that one direct parameter value is changed onto the other" (cf. section 3.5.2). With the above mentioned consequences from introducing alterations and the dependencies highlighted between the direct parameters, the change propagation patterns with respect to the characteristics of direct parameter data are identified.
2. *Effort* : The building blocks as well as their relations identified in the Ph.D. research reflect the effort contributed to generate the KID along the PD project (cf. section 3.5.1). Two building blocks are identified as related since there exists the mutual relation reflecting the effort trajectories as generation and contribution between them. Through the mutual relation, the alteration introduced into one building block by shifting its owned direct parameter(s) could cause another alteration(s) into the related one. In consideration of that, the change propagation patterns are extended. In section 4.3.2, we will explore these patterns.
3. *Temporal* : The constructs are composed through aggregating the building blocks from the three knowledge areas respectively. When identifying and analyzing change propagations between two constructs, we begin to consider the timing moment when alterations are introduced and thus adopt the concept of "*freeze*" from (Eger *et al.*, 2005) to specify an end point of a procedure, after which any alteration can not be introduced. Or if it has to be introduced, it will cost a huge amount of money and time to cope with. Often, this effort is too big to make it

acceptable. Relying on that, we also consider the constraints from temporal aspects of the PD project when identifying and analyzing the change propagation patterns.

4.3.1 Direct parameter oriented change propagation patterns

Through the relations between two related direct parameters (denoted as "Upstream Direct Parameter" and "Downstream Direct Parameter"), the impact of the alteration introduced into Upstream Direct Parameter can be transferred into Downstream Direct Parameter in form of an alteration according to the conceptual model. Depending on the distribution of the three parameter value domains (i.e., tolerance, robustness and invalid domains), the above identified consequences are combined and suggest a set of conditions which are described in Figure 4.13.

1. Upstream Alteration Downstream Alteration (UADA)
2. Upstream Alteration Downstream Change (UADC)
3. Upstream Alteration Downstream Dysfunction (UADD)
4. Upstream Change Downstream Alteration (UCDA)
5. Upstream Change Downstream Change (UCDC)
6. Upstream Change Downstream Dysfunction (UCDD)
7. Upstream Dysfunction Downstream Alteration (UDDA)
8. Upstream Dysfunction Downstream Change (UDDC)
9. Upstream Dysfunction Downstream Dysfunction (UDDD)

The above nine conditions of the consequence from introducing alterations into direct parameters directly suggest six change propagation patterns between the involved direct parameters except the conditions (UDDA, UDDC and UDDD) in which the dysfunctional changes occur in the upstream direct parameters. As introduced in section 4.2.2, the dysfunctional change occurrence infer that the shifted parameter value has been invalid and can not be treated any further, and thus it is irrational that the dysfunctional change should be taken account of in the further change propagations and they are eliminated (highlighted by the grey shadow in Figure 4.13) when educing the further change propagation in our research.

4.3.2 Building block oriented change propagation patterns

Based on the above six change propagation patterns, we extend to analyze the change propagations between building blocks with taking the *effort-based relations* (i.e., generation and contribution relations) into account. As we introduced in section 3.5.1, the mutual relations exist between the building blocks from the same knowledge area

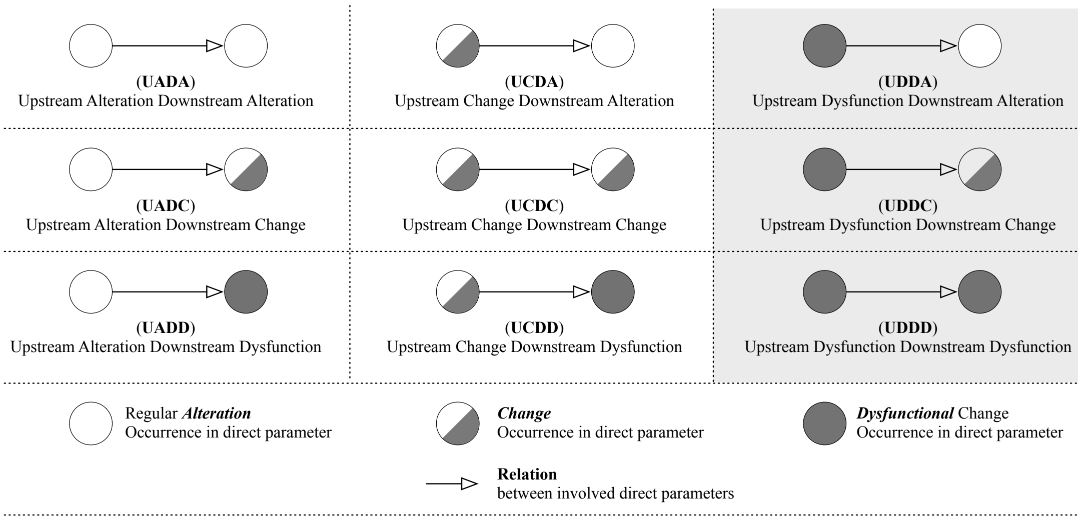

(UADA)
Upstream Alteration Downstream Alteration
(UCDA)
Upstream Change Downstream Alteration
(UDDA)
Upstream Dysfunction Downstream Alteration
(UADC)
Upstream Alteration Downstream Change
(UCDC)
Upstream Change Downstream Change
(UDDC)
Upstream Dysfunction Downstream Change
(UADD)
Upstream Alteration Downstream Dysfunction
(UCDD)
Upstream Change Downstream Dysfunction
(UDDD)
Upstream Dysfunction Downstream Dysfunction
Regular Alteration Occurrence in direct parameter
Change Occurrence in direct parameter
Dysfunctional Change Occurrence in direct parameter
Relation between involved direct parameters

FIGURE 4.13 – Conditions of combining consequences

besides the *non-effort relations* between the direct parameters belonging to the building blocks from the same knowledge area or the different ones. These two types of relations are specified as *change propagation channels* in our research. To identify the change propagation channels (effort-based mutual relations, and non-effort-based relations between the direct parameters belonging to building blocks) between the building blocks, we propose two indicators :

– *Trajectory* : According to the product evolution model in section 3.3.2 and the interactions between the knowledge areas mentioned in section 3.5, the relations (mutual relations, dependencies) between any pair of involved building blocks (denoted as upstream building block and downstream building block) are specified as directed and identified with the direction as *forward*, *feedback* and *vertical*. The first two directions are reflected by the mutual generation and contribution relations between the building blocks, whereas the third direction is reflected by the non-effort-based relations, such as the dependency between two geometric parameters (cf. the example mentioned in section 4.1.2). In further, we are enabled to categorize change propagations by investigating the trajectory as *effort-based change propagation* and *non-effort-based change propagation*. The former refers to the change propagations transferred through effort-based relations in either forward or feedback direction, and the latter refers to those through non-effort-based relations in vertical direction.
– *Involved building block* : Given the various directions and multiple types of the relations between building blocks, we investigate all the potential involved relations of one building block and categorize them as *inward* and *outward*. With

respect to the relation chains mentioned in section 3.5.1, we identify four scenarios where a building block could be connected by change propagation channels, i.e., One-in-One-out (OO), One-in-Many-out (OM), Many-in-One-out (MO) and Many-in-Many-out (MM) scenarios.

With the above indicators, we then identify the building block oriented change propagation patterns based on the direct parameter oriented ones, during which we extend to investigate both the effort-based and the non-effort-based relations reflected along the PD project. The building block oriented change propagation patterns are categorized into two group corresponding to the trajectory, and in each group the four scenarios are used to specify each pattern.

Among the building blocks, the one where the change propagation starts is identified as the *initial* building block and denoted as BB_i, i.e., the start point of change propagation, and the building block where the change propagation ends is identified as the *final* one and denoted as BB_f. Referring to the other building blocks along the change propagation, we identify them as the *intermediate* building blocks with denoting them as BB_m.

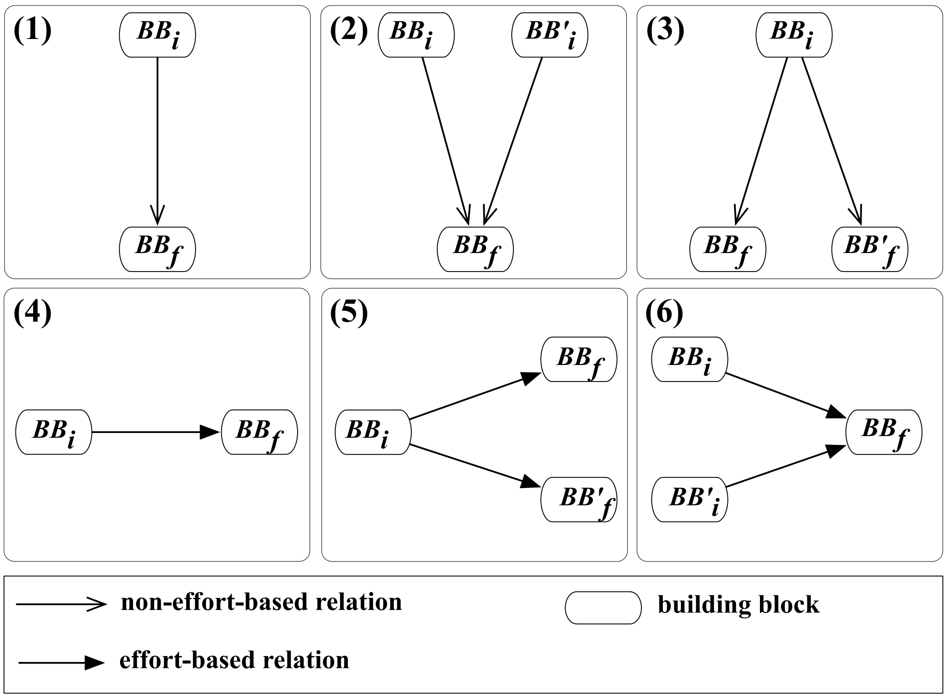


FIGURE 4.14 – Effort-based change propagation patterns (1, 2, 3) and non-effort-based change propagation patterns (4, 5, 6)

In Figure 4.14, we present both effort-based and non-effort-based change propagation patterns by illustrating the initial building block (denoted as BB_i), the final building block (denoted as BB_f) and the dependencies/relations between them. The (1) and (4) patterns in Figure 4.14 illustrate the meta building block oriented change propagation patterns since they represent the effort-based and the non-effort-based change propagations respectively. With combining the above two meta patterns, the rest patterns (i.e., (2), (3), (5), (6) patterns in Figure 4.14) are educed which cover the other more complex effort-based and non-effort-based change propagation scenarios.

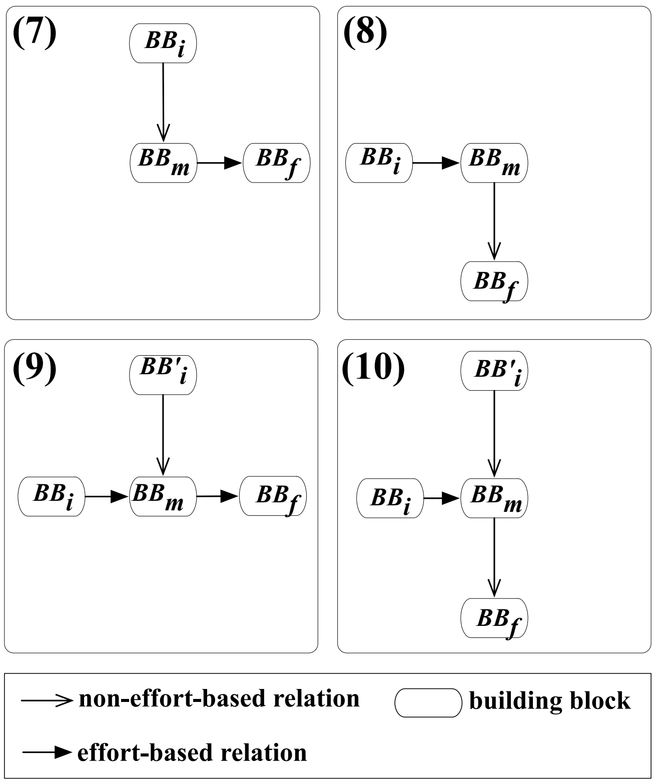


FIGURE 4.15 – Hybrid-channel change propagation patterns (7, 8, 9, 10)

Then we mix the effort-based and the non-effort-based change propagation channels and educe the hybrid change propagation patterns.

In Figure 4.15, the change propagation patterns consist of initial building block (denoted as BB_i), the intermediate building block (denoted as BB_m), the final ones (denoted as BB_f) and the simultaneous effort-based and non-effort-based change propagation channels between them. In the (7) and (8) patterns, the basic situations of hybrid-channel change propagation patterns are illustrated, and then the more complex situations are introduced by the rest patterns (i.e., (9), (10) patterns).

Given the above ten patterns, we educe the following practices supporting to recognize the mechanism of change propagation and establish the change process. The practices are composed relying on the five attributes proposed by (Fricke *et al.*, 2000) for indicating the tendency of better change management.

1. Range : change propagation should be controlled within a smaller range, i.e., the fewer building blocks and direct parameters are involved, the better. Given a change propagation channel, the final and intermediate building blocks along it should be paid more attention to for handling changes in order to eliminate the further propagated changes.
2. Timing : along the project timeline, the building blocks (i.e., initial and intermediate building blocks) in which changes occur earlier should be prepared with some plans of coping with change to prevent the potential change propagation and limit the further cost.
3. Critical building block : An intermediate building block that is propagated with the changes from both effort-based relation and non-effort-based relation is considered as a critical building block (see BB_m in pattern 9 and 10 of Figure 4.15). Within the given change propagation, those critical building block require more efforts in order to handle the difficulty and cost of coping with change.
4. Multiple mapping : given the multiple mapping in change propagation, the building block which either receives or transform multiple dependencies/relations should be given with the priority of coping with changes. According to the patterns stating multiple mapping, the building blocks connecting with others through multiple mapping relations should be given the priority of coping with changes.
5. Change propagation pattern : the perceived change propagations can be treated and converted into the combination of the patterns.

4.3.3 Construct network

According to the COCA framework, the mutual effort-based relations enable us to track and trace constructs in accordance with PD project timeline. As the change propagation channels, the generation relations enable us to discover and predict some potential changes caused by change propagation in the later phase(s) along the evolvement of PD project. Meanwhile, the contribution relations enable us to investigate some initial change(s) occurring in the earlier phase(s) of project and causing the current change through the change propagation.

In our research, a construct is specified as the “artefact” through aggregating the building blocks from the three knowledge areas respectively, and each building block is characterized by a set of direct parameters. Therefore, the connections between constructs are embodied by the relations between their owned building blocks. Taking the connections between the constructs as edges and the constructs themselves as nodes, we are enabled to build up a construct network in which change propagations are investigated and analyzed.

In Figure 4.16, a construct network structured by aligning to the product evolution process is illustrated. In the network, three types of change propagations are identified corresponding to the categorized change propagation channels. Then we extend to consider the constraints from temporal aspect, i.e., the freeze specifying the end point after which any alteration can not introduced.

4.4 Chapter Summary

Corresponding to the highlighted knowledge areas in the COCA framework in Chapter 3, each construct is composed of three respective building blocks. Each building block emerges a set of attributes which are characterized by the direct parameters. It is the direct parameter that the alterations fire the shifting effect to. Based on those considerations, we proposed the conceptual model through investigating the direct parameters, which is used to identify change and change propagation. In order to improve the conceptual model, we discussed the direct parameters by identifying their characteristics and the domains of their values. To achieve this purpose, we turned to identify and analyze the real data used by the focal company, i.e., the data format and type of the direct parameters. Furthermore, we considered boundary values of the direct parameters by taking the five types of expectations into account. According to the conceptual model of change propagation, the phenomena of change propagation can be perceived between two related direct parameters, building blocks and constructs. In this case, we proposed a hierarchical method of identifying and analyzing change propagations patterns. In another word, we described the patterns at three different levels, i.e., direct parameter oriented change propagation patterns, building block oriented change propagation patterns, and construct network of change propagation.

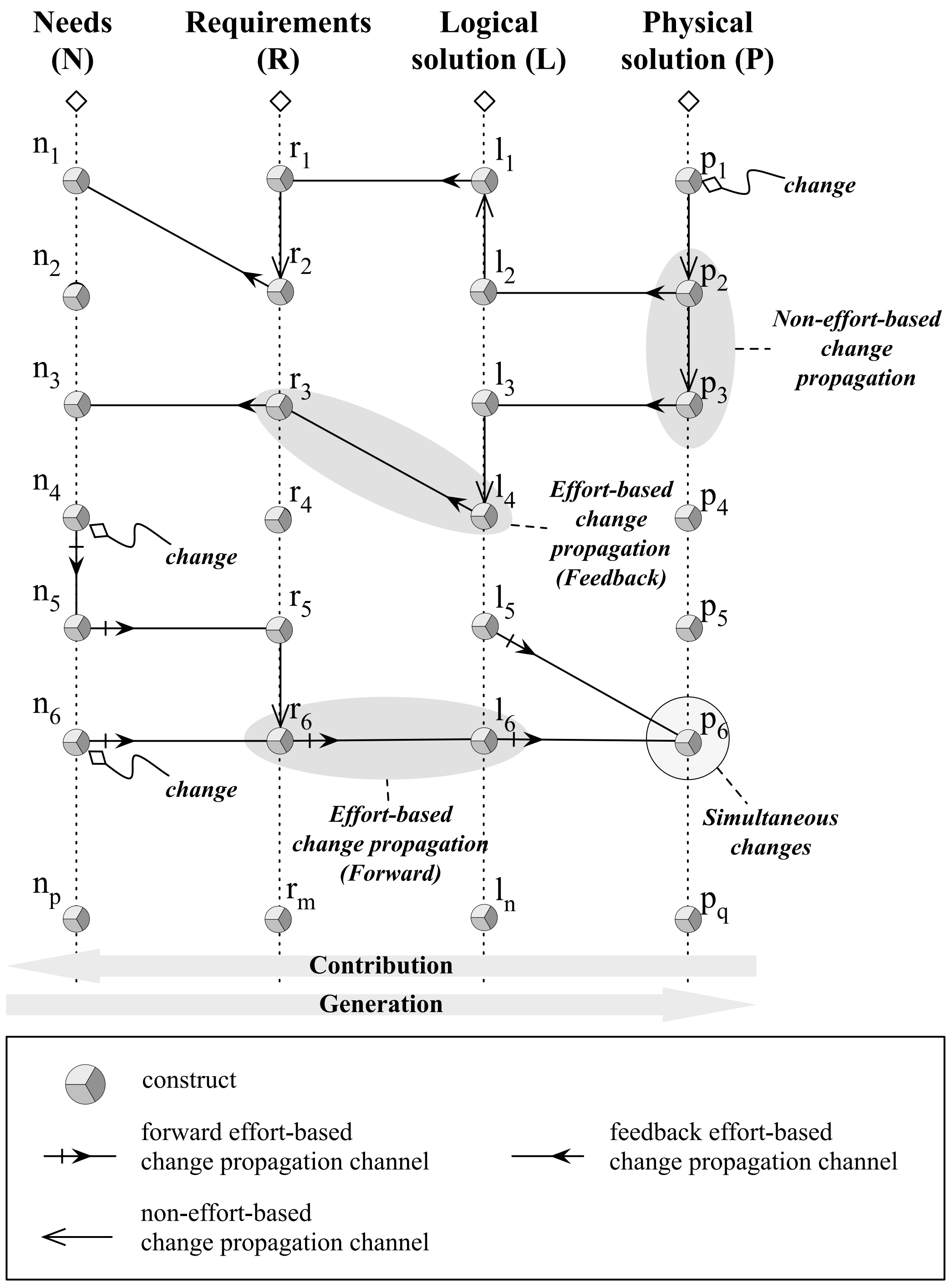


FIGURE 4.16 – Construct network

5 Simulating Change Propagations

▷ *In this chapter, we propose the methodology of simulating change propagations including the objectives, coping strategies and representations. We build up a network in which the constructs are treated as nodes and the relations between them as edges. In the network, we identify and analyze changes and change propagations according to a proposed conceptual models of change occurrence and change propagation. Then we discuss the agent-based technical solution used to simulate change propagations implemented on a computer. The system architecture is first proposed, which is followed by an introduction to agent-based system framework named JADE. JADE is an open source environment enabling us to develop agent-based system by Java language. Then we explain how to configure the agent by introducing the life-cycle of an agent, communications between agents and the agents of different roles. At last, we present the data structures that serialize the agents.* ◁

Plan of Chapter

With the modeled constructs and their relationships (i.e., the effort-based, the non-effort-based relations), a change propagation network is established, in which constructs are designated as the nodes and their relationships are the edges. By adopting the conceptual models of change occurrence and change propagation, an alteration introduced into one construct can be identified and analyzed to determine whether a change is occurring or not. If a change occurs, its influence can be transferred to other node(s) (i.e., other construct(s)) through the edge(s) (i.e., the relation(s) between constructs).

5.1 Architecture of construct network

As we have proposed in the previous chapter, the relationships between the constructs are embodied and categorized into the *effort-based* and the *non-effort-based* relations. Referring to the effort-based relations, their directions are classified as *forward* and *feedback* corresponding to the product evolution directions (i.e., generation and contribution). Compared with that, the non-effort-based relations are all of *vertical* direction. The relations as well as the linked constructs compose a network, i.e., the construct network, and reflect a global model of the PD project. In the network, the project progress reflected by the product evolution, the project execution and the partner participation are represented by the effort-based relations and their linked constructs. Meanwhile, the other parameter couplings are represented by the non-effort-based relations. All the above relations work as the change propagation channels between the constructs. Along the change propagation channels, the involved constructs, on one hand, receive the alterations causing the change occurrence from some of the connected channels, and on the other hand they propagate the influence of the occurred change to the other related ones through other channels.

We then analyze the nodes and the edges in order to identify and study the characteristics of the change propagations in it. Relying on that, we elaborate the rationale of simulation.

5.1.1 Nodes of construct network

As we mentioned in section 3.6, a construct is an artefact obtained through aggregating the building blocks from the three knowledge areas respectively. Moreover, each building block is characterized by a set of direct parameters. In other words, the KID of each construct is organized in a hierarchical way (see Figure 5.1).

Relying on the hierarchical structure of constructs, we are enabled to treat the change propagations among the constructs as the multi-layer communication. According to the construct structure, we build up three communication layers through encapsulating their internal behaviours respectively. During simulating change propaga-

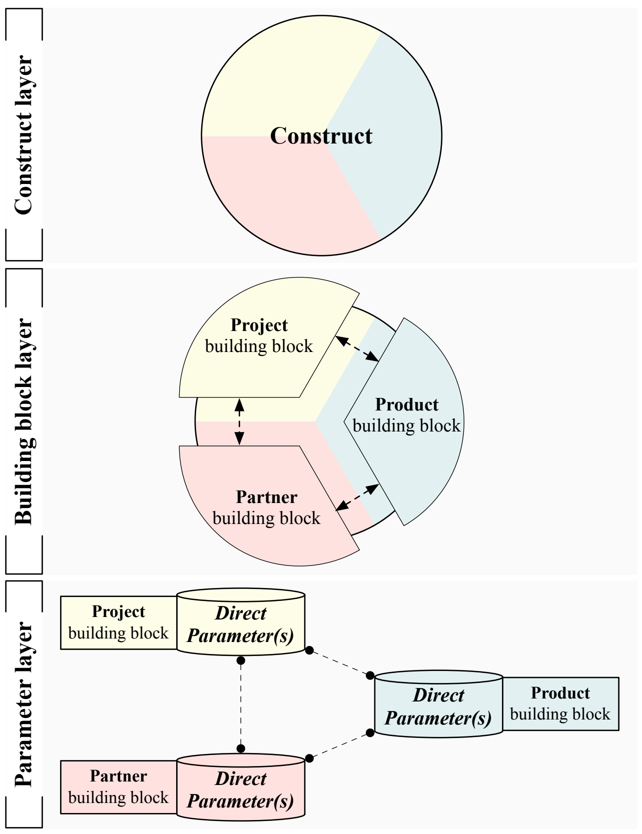


FIGURE 5.1 – Hierarchical structure of construct

tions, each layer is able to either provide services to the others or call support from the others (also see Figure 5.1).

The layers are described as follows :

1. *Construct layer* : this is the scope within which the constructs receive alterations, forward the alterations to other procedures and determine the response according to the preset plans. The introduced alterations into one construct would be treated in local to be qualified as causing the consequences (i.e., regular alteration, change occurrence, dysfunctional change occurrence), and they are quantified to invoke the change propagations to other constructs.
2. *Building block layer* : this layer is the scope within which the building blocks search the involved direct parameters in term of the query from their located construct(s) and look up the relations (cf. Table 3.7 in section 3.6) for supporting to execute the potential change propagations.

3. *Parameter layer* : this layer is the scope where the alterations introduced into the involved direct parameters are treated according to the conceptual model of change occurrence and propagation. In this layer, the relations between building blocks are embodied as the dependencies between the direct parameters.

During the simulation of change propagations in the construct network, each of the multiple layers is responsible for its own tasks.

The constructs are the portals which the alterations are introduced into, and they are incorporated with the plans of treating the alteration to determine the corresponding actions and communicate with each other in the various conditions during the simulation.

The building blocks of the constructs are corresponding to the three knowledge areas, and each of them preserve and maintain the relations with others (cf. Table 3.7 in section 3.6). According to the communication specifications given by the constructs, the involved building blocks select the right relations and call for the concerned direct parameters in which the alterations are treated.

The direct parameters are incorporated with the procedures of perceiving the consequence (i.e., change occurrence) due to the alterations, treating the change, and computing the influence of the changes during the simulation. From the viewpoint of the direct parameters, all the relations are the couplings that are formalized as the dependencies (cf. section 3.5.2).

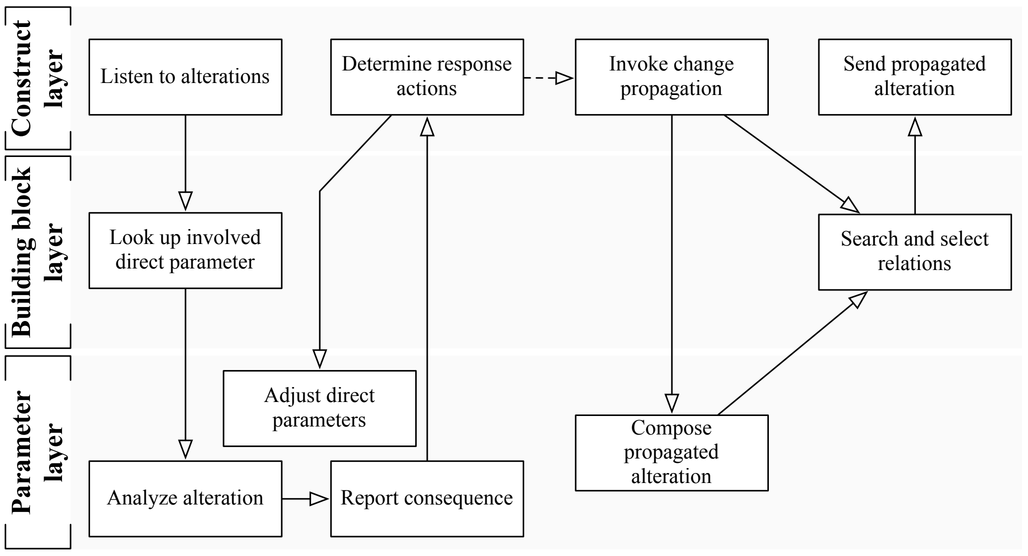

Construct layer
Building block layer
Parameter layer
Listen to alterations
Determine response actions
Invoke change propagation
Send propagated alteration
Look up involved direct parameter
Search and select relations
Adjust direct parameters
Compose propagated alteration
Analyze alteration
Report consequence

FIGURE 5.2 – Internal actions of construct

With the multiple layers, a construct is initially listening to any alteration within the construct network. For an incoming alteration, the construct makes query to the correspondence building block. Then the latter identifies the referred direct parameter specified in the alteration. The concerned direct parameter analyzes the alteration with the conceptual model of change occurrence and propagation and reports the consequence to the building block which also sends the outcome back to the construct. According to the preset plans, the construct determines the procedures of coping with the consequence from the introduced alteration. As one of the response actions, the construct invokes the change propagations to the other ones. The construct asks the involved building block to search and select the concerned relation(s) for the change propagation, and it composes another alteration to be propagated based on the computation by the direct parameter(s) at the same time. Through the found relations, the construct sends the propagated alteration to the destination ones (see Figure 5.2).

In the construct network, the constructs, as the nodes of the network, are interconnected by the connections for their communication during the simulation of change propagations. Corresponding to the multi-layer structure of constructs, their connections, i.e., the edges of the network, manifest themselves with the various functions.

5.1.2 Edges of construct network

The connections among the constructs, i.e., the edges of the construct network, provide the channels in which the influence of the occurred change are transferred and the messages are exchanged among the constructs. Corresponding to the multi-layer structure of the constructs, the connections between them are reflected by relations among the building blocks and are in further implemented by the dependencies among the direct parameters (see Figure 5.3).

In the Ph.D. research, we specially designate *connections* as the linkages among the constructs, *relations* among the building blocks and *dependencies* among the direct parameters.

As illustrated in Figure 5.3, between each pair of the building blocks of the same knowledge area but belonging to the different constructs, there exists either an effort-based or a non-effort-based relation (i.e., the *external building block relation* in Figure 5.3). The former relation can only be between the building blocks belonging to the product management knowledge area, whereas the latter relation can be between the ones belonging to any of the three knowledge areas (cf. section 3.6). At the same time, between the building blocks within the same building block, only non-effort-based relations (i.e., the *internal building block relations*) can be identified, and they reflect the aggregation of the KID along the PD project. The relations between the building blocks are ultimately implemented by the *internal* and *external direct parameter dependencies* between their owned direct parameters, during which the characteristics of the dependencies and the method of treating them are configured and determined in accordance with the types of

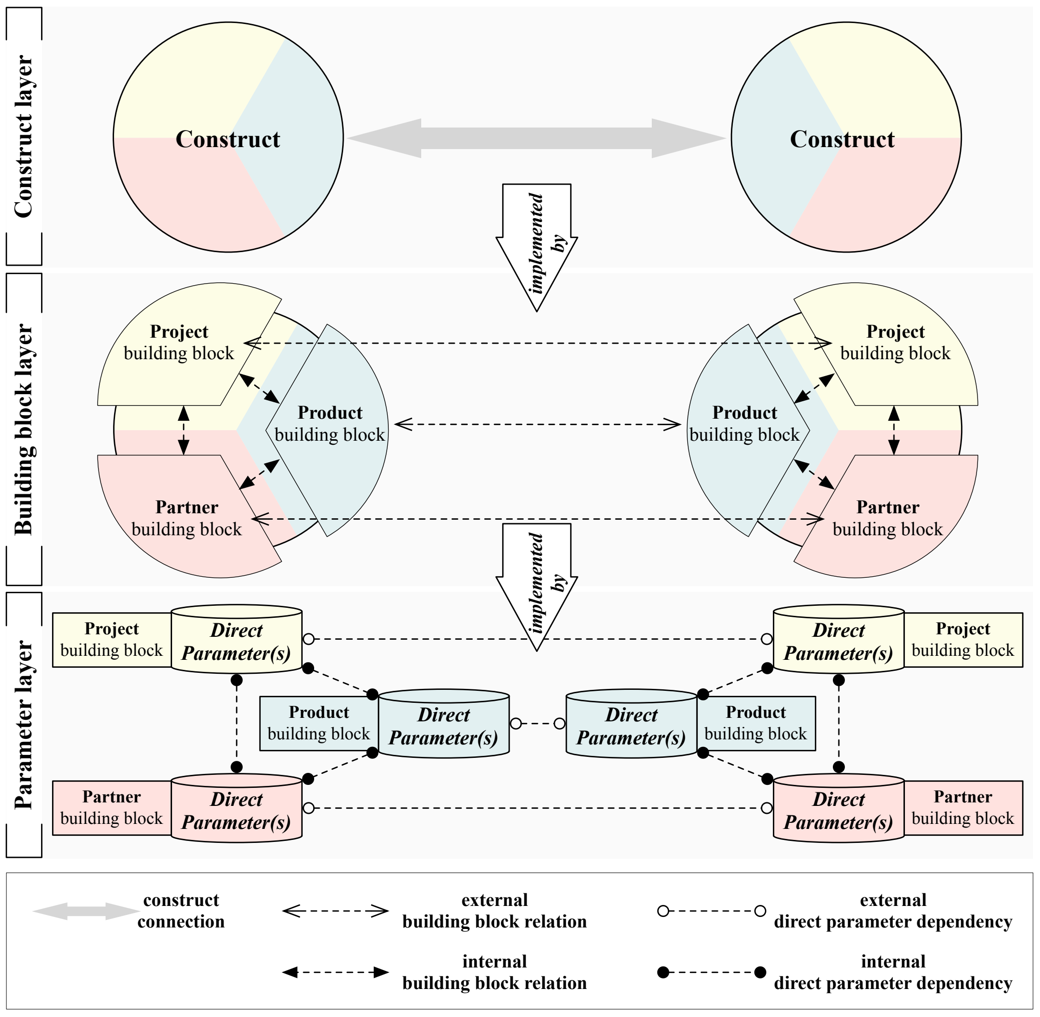


FIGURE 5.3 – Construct connections

building block relations (i.e., intra-knowledge area effort-based, intra-knowledge area non-effort-based, inter-knowledge area non-effort-based relations, also cf. section 3.6).

On one hand, the connections among the constructs encapsulate the characteristic data and functions with respect to the multiple layers which supports to handle the complexity of change propagation. On the other hand, the connections as well as the structure of the constructs are also used to define the structure of the messages exchanged among the constructs during the simulation.

5.2 Rationale of Simulation

So far, we have prepared and reached the prerequisites for the simulation of change propagations. The prerequisites are mainly described as follows :

1. Identify and model the PD project in which the simulation is proposed to imitate change propagations.
2. Describe and model the phenomena of change occurrence and change propagations.
3. Propose the method of identifying and analyzing change occurrence and change propagations.
4. Investigate the factors involved in analyzing change propagations, such as the characteristics of direct parameters.

Relying on the architecture of the construct network, we then consider simulating change propagations within the network. In our research, we propose to distinguish the alteration from the occurred change (see section 4.1). In this way, we are enabled to identify the particular event (i.e., an occurred change) out of the parameter tolerance. Therefore, our simulation solution is started with introducing the alterations. Through analyzing the alterations, the normal variations are filtered from the change occurrence. Referring to the occurred change, the impact due to it should be formulated according to the prescribed change management strategy and then be composed into another alteration(s) to invoke the change propagation. Meanwhile, the involved direct parameter(s) during the change occurrence is/are adjusted to represent accepting the consequence from the occurred change within the construct.

5.2.1 Introducing alteration

By studying the involved direct parameters where the alteration would be introduced (see section 4.2), we identify four potential effects the alteration could bring in the direct parameter values :

1. *Retain* : After introducing the alteration, the direct parameter value is not modified, i.e., the alteration does not shift the parameter value.
2. *Increase* : Given a direct parameter is assigned with the default quantitative value, if the alteration shifts the direct parameter value evolving towards the maximum boundary value, then the alteration is qualified as executing increasing behaviour.
3. *Decrease* : Oppositely, if the alteration shifts the direct parameter quantitative value evolving towards the minimum boundary value, then it is qualified as executing decreasing behaviour.

4. *Replace* : Specially, if the direct parameter is assigned with the value of enumeration data type and it is shifted to another enumeration value due to the alteration, then the alteration is qualified as executing replacing behaviour.

With the introduced alteration, the direct parameter value is shifted towards its preset boundary values (cf. section 4.2.2). If some dysfunctional change occurs, then it is unnecessary to continue with tracking the further consequence since the involved building block and even the construct can not be treated as the valid one. Thus, the corresponding building block as well as the construct are flagged for further treatment. However, at the very first attempt we do not analyze them. In other words, *one change propagation path would be identified as finish when a dysfunctional change is occurring*. Let us consider the simple example (cf. 3.4), the direct parameter of radius of Component B is designed as $7.5cm$ ideally. In the current example, it allows the variations around the ideal value, and both its tolerance and absolute domains are configured as $[5cm, 16cm]$. When an alteration that the radius parameter is increased by $7.5cm$ is introduced, i.e., the radius parameter is assigned with $15cm$, the alteration is identified as a regular alteration according to the conceptual model of change occurrence. Then through the change propagation from Component B to Component A, we could learn that there is an obvious dysfunctional change occurring in Component A since the Component A is not valid any further, i.e, it would not be around Component B as a whole one. In this case, it is unnecessary to consider any change propagation from the dysfunctional change within Component A.

In the construct network, a particular phenomenon that more than one alteration is introduced into one construct at the same time and cause the change occurrences can be perceived. We specify these specific alterations as the *simultaneous alterations* and the changes due to them as the *simultaneous changes*.

5.2.1.1 Simultaneous changes

The simultaneous alterations introduced into one construct could cause simultaneous changes that bring in the various possible consequences given the construct structure and the orders of treating the alterations.

The construct structure exposes the multiple direct parameters belonging to the multiple building blocks. According to the conceptual model of change occurrence, the simultaneous changes are caused by the simultaneous alterations are finally identified within the direct parameter(s). With the simultaneous changes and the direct parameter(s) they are occurring, we identify three basic conditions of change occurrence in which each alteration do cause one change occurring. Associating with the conditions, we will use the simple example in section 3.4 to strengthen the explanation. For convenience, we re-present the geometric specifications of the product as follows.

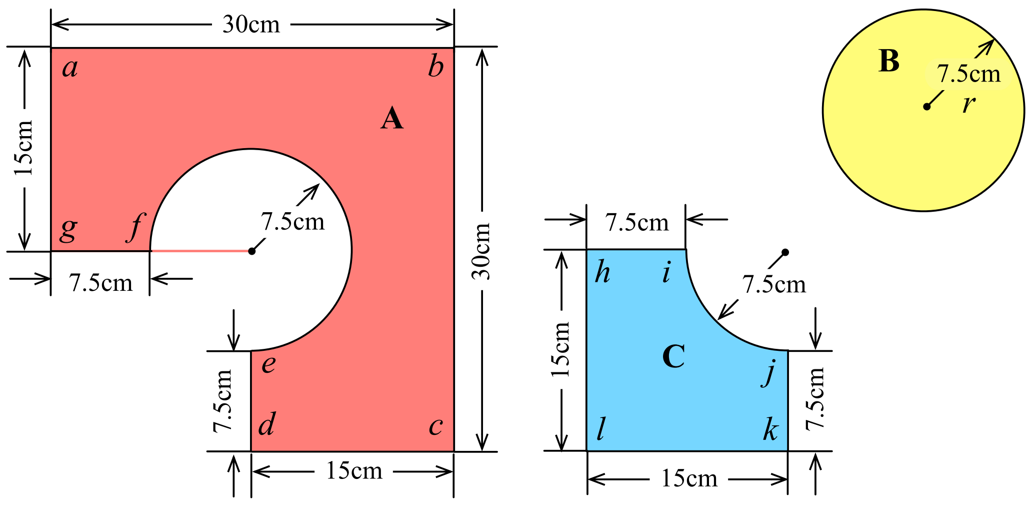


FIGURE 5.4 – Geometric specifications

1. *Multiple changes occurring in one direct parameter* (denoted as C_1)
   In this condition, multiple alterations are introduced into one direct parameter and cause the simultaneous changes at the same time. The order of introducing the alterations could cause the different consequences (see Figure 5.5).

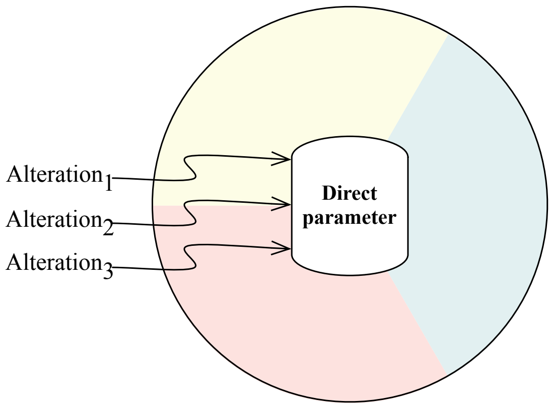


FIGURE 5.5 – Multiple alterations causing simultaneous changes in one direct parameter

In the simple example, there are two alterations being introduced into a direct parameter of Component A, i.e., the length of edge $< g, f >$ and causing two simultaneous changes. One change (denoted as $change_1$ due to $alteration_1$) makes the edge $< g, f >$ increased by $5cm$, and the other one (denoted as $change_2$ due to $alteration_2$) makes the edge $< g, f >$ decreased by $7.6cm$. Depending on the order of introducing the alterations, two consequences are presented. If $alteration_1$ is introduced first, then the consequence from the simultaneous changes would make the parameter value of the edge $< g, f >$ shifted to $12.5cm$ and then to $4.9cm$. However, if $alteration_2$ is introduced first, then the first caused change would make the parameter of the edge $< g, f >$ invalid, i.e., a dysfunctional change occurs.

To be noticed, although there are several possible orders of introducing the alterations, the consequences due to the simultaneous change could be the same. In the above example, if $alteration_2$ only make the edge $< g, f >$ decreased by $6cm$, i.e, it would not cause dysfunctional change, then the two possible orders of introducing the alterations could cause the same final consequence.

2. *Multiple changes occurring in multiple independent direct parameters* (denoted as C_2)

   Each of the multiple alterations is introduced into each of the direct parameters which do not have any dependency. The consequence from the occurred changes due to introducing the alterations is unique, i.e., the order of treating the alterations does not affect the consequence (see Figure 5.6).

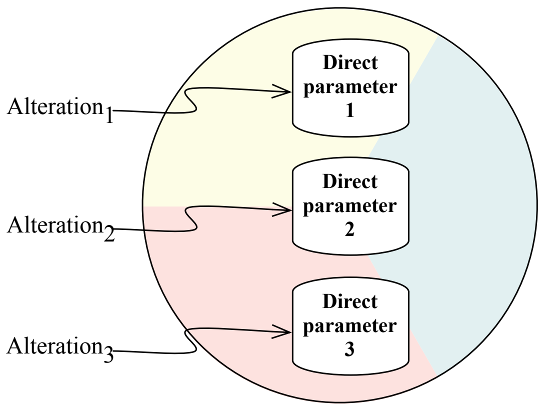


FIGURE 5.6 – Multiple alterations causing simultaneous changes in multiple independent direct parameters

Still in the simple example, the edge $< a, b >$ and the edge $< b, c >$ are with the independent length parameters. The order of introducing the two respective alterations to these two parameters would have the same consequence.

3. *Multiple changes occurring in multiple dependent direct parameters* (denoted as C_3)

   Each of the multiple alterations is introduced into each of the direct parameters which are related by the dependencies organized as a acyclic graph. In this condition, the order of treating the alterations as the changes as well as the dependency specifications would affect the consequences (see Figure 5.7), i.e, the dependency would bring in another alteration to the destination direct parameter. The reasons for eliminating the loops in the dependency graph is that any loop in the graph might cause a resonance-like effect. The effect could amplify the influence of occurred changes during the change propagations and also reflect some too-high couplings existing the elements of the PD project.

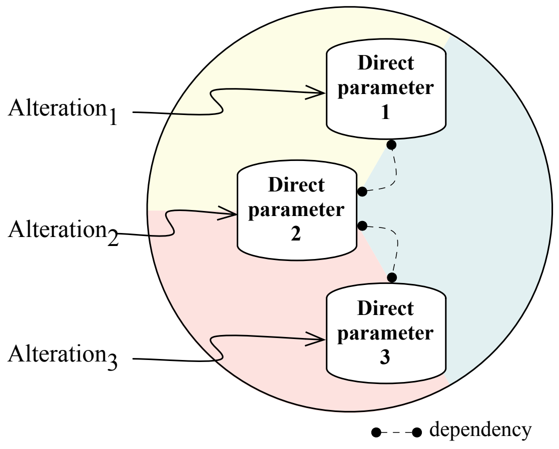


FIGURE 5.7 – Multiple alterations causing simultaneous changes in multiple dependent direct parameters

Let us still use the length parameters of the edge $< a, b >$ and the edge $< b, c >$ in the simple example, but we make them dependent by adding a constraint that the area value of Component A is a constant (i.e., $667.04cm^2$). Resulting from that, if the length value of the edge $< a, b >$ is increased, then the length value of the edge $< b, c >$ is decreased correspondingly and verse vice. Suppose now that one change (denoted as $change'_1$ due to $alteration'_1$) makes the edge $< a, b >$ increased by $3cm$, and the other one (denoted as $change'_2$ due to $alteration'_2$) makes the edge $< b, c >$ decreased by $4cm$.

If $alteration'_1$ is introduced first, then the consequence from the simultaneous changes would make the parameter value of the edge $< a, b >$ shifted to $33cm$. Due to the first alteration, another alteration through the dependency between the direct parameters would introduced into the edge $< b, c >$ and cause a change shifting its parameter value to $27.27cm$, i.e, decreased by $2.73cm$. However, due to the introduced $alteration'_2$, the edge $< b, c >$ should be decreased by $4cm$. In this condition, some decision should be made by the simulation user.

5.2.1.2 Treating simultaneous changes

In consideration of the characteristics of the above three conditions, we suggest a set of simulation strategies.

For the simultaneous alterations in the first condition, i.e., C_1 : multiple changes occurring in one direct parameter, we treat them with the full permutation ordering algorithm and then introduce each of them into the direct parameter by turns. Given the number of the simultaneous alterations as n_a, we execute the simulations for $n!$ times.

For the alterations in the second condition, i.e., C_2 : multiple changes occurring in multiple independent direct parameters, it is not necessary to order them since the consequences caused by each change would not affect each other. In this condition, we select each of the alteration by chance and introduce them one by one into the corresponding direct parameter.

For the alterations in the third condition, i.e., C_3 : multiple changes occurring in multiple dependent direct parameters, we use the same ordering algorithm as that in the first condition to treat the changes. The number of the solutions to introduce the alterations is also calculated by the factorial operations onto the number of the alterations. Referring to the potential conflicts between the extra alteration(s) brought in through the dependency(s) due to the introduced one(s) and the alteration(s) to be led in, the decision is left to the simulation user to make.

Referring to the conditions combined by the above three ones, we treat them as follows.

1. In the combined condition, the involved direct parameters which do not have any dependency with each other are firstly selected as one group (denoted as "independent direct parameter group"). In the group, if C_2 condition is identified, then the C_2 condition is treated through introducing the concerned alteration(s) into the direct parameter(s).
2. After treating the C_2 condition in the independent direct parameter group, if C_1 condition is identified, then the C_1 condition is treated.
3. In the combined condition, the involved direct parameter which are connected by the dependencies are selected as one group (denoted as "dependent direct parameter group"). In this group, if C_1 condition is identified, then its concerning alterations are ordered by the strategy preset in the C_1 condition. If there is no C_3 condition being identified in the group, then the identified C_1 condition is treated.
4. If C_3 condition is identified after ordering the alterations concerned in the C_1 condition, then the ordered alterations concerned in the C_1 condition are combined together with the ones concerned in C_3 condition. Then the alterations are treated by the preset strategy in the C_3 condition.

5.2.2 Invoking change propagation

Between the constructs, the phenomena of change propagations reflect the global consequence from introducing alterations into the PD project. The change propagations are modeled and identified with respect to the knowledge areas as well as the relations between the involved building blocks. In this way, the change propagation phenomena are analyzed according to the specific management issues and the characteristics of change propagation channels. In further, the change propagations are regarded as the chained consequences of introducing alterations into the direct parameters which are interrelated to each other with dependencies.

Relying on the simulation strategies of change occurrence, we extends to study simulating the change propagations between two dependent direct parameters (denoted as *upstream direct parameter* and *downstream direct parameter* in accordance with the direction of their dependency). It mainly concerns two issues :

1. Between the two dependent direct parameters, identifying and determining the possible propagation scenario(s) for the simulation, i.e., creating the propagation plans.
2. Relying on the identified propagation scenarios and the structure of constructs, proposing the procedures of simulating the change propagations between two constructs.

5.2.2.1 Propagation plans

With the simulation strategies of introducing alterations, we firstly study the change occurrence within the upstream and the downstream direct parameter respectively, and then combine the change occurrences in both the direct parameters to propose the propagation scenario(s).

Referring to all the alterations into each direct parameter, if they arrive at the direct parameter by turns instead of being introduced simultaneously, then they are just treated one by one at their arrival order which means there only exists one simulation scenario. Otherwise, the alterations are identified as the simultaneous ones which means there exist a number of possible orders of introducing the alteration (cf. section 5.2.1.2). Therefore, we focus on investigating the change propagations in which the simultaneous changes occur in the upstream and downstream direct parameters.

As illustrated in Figure 5.8, the upstream direct parameter receives the potential alterations in two ways. First, some alterations are introduced through the inward dependency(s) (the number of the inward dependencies is denoted as N^u_{dep}) to the upstream direct parameter (the number of the alterations brought in by one of the inward dependencies is denoted as n^u_i and $i \in [1, N^u_{dep}]$ refers to the identity of the dependency, i.e., $dependency_i$). Second, some other alteration can be introduced into the direct parameter directly (the number of the direct introduced alterations is denoted as n^u_{alt}). Then the number of simulation solutions to introduce all the alterations into the upstream direct parameter (denoted as $S^u_{alteration}$) is calculated as follows.

$$S^u_{alteration} = (\sum_{i=1}^{N^u_{dep}} n^u_i + n^u_{alt})!$$

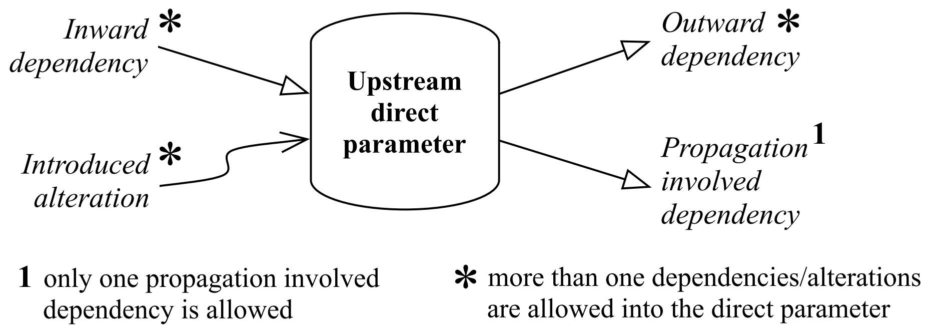

Inward dependency *
Introduced alteration *
Upstream direct parameter
Outward dependency *
Propagation involved dependency 1
1 only one propagation involved dependency is allowed
* more than one dependencies/alterations are allowed into the direct parameter

FIGURE 5.8 – Alterations and dependencies in upstream direct parameter

According to the simulation solutions, all the alterations are introduced into the upstream direct parameters and then the consequences are transferred through the outward dependency(s) and the propagation involved dependency.

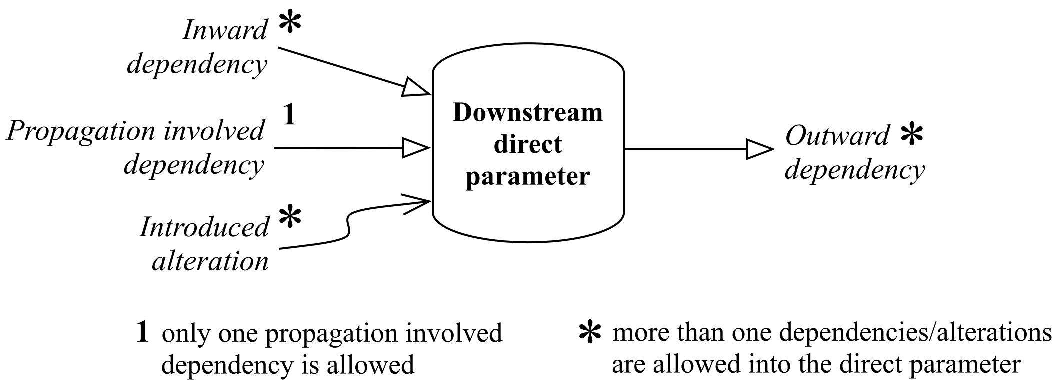

Inward dependency *
Propagation involved dependency 1
Introduced alteration *
Downstream direct parameter
Outward dependency *
1 only one propagation involved dependency is allowed
* more than one dependencies/alterations are allowed into the direct parameter

FIGURE 5.9 – Alterations and dependencies in downstream direct parameter

As the upstream direct parameter does, the downstream direct parameter receives some alterations from its inward dependency(s) (the number of the inward dependencies is denoted as N^d_{dep} and the number of the alterations brought in by one of the inward dependencies is denoted as n^d_j and $j \in [1, N^d_{dep}]$ refers to the identity of the dependency, i.e., $dependency_j$) and/or some other ones are introduced directly (the number of the direct introduced alterations is denoted as n^d_{alt}). Besides these two ways, the downstream direct parameter also receives the alterations through the dependency with the upstream direct parameter (see Figure 5.9).

Through combining both the direct parameters connected by the propagation involved dependency, we calculate the total number of simulation solutions to introduce all the received alterations (denoted as $S^d_{alteration}$) as follows.

$$S^d_{alteration} = (\sum_{j=1}^{N^d_{dep}} n^d_j + n^d_{alt} + S^u_{alteration})!$$

which could be treated as follows in further.

$$S^d_{alteration} = (\sum_{j=1}^{N^d_{dep}} n^d_j + \sum_{i=1}^{N^u_{dep}} n^u_i + n^d_{alt} + n^u_{alt})!$$

5.2.2.2 Procedures of propagation

Through the above section, we identified the possibilities of the change propagations between two direct parameters under the various conditions and calculated the number of the simulation solutions. Then relying on the multi-layer structure of constructs, we take the characteristic issues from the layers into account and then propose the detailed simulation procedures. With incorporating the simulation procedures, an introduced alteration into each construct in the network could be responded in order to imitate the possible chained consequences, i.e., simulating change propagations.

Corresponding to the top-down viewpoint onto the multi-layer structure of construct, we identify the characteristic issues as follows :

1. Freezes : In the construct network, the constructs are incorporated with the freezes in order to cover the time and/or the cost management issue. The freezes are categorized into three classes that are mapping to the aggreating levels of the KID of the PD project, i.e., construct, building block, direct parameter.
   The construct freezes are mainly used to indicate the project progress. Once a construct is frozen, it will be eliminated from the network and never participate the simulation.
   The building block freezes present the specific time-based constraints according to the management issues from the knowledge areas. When a building block is frozen, its owned relations to the other ones are cut off. In this way, the building block will neither accept nor generate any alteration.
   The direct parameter freezes are used to lock the parameter value to indicate the progress of the procedures during executing a task. When a direct parameter is frozen, its upstream dependent direct parameter(s) would be constrained and prevented from being changed.
2. Consequences : As the conceptual model of change occurrence and change propagation, there are three possible consequences (cf. section 4.2.2). When a dysfunctional change occurs, it is meant that the corresponding direct parameter is

invalid and the involved building block would manifest itself with invalid performance and should not be adopted by the construct any further. Moreover, the change propagation(s) taking the dysfunctional change as the initial one is/are also invalid. Therefore, a set of regulations are created in order to eliminate the invalid simulation solutions.

If the alteration would cause regular alteration or change, then it is accepted and the corresponding direct parameter is modified according to the introduced alteration specifies. Otherwise, i.e., a dysfunctional change is identified, the corresponding direct parameter value would not be modified. Then the current simulation solution is terminated and the aborting event is recorded. If there are still other simulation solution(s) is/are waiting, the next simulation solution is started.

5.3 Agent-based System Architecture

Due to the topology of the relations between the constructs, some of change propagations can be executed with more than one solution in order to simulate all the possible consequences of change occurrence. To simulate the change propagation effectively and efficiently, all the possible change occurrence and the propagations should not only be planned but some communications between the involved constructs should be executed autonomously. The communications enable to achieve the co-decision-making in order to simulate the change propagations that could be composed in multiple solutions.

To satisfy the above requirements for simulating all the possible change propagations within the network, we turn to the agent-based technical solution to model the change propagation network and the behaviour of constructs. By means of agent-based technical solution, each construct is modelled as one agent and the deployed procedures in it are modelled and designed as the behaviours of the agent. Through the relations between the constructs, the agents can communicate with each during simulating change propagations.

Looking for keeping the chapter understandable for non-expert readers, we will provide only those concepts necessary for understanding the core idea of agent-based technical solution for simulation.

To deploy the agent-based technical solution, we compose the requirements to the agent-based system. To implement the simulation methodology, we identify several agent classes which are deployed to perform their different behaviours. Meanwhile, each class of agent is designed with the multiple states along their life cycle. These states can be transformed according to the potential situations. Among the agents, we model their interactions with specifying the concerned agent behaviours as well as the communicating messages.

The simulation solution is constructed in a multi-layer architecture that contains four layers. The top layer is logically closer to the visual presentation (i.e., Presentation Layer), whereas the bottom layer is logically closer to the data abstraction (i.e., Data Layer).

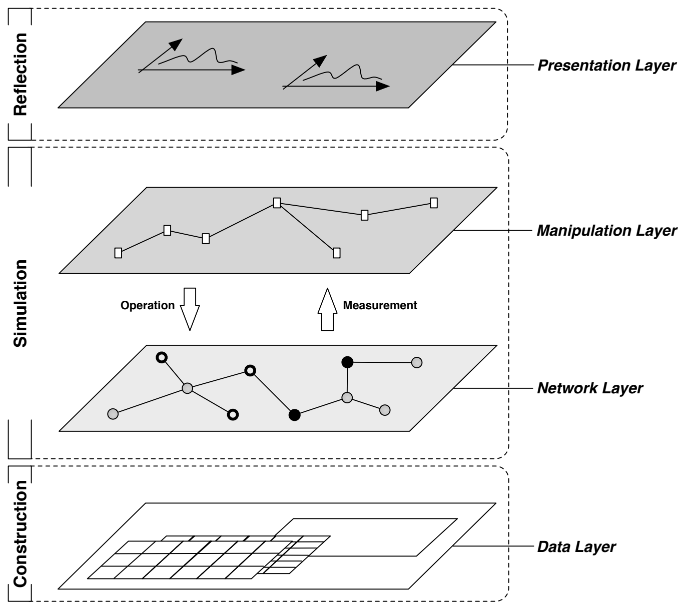


FIGURE 5.10 – Multi-layer system architecture

1. Data Layer : This layer provides the structured parameters specifying the decomposition of PD project to the overlying layer and constructs the perspective of resolving PD project.
2. Network Layer : This layer receives the structured parameters from the underlying layer and defines a network in which changes and change propagations could be invoked and perceived, and it provides the measurement data to the overlying layer to report the current state. When performing the simulation, the data encapsulated in this layer is operated by the overlying layer according to the simulating method.
3. Manipulation Layer : This layer sends the configuration parameters to the overlying layer for presenting the outcome of simulation, and meanwhile it receives the measurement data from the underlying layer to perceive the evolvement of project. During the simulation, this layer operates the data from the underlying layer according to the simulating method.
4. Presentation Layer : This layer receives the configuration parameters of visual presentation from the underlying layer and reflects the continuous outcome of simulation with graphical views.

As composing simulation function, manipulation layer consists of agents and the channels between them. The agents communicate with each other through exchanging messages through the channels. Network layer is composed with constructs and the couplings between them. According to the decomposition of PD project, the constructs within network layer represent the end elements belonged to each of the three knowledge areas (i.e., product, project and partner) and refers to components, tasks and actors. Then the couplings between the constructs can be categorized into internal couplings that state the interrelations between the constructs from the same knowledge area and external couplings indicating the interrelations between the constructs belonging to different knowledge areas (see Figure 5.10). During simulation procedure, the agents within the manipulation layer receives the measurements from network layer to acquire the status of construct network, and meanwhile they interact with each other to determine the operations made onto the constructs. Through the bidirectional communication, the agents in the manipulation layer keep controlling all the constructs and deduce the potential outcomes due to the occurrence of changes.

5.4 Agent-based System Framework : JADE

In this Ph.D. research, we selected an open source software called "JADE[1]" to implement the agent-based system under the the LGPL license[2]. "JADE" is the abbreviation of "Java Agent DEvelopment Framework", and it enables us to develop agent-based system with a simplified software framework implemented by Java language. Meanwhile, JADE complies with the FIPA (Foundation for Intelligent Physical Agents) specifications[3] and includes all the mandatory components designated in the FIPA specifications. The agents created by JADE adopt the full FIPA communication model to exchange information with each other with the FIPA ACL (Agent Communication Language) messages.

The software architecture of JADE is based on the coexistence of multiple Java Virtual Machines (JVM), and the communication between the JVMs is implemented with Java Remote Method Invocation (RMI). Each JVM works as a container of agents to provide an runtime environment for agent execution, and it also enables multiple agents to execute on the same host at the same time.

For an agent, the actual action it takes is executed by some "behaviours". JADE allows the programmers to extends the pre-defined "Behaviour" class for implementing the procedure of carrying out a task by one agent. The extended behaviour can be added to the agent and started at any time, for example during configuring the agent, or when the agent is executing some other behaviour. Moreover, multiple behaviours can

1. The website URL is : http ://jade.cselt.it
2. i.e., Lesser General Public License Version 2
3. Access by http ://www.fipa.org/specifications/

be executed by one agent concurrently in a cooperative scheduling mechanism. JADE provides three particular types of behaviours with various execution modes.

– One-Shot Behaviour : the behaviour executes its defined action only for once then completes immediately. The behaviour can be implemented by extending the "OneShotBehaviour" class.
– Cyclic Behaviour : the behaviour executes the same action time after time and never complete and it can be implemented by extending the "CyclicBehaviour" class.
– Generic Behaviour : the behaviour determines to execute its action depending on the pre-embedded status. The behaviour can be completed when a specific condition is met. The behaviour can be implemented by extending the "Behaviour" class.

Based on the multiple types of behaviours, JADE enables to combine the behaviours to build up more complex ones for various objectives.

5.5 Agent Configuration

To simulate the change propagations in the construct network, each agent is constructed with a set of plans under the potential scenarios during its life cycle. With respect to the FIPA standard [2002], we categorize four states during the life cycle of the agents.

– Initiated : The agent is created and to be registered with the container, and it is waiting for assigning with its name and address used for the further communication with others.
– Active : The agent has been registered with the container with the assigned name and address, and then it can be operated in the agent communication.
– Suspended : The agent is temporarily stopped and none of its behaviour(s) can be executed.
– Terminated : The agent is absolutely terminated, and all its resource is disposed and all its behaviour(s) is/are shut down. The agent is deregistered from the container.

During the life cycle, each agent manifest itself with one of the above mentioned states corresponding to the action it is operating. One agent start to live through initialization action during *initiated* state, and it is terminated through quit action during *terminated* state. During the other two states (i.e., *active* and *suspended* states), the agent operates a set of actions in response to various situations. (See Figure 5.11)

We design five classes of agents which cooperate with each other through communications to simulate change propagations during the project process. The cooperation between the agents is illustrated in Figure 5.12.

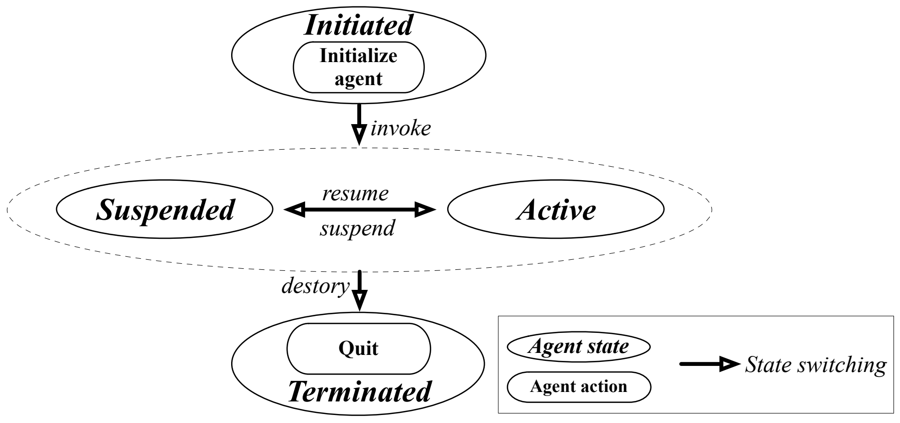


FIGURE 5.11 – Generic life cycle of agents

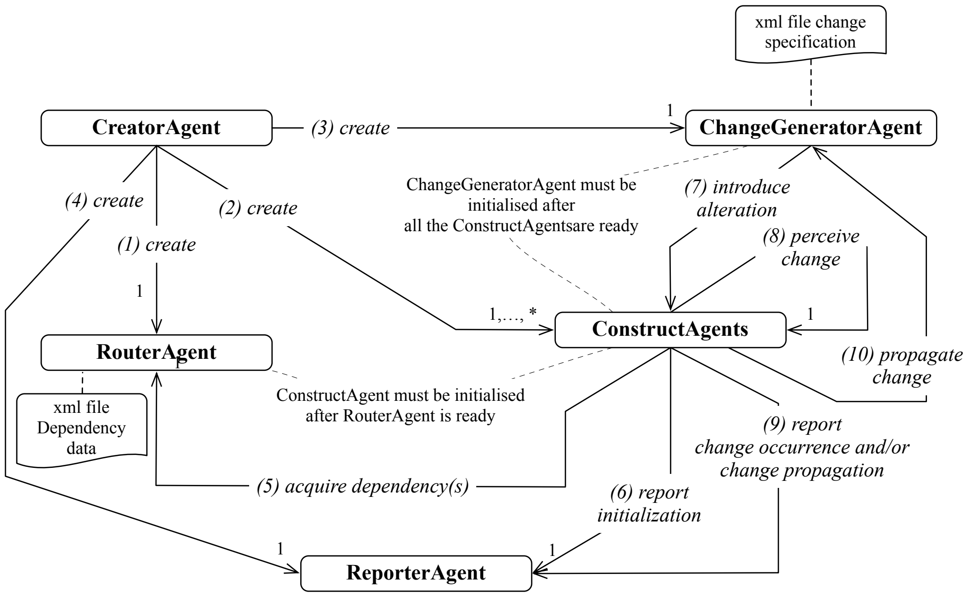

xml file change specification
CreatorAgent
(3) create
1
ChangeGeneratorAgent
(4) create
(2) create
(1) create
ChangeGeneratorAgent must be initialised after all the ConstructAgentsare ready
(7) introduce alteration
(8) perceive change
1
RouterAgent
1,..., *
ConstructAgents
1
xml file Dependency data
ConstructAgent must be initialised after RouterAgent is ready
(10) propagate change
(9) report change occurrence and/or change propagation
(5) acquire dependency(s)
(6) report initialization
1
1
ReporterAgent

FIGURE 5.12 – Agent communications

In Figure 5.12, the cooperating procedures (by the arrows) are labeled with the numeric phrase indicating the particular intention owned by the agents during the cooperation. Each procedure is described as following :

(1) CreatorAgent *creates* one RouterAgent which will then load the routing information.

(2) CreatorAgent *creates* one or more ConstructAgents corresponding to all the constructs obtained through modeling methodology (see Chapter 3).

(3) CreatorAgent *creates* one ChangeGeneratorAgent which is then ready for introducing alterations to the ConstructAgents.

(4) CreatorAgent *creates* one ReporterAgent which collects data from all the other ones to support data visualization.

(5) ConstructAgent sends query to RouterAgent to *acquire* the dependency information (i.e., the routing information in the construct network).

(6) Each initialized ConstructAgent *reports* to ReporterAgent that it is ready to participate in the simulation.

(7) When all the ConstructAgents have been ready, ChangeGenerator *introduces* alterations to the particular one(s).

(8) With the received alteration, each of the involved ConstructAgents analyzes it to determine whether a change is *perceived*.

(9) Given an occurred change within a ConstructAgent, the agent *report* the change occurrence and the relevant change propagation to ReporterAgent.

(10) Given an occurred change within a ConstructAgent, the ConstructAgent transfers the impact of the occurred change to *propagate* the change.

Among the agents, the following constraints are specified :
– The CreatorAgent must be the first created agent among all the agents.
– Only after the RouterAgent has been created and ready, then the ConstructAgents can be created.
– Only after all the ConstructAgents have been created and ready, then the ChangeGenerator can be created.

5.5.1 CreatorAgent

CreatorAgent is the first agent that is created once the agent container is built up. It is responsible for loading the configuration of all the other agents and creating them accordingly. Based on the generic life cycle (see Figure 5.11), the life cycle of the CreatorAgent is illustrated in Figure 5.13.

To explain the actions of the CreatorAgent in details, we present Table 5.1.

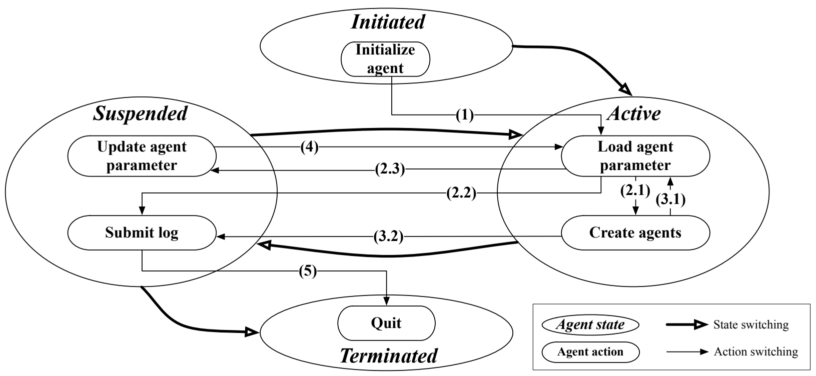


FIGURE 5.13 – Life cycle of CreatorAgent

TABLE 5.1 – Actions of CreatorAgent

Involved state	*Action*	*Description*	*Condition*
Initiated	Initialize agent	The agent is created as the first one and bound with the data	(1) If the initialization is complete, then agent switches to active state
Active	Load agent parameter	When the agent is in active state, it begins to load the parameters of the agents to be created	(2.1) If the agent parameters have been loaded, then the agent begins to create the corresponding agent(s) (2.2) If the agent is to be terminated, then it would submit the log data (2.3) If the CreatorAgent's parameters are to be updated, then the agent would be suspended
	Create agents	When the agent has load the parameters of the objective agents, it creates the agents accordingly	(3.1) If the CreatorAgent is required to create another agents after the current creation, then it continues to load some new parameters (3.2) If the agent is to be terminated, then it would submit the log data
Suspended	Update agent parameter	When some parameters of an activated agent is required to be updated, the agent is suspended with updating	(4) If the agent completes updating its parameter, then it resumes to be ready to load agent parameter
	Submit log	When the agent has updated the operating data or is required to report its log, it submits the log to the designated destination	(5) If the agent is going to quit, then it switches to terminate itself
Terminated	Quit	The agent is terminated with disposing its data	-

5.5.2 RouterAgent

RouterAgent is created by the CreatorAgent and is responsible for loading the agent communication configuration parameters to respond the routing information query from other agents. The life cycle of the agent is illustrated in Figure 5.14.

The actions of the RouterAgent are explained in details. (see Table 5.2)

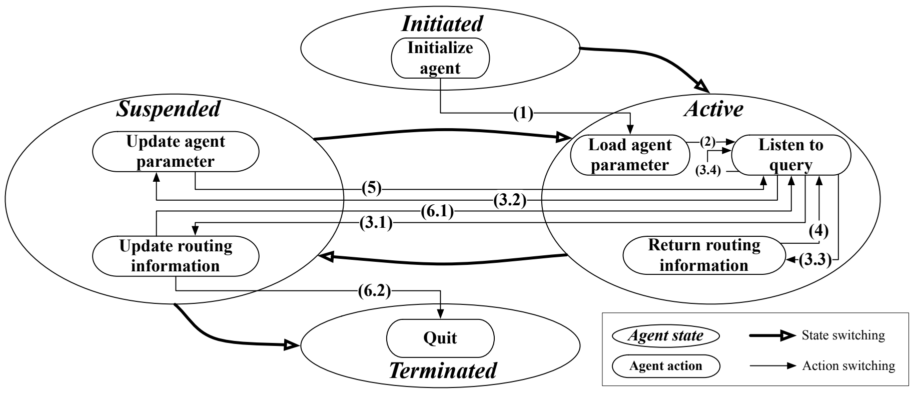


FIGURE 5.14 – Life cycle of RouterAgent

TABLE 5.2 – Actions of RouterAgent

Involved state	*Action*	*Description*	*Condition*
Initiated	Initialize agent	The agent is created and bound with the data	(1) If the initialization is complete, then agent switches to active state
Active	Load agent parameter	When the agent is in active state, it begins to load its own parameters and the routing parameters of the ConstructAgents	(2) If the parameters have been loaded, then the agent begins to listen to the query from ConstructAgents
	Listen to query	When the agent receives a query for routing information from ConstructAgent(s), it begins to search the owned routing data in order to respond to the query	(3.1) If the routing data/information is to be updated, then the agent would be suspended (3.2) If the parameters of the RouterAgent are to be updated, then the agent would be suspended (3.3) If the RouterAgent finds out the result to answer the query, then it is to return the routing information (3.4) If the RouterAgent does not execute any other action, then it keeps listening to query
	Return routing information	The RouterAgent returns the routing information as the result to the query from the ConstructAgent(s)	(4) If the agent has send back the result, then it would continue to listen to another new query
Suspended	Update agent parameter	When the parameters of the RouterAgent is required to be updated, the agent is suspended with updating	(5) If the agent completes to update data, then it would continue to listen to query
	Update routing information	When the RouterAgent is required to update the routing information, it is suspended	(6.1) If the agent is still in working, then it would resume to listen to query after updating the routing information (6.2) If the agent is going to quit, then it switches to terminate itself
Terminated	Quit	The agent is terminated with disposing its data	-

5.5.3 ConstructAgent

ConstructAgent : The agent is created by the CreatorAgent after the RouterAgent is initialized and activated. Each of ConstructAgents represents a construct in our methodology. After being initialized and activated, each ConstructAgent notifies its existence to the RouterAgent to indicate that the ConstructAgent is ready for further operations.

As the life cycle illustrated in Figure 5.15, we explain the actions executed in details. (see Table 5.3)

TABLE 5.3 – Actions of ConstructAgent

Involved state	*Action*	*Description*	*Condition*
Initiated	Initialize agent	The agent is created and bound with the data	(1) If the initialization is complete, then agent switches to active state
Active	Listen to alteration	When the agent is in active state, it begins to listen to whether there is an alteration is introduced	(2.1) If an alteration is introduced, then the agent begins to analyze it (2.2) If there is neither any alteration being introduced nor agent state being switched, then the agent keeps listening (2.3) If the agent parameters are to be updated, then the agent would be suspended (2.4) If the agent is to be terminated, then it would submit the log data
	Analyze alteration	When the agent perceive an introduced alteration, it begins to analyze the alteration	(3.1) If the alteration causes some data to be out of its tolerance, then the agent would continue to identify change (3.2) If the alteration only shifts the data within the tolerance, then the agent would keep listening to alterations
	Identify change	When the alteration shifts some data out of the tolerance, the agent begins to identify some change due to it	(4) If the agent identify the change, it would determine some procedure to cope with the change
	Determine procedure	With an occurred change is identify, the agent begins to determine some procedure to cope with the change according to the predefined strategies, operations.	(5) If a procedure is determined, then the agent would update its state resulting from the occurred change
Suspended	Update data	When some procedure is determined to cope with the occurred change, the agent suspends and updates the data it is operating due to the change	(6) If the agent completes to update data, then it would submit the log information to the designated destination
	Submit log	When the agent has updated the operating data or is required to report its log, it begins to submit the log to the designated destination	(7.1) If the agent is still in working, then it would resume to listen to alteration after submitting the log (7.2) If the agent is going to quit, then it switches to terminate itself
	Update agent parameter	When some parameters of the agent is required to be updated, the agent is suspended	(8) If the agent completes updating parameters, then itresumes to listen to alteration
Terminated	Quit	The agent is terminated with disposing its data	-

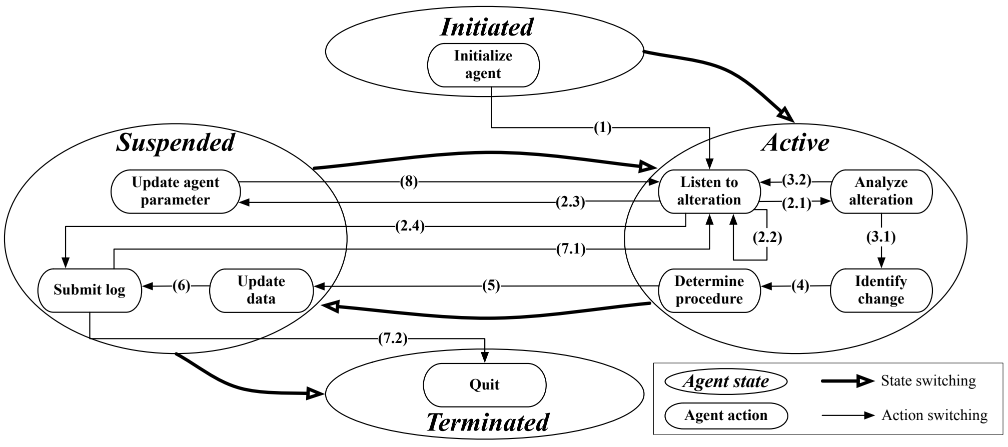


FIGURE 5.15 – Life cycle of ConstructAgent

5.5.4 ChangeGeneratorAgent

ChangeGeneratorAgent : The agent is created by the CreatorAgent after all the ConstructAgents are intialized and activated. The agent is responsible for generating the specified alterations and introducing them to the corresponding ConstructAgent(s). At the same time, the ChangeGeneratorAgent is also responsible for propagating the impact of changes by introducing the propagated alterations.

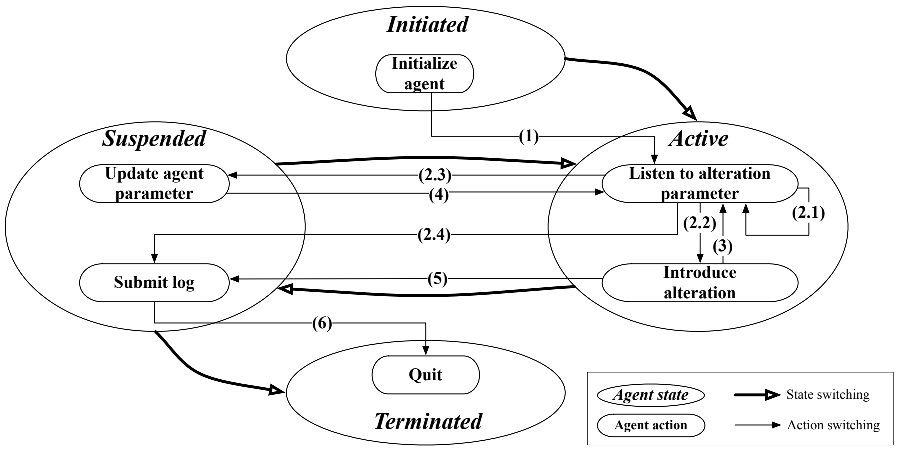


FIGURE 5.16 – Life cycle of ChangeGenerator

In Table 5.4, we describe the actions of ChangeGeneratorAgent in details.

TABLE 5.4 – Actions of ChangeGeneratorAgent

Involved state	*Action*	*Description*	*Condition*
Initiated	Initialize agent	The agent is created and bound with the data	(1) If the initialization is complete, then agent switches to active state
Active	Listen to alteration parameter	When the agent is in active state, it begins to listen to alteration parameters for composing alteration	(2.1) If the agent does not execute any other action, then it keeps listening to alteration parameters (2.2) If the alteration parameters have been complete to compose an alteration, then the agent would introduce the alteration to ConstructAgent(s) (2.3) If the parameters of the ChangeGeneratorAgent are to be updated, then the agent would be suspended (2.4) If the agent is to be terminated, then it would be submit the log data
	Introduce alteration	When the agent receives the parameters for composing an alteration, it composes and then introduces the alteration to the corresponding ConstructAgent(s)	(3) If the agent has finished introducing the alteration, then it would continue to listen to new alteration parameters
Suspended	Update agent parameter	When the parameters of the ChangeGeneratorAgent is required to be updated, the agent is suspended with updating	(4) If the agent completes to update data, then it would continue to listen to alteration parameters
	Submit log	When the agent has introduced some alteration or is required to report its log, it begins to submit the log to the designated destination	(6) If the agent is going to quit, then it switches to terminate itself
Terminated	Quit	The agent is terminated with disposing its data	-

5.5.5 ReporterAgent

The agent is created by the CreatorAgent after all the other agents have been initialized and activated. The agent is responsible for collecting data from all the other agents and forwarding it to the user interface for presentation.

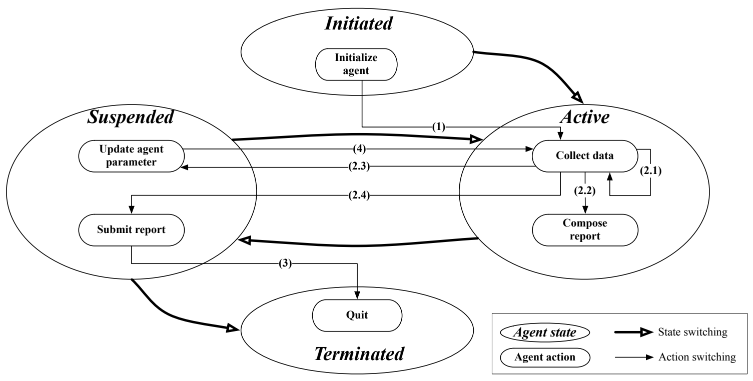


FIGURE 5.17 – Life cycle of ReporterAgent

The actions of ReportAgent are stated in Table 5.5.

TABLE 5.5 – Actions of ReporterAgent

Involved state	*Action*	*Description*	*Condition*
Initiated	Initialize agent	The agent is created and bound with the data	(1) If the initialization is complete, then agent switches to active state
Active	Collect data	When the agent is in active state, it begins to collect the data produced during the agent communication	(2.1) If the agent does not execute any other action, then it keeps collecting action (2.2) If the collected data is ready, then the agent would compose report for the further presentation (2.3) If the parameters of the ReporterAgent are to be updated, then the agent would be suspended (2.4) If the agent is to be terminated, then it would be submit the log data
	Compose report	When the agent has collected the required data for presentation, it composes a report to support the data visualization	(3) If the agent has finished composing the report, then it would continue to collect data
Suspended	Update agent parameter	When the parameters of the ReportAgent is required to be updated, the agent is suspended with updating	(4) If the agent completes to update data, then it would continue to collect data
	Submit log	When the agent is required to report its log, it begins to submit the log to the designated destination	(6) If the agent is going to quit, then it switches to terminate itself
Terminated	Quit	The agent is terminated with disposing its data	-

5.6 Specifications of Data Entities

Based on the model of PD project and the rationale of simulating change propagations, we transform the KID into the structured entities. In this way, the KID can be parsed and operated by the implementation solution. Referring to each entity, the data items as the properties are explained, and meanwhile, the relations between the entities are also presented. The details of the data entities are presented in appendix D. The entities include :

1. Construct : As aggregating the KID from the multiple knowledge areas, one construct includes the linkages to the building blocks belonging to the knowledge areas. At the same time, in considering that the constructs are also regarded as the "location" where changes are observed to occur, each construct entity preserves recording the potential changes and change propagations. (see Table D.1 for details)
2. Building block : One building block contains the KID from one of the knowledge areas, and the information/data is organized by a set of direct parameters. The change(s) occurring within a construct can be investigated in the specific building block in further. So each building block entity also preserves recording the potential changes and change propagations. (see Table D.2 for details)
3. Direct parameter : One direct parameter describes one of the attributes of a building block. Referring to one direct parameter entity, the value of the attribute as well as the tolerance are preserved. The concerned dependencies are also recorded. (see Table D.3 for details)
4. Dependency : Considering that dependency refers to the effect that changes one direct parameter onto another one, a dependency entity contains the formalized specification of the relations transforming the effect between the direct parameters. (see Table D.4 for details)
5. Alteration : An alteration is the shift modifying the value of the involved direct parameter. In an alteration entity, the shift is specified corresponding to the direct parameter data type. (see Table D.5 for details)
6. Alteration list : Corresponding to one building block, the introduced alterations are organized as a list which can be referenced by its identity. (see Table D.6 for details)
7. Change : An identified change is described in a change entity, and the due alteration is also recorded in the entity. (see Table D.7 for details)
8. Change propagation : One perceived change propagation is described as one change causes another one through the dependency between two involved direct parameters. (see Table D.8 for details)
9. Change list : Corresponding to one building block, the occurred changes are organized as a list which can be referenced by its identity. (see Table D.9 for details)

10. Direct parameter list : Corresponding to one building block, the involved direct parameters are organized as a list which can be referenced by its identity. (see Table D.10 for details)
11. Dependency list : Corresponding to one direct parameter, the concerned dependencies are organized as a list which can be referenced by its identify. (see Table D.11 for details)

The relations between the above entities as well as their properties are stated in a Entity-Relation diagram illustrated by Figure 5.18.

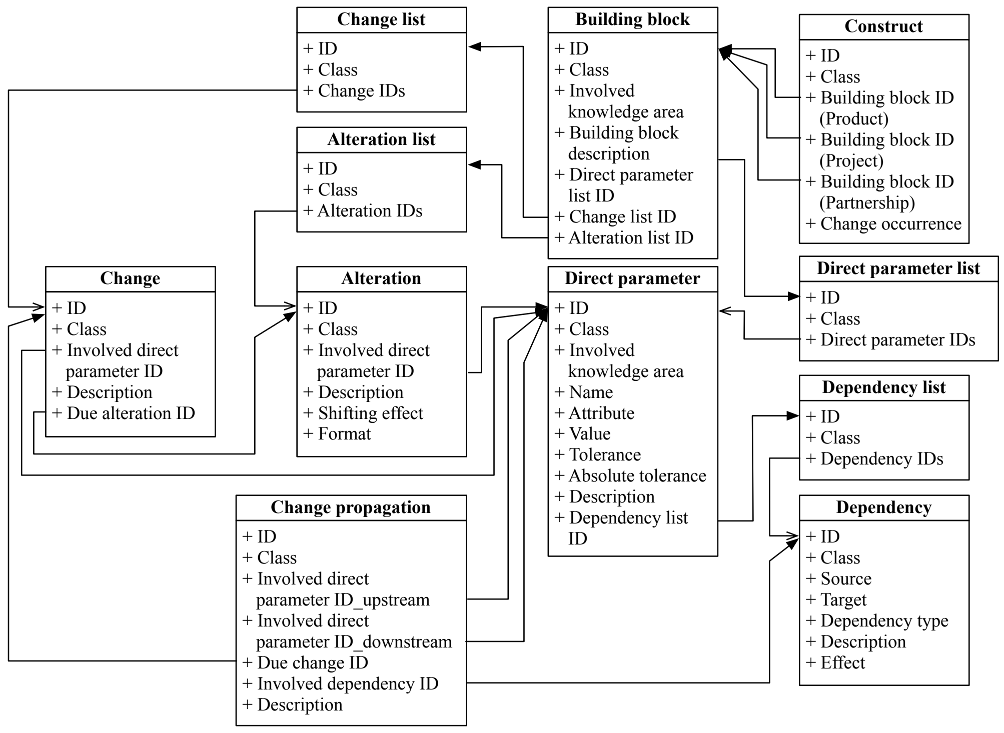


FIGURE 5.18 – Entity-Relation diagram

5.7 Class Specifications

Figure 5.19 illustrates the class diagram of the agent-based system.

As Figure 5.19 illustrating, a *PD project* contains three *knowledge areas* covering the product, project and partnership management issues. From each knowledge area, a set of *building blocks* are extracted as the data entities representing the KID emerged during the PD project. All the building blocks categorized into *product building blocks*, *project building blocks* and *partnership building blocks*. Among the building blocks, there are *relations* of *effort-based* and *non-effort-based* classes. The product building blocks are also treated as the main clue to create the *constructs* which enable us to consider treating change propagations within and between the three knowledge areas simultaneously.

Referring to the product building blocks belonging to the product management knowledge area, they emerge four *states* which evolve through *needs*, *requirements*, *logical solution* and *physical solution* which are recorded in the four corresponding *deliverables*. Each deliverables is a set of *documents* containing all the information/data generated during the PD project. The project building block is decomposed into a set of *phases*, and each phase can be decomposed into a set of *tasks* in further. The partnership building blocks are specified as two *partner classes*, i.e., end product suppliers and enabling product suppliers. For each partner class, there could exist a number of *partner individual* representing the various companies, and the partner individuals perform different *partner responsibilities* during participating in the tasks of the PD project.

Each building block is chararcterized by a set of *direct parameters* with the pairs of attribute and parameter value. Each direct parameter is constrained by a couple of *tolerance* domain and *absolute tolerance* domain, and each domain is expressed by the minimum/maximum *boundary values* and *absolute boundary values*.

Between two direct parameters, there could exist a *dependency* which reflects the effect of changing one direct parameter onto the other.

Each direct parameter could be introduced into an *alteration* that would cause the direct parameter value shifted. By comparing the consequence due to the alteration with the direct parameter tolerance/absolute tolerance, the *causes of change* are suggested and a number of *changes* are identified. The dependencies between the direct parameters works as the *change propagation channels* that transfer the effect of change from one direct parameter to the other connected one.

The dependencies between the direct parameters belonging to the different building blocks implement the *relations* between their located building blocks. The building blocks relations are categorized into the *effort-based* and *non-effort-based* ones. The former reflects the product evolution process, and the latter reflects the rest couplings between different building blocks. In further, the relations of the building blocks owned by different construct implement the *connections* between different constructs.

Corresponding to each of constructs, an *agent* is initiated and activated. The agents are incorporated with a set of plan used to determine the preset actions during simulating change propagation in various possible scenario.

5.8 Chapter Summary

This chapter discussed the methodology of how to simulate change propagation. Based on constructs and their relations (represented by the direct parameters), a change propagation network is established. Each node (construct) is equipped with the intelligence of identifying, estimating and coping with changes according to the principles of conceptual models of change occurrence and change propagation. The simulation solution starts with introducing the alterations. Through analyzing the alterations, whether a change emerges or not will be judged. If a change emerges, the coping actions will be determined and then analyzed to decide whether the change will be propagated to the neighbor nodes.

We also proposed a multi-layer system architecture that contains four layers, i.e., Presentation Layer, Manipulation Layer, Network Layer and Data Layer. The Presentation layer is the top layer logically closer to the visual presentation, whereas Data Layer is the bottom layer logically closer to the data abstraction. Then this chapter turned to the introduction of the programming environment. We selected an open source software called JADE that enables us to develop agent-based system with a simplified software framework implemented by Java language. After the introduction to JADE, it came to agent configuration. We categorized four states during the life cycle of the agents and designed five classes of agents which cooperate with each other through communications to simulate change propagations during the project process. At the end of this chapter, we presented the specifications of data entities as well as an Entity-Relation diagram showing the relations between the entities and their properties.

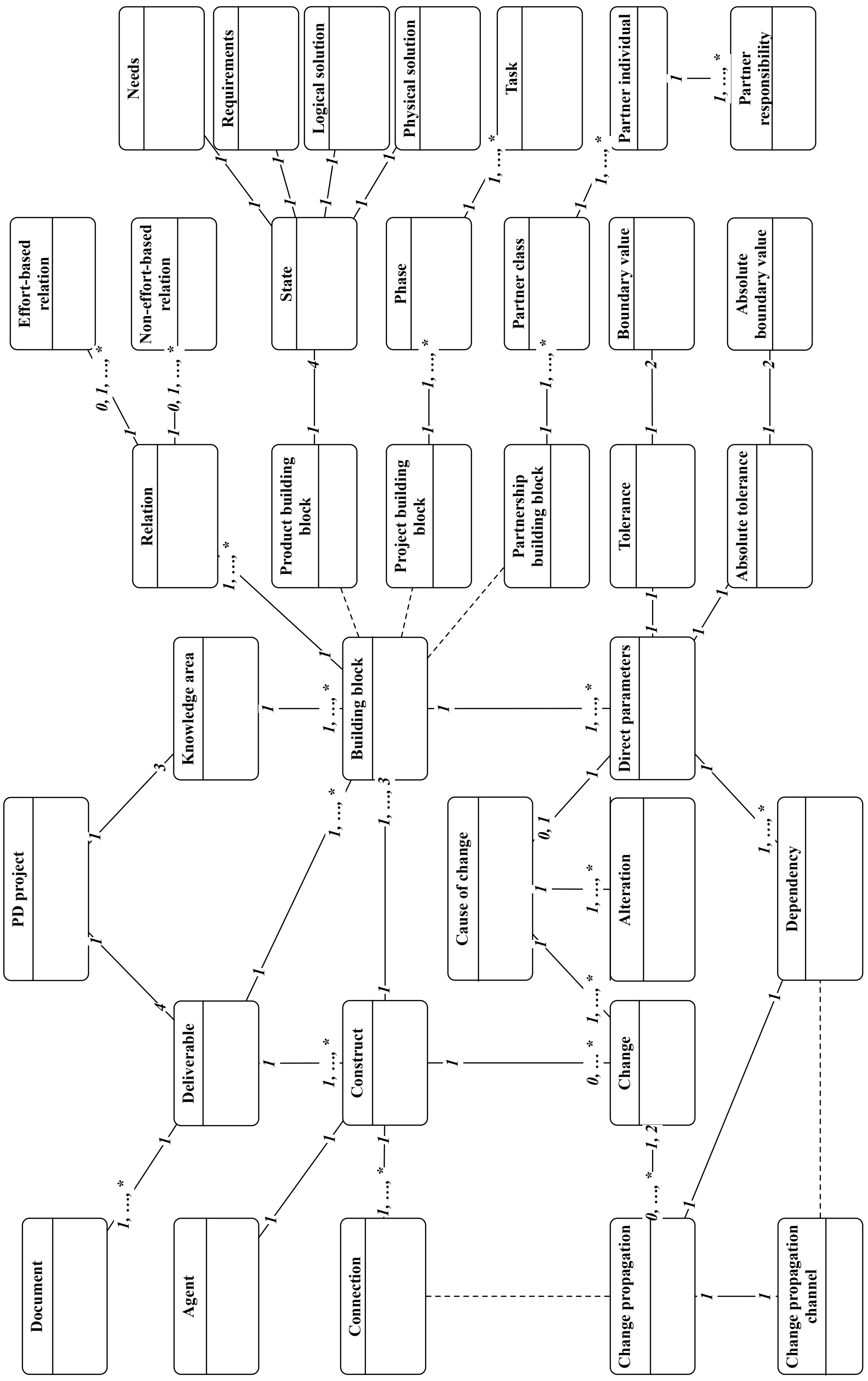


FIGURE 5.19 – Class diagram

6 Verification and Application

▷ *In this chapter, we aim to verify the concepts, methods and the prototype through an illustrative case. We firstly introduce a verification paradigm illustrating our verification procedure. After, we present the illustrative case that is used throughout this chapter. Then we introduce an application to the COCA methodology with the case, verify the models used to identify change and change propagations, and verify the prototype on the case. At the end of this chapter, we show how to apply the prototype in detail.*

◁

Plan of Chapter

6.1 Verification Paradigm

In general, verification is the evaluation of whether or not a product, service, or system meets the requirement, the specification, the imposed condition or a regulation, which is often an internal process (IEEE, 2011). The verification can be expressed by a question "Are we building the things right ?" (Boehm & DeMarco, 1997). Sargent (2012) has proposed a generalized view of verification and validation process encompassing the conceptual model and the computerized model. In his process, conceptual model is verified or validated by determining whether the theories and assumptions underlying the conceptual model are correct and whether the model represents the problem entity in a reasonable way. Whereas, computerized model verification is defined as assuring that the conceptual model is implemented on computer correctly. Our verification paradigm follows the process proposed by Sargent (2012) but slightly modified to fit our research (see Figure 6.1).

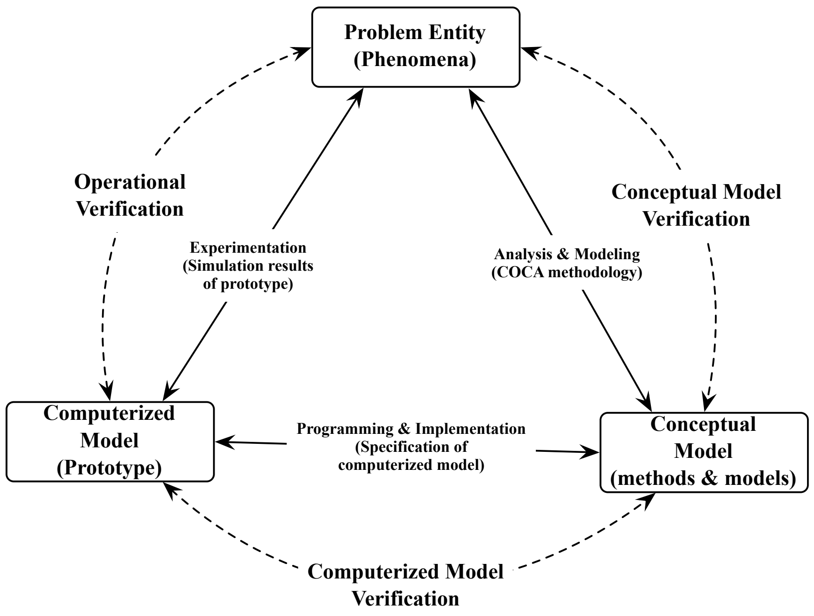

Problem Entity (Phenomena)
Operational Verification
Conceptual Model Verification
Experimentation (Simulation results of prototype)
Analysis & Modeling (COCA methodology)
Computerized Model (Prototype)
Programming & Implementation (Specification of computerized model)
Conceptual Model (methods & models)
Computerized Model Verification

FIGURE 6.1 – Verification paradigm, adapted from (Sargent, 2012)

In the paradigm, *Problem Entity* is the phenomena to be analyzed and modeled. In our research, the phenomena is change and change propagation including the background and context of their occurrence, the cause and situations of their occurrence, the relations between change and change propagation, etc. (refer to Chapter 2). *Conceptual Model* is the representation of the problem entity. Here is the model and method we establish to analyse and identify change occurrence and change propagation (refer to Chapter 4). *Computerized model* is the implementation of conceptual models on computer. It is the prototype we design and develop to implement our model and method (refer to Chapter 5).

Conceptual Model Verification

Conceptual model verification intends to justify whether the model and method that reflects the problem entity is correct. In order to establish the conceptual model, we firstly analyze the problem entity within the scope of our research (refer to Chapter 3). In this case, we verify the COCA framework first by introducing an application with the illustrative case. And then we use the data from the case to prove the correctness and feasibility of the conceptual models of identifying change occurrence and change propagation. The more specific concepts and methods we are going to verify can be seen from Table 6.1.

Computerized Model Verification

Computerized model is developed according to the specification which is a written detailed description of how we translate the conceptual model to an implementable model on computer (refer to Chapter 5). Therefore, Computerized model verification is to verify whether the implementation is consistent with specification, i.e., whether the concepts and methods are implemented correctly. We test the prototype from two aspects. One is to check if the agents work as we expect ; the other is to see if the representations of the results is corresponding to the design. The more specific concepts and methods we are going to verify can be seen from Table 6.1.

Operational Verification

At last, the output behaviors of the computerized model will be tested through executing the experimentation, that is operational verification. We achieve this point by introducing the data from the case into the prototype and compare the simulation results with our expectation. The more specific contents can be seen from Table 6.1.

TABLE 6.1 – The concepts and methods to be verified

Verfication	*Methods*	*Concepts*	*Refer to*
Conceptual Model Verification	COCA Framework	knowledge area, product evolution process, construct	section 6.3
	Conceptual models	change occurrence, change propagation, tolerance, absolute tolerance	section 6.4
Computerized Model Verification	Simulating methodology	multi-layer construct structure, simultaneous changes	section 6.5
	Agent-based technical solution	agent intelligence, agent communication	section 6.5
Operational Verification	prototype	user interface	section 6.6

6.2 Illustrative Case Background

This section is to give an overview of the illustrative case. This illustrative case is a made-up example but tends to be as realistic as possible and is made by gathering data from the scientific journals and specialized website. Here we provide a general introduction of the illustrative case, the details can be referred to Annexe B.

Alouette Bicycles (a.k.a. A.B.) is a company that engineers and makes high-performance full bicycle suspension for mountain bikes. The company offers suspension designs ranging from the simple efficiency of the single pivot designs to the cutting edge performance. A.B. formulates the product strategy mainly through considering two aspects : On one hand, A.B. learns the general requirements from the marketplace in aim of earning benefit and keeping competitiveness. On the other hand, A.B. also makes answers to the special requests for quotation from the professional athletes.

Correspondingly, there are mainly two types of orders arriving at A.B., one type is for ordering the particular configured suspension systems that are specially installed into the bikes used in professional competitions. The other type of order comes from generic market requirement and the products with categorized configurations are supplied to the regular customers, such as the amateur enthusiasts. For the former type of order, the focal company produces the products in Assembly-To-Order (ATO) strategy, whereas the latter type of order is answered by the products produced in Make-To-Stock (MTS) strategy.

The final product in this example is a full suspension system for mountain bike. It is generally used to hold the rider off the ground when riding a bike. Because the various road conditions passing under the bike, the rider will have different feelings during riding the bike, such as bumping long the road, too much vibration to the hands, etc. With the full suspension system equipped within the bike, the impacts, influences (especially the ones causing the negative feelings) can be absorbed or insulated, and the rider is therefore enable to gain more comfort, more control. The full suspension system for mountain bike includes the front fork and the frameset. The front fork is installed with the front suspension in the two branches (i.e., the stanchions explained later) to implement absorbing shocks, whereas the frameset implements the suspension by a rear shock (i.e., an absorber) between the front section and the rear swingarm (see Figure 6.2).

In Figure 6.2(a), a mountain bike with a full suspension system is demonstrated and the final product in this example is highlighted by the colored stroke. The final product is then illustrated in 6.2(b), and the main modules/components are listed as follows :

- Front fork (green stroke area)
- Front section (red stroke area)
- Rear swingarm (blue stroke area)
- Rear shock (pink stroke area)

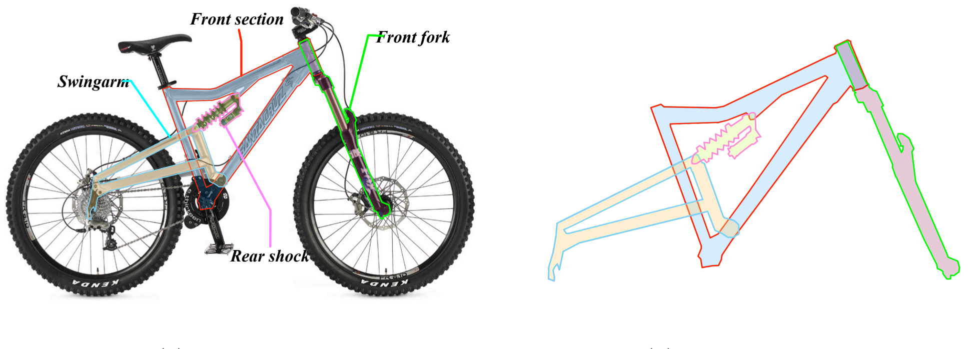


FIGURE 6.2 – The full suspension system for mountain bike

The details of those modules/components of the full suspension are introduced in section B.1 of Annexe B.

6.3 Applications of COCA Framework

To apply the COCA methodology into the illustrative follows the process outlined in Chapter 3, which involves two main aspects :

– highlighting knowledge areas
– identifying constructs

Each of them will now be discussed in detail.

6.3.1 Highlighting knowledge areas

To specify the three knowledge areas is based on the data and information collected from case in terms of project, product and supply chains.

Project management knowledge area

The project aims to provide the full suspension system under the constraints from time, cost and quality. According to the hierarchical model of systems in project proposed in Chapter 3, this project is first modeled into five phases, and then each phase is modeled as a set of tasks.

Phase 1 : planning. The objectives of full suspension system of new generation from the market analysis are studied and further the project mission are established. This phase contains six tasks : identify market opportunity, prepare Task/resource plan, approve resource and plan timing, establish technical and economic documents, determine project quality objectives, approve project mission.

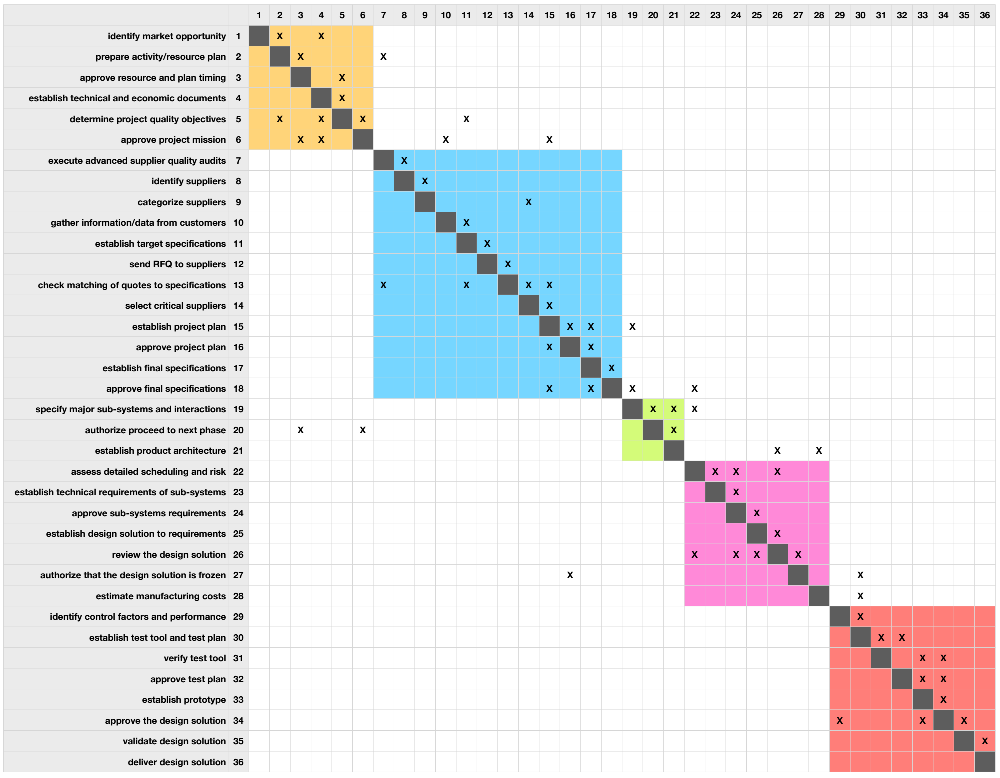
Phase 2 : concept development. The specification of full suspension systems generated and evaluated. Meanwhile, suppliers of certain components will be selected through negotiation and comparison. This phase contains twelve tasks : execute advanced supplier quality audits, identify suppliers, categorize suppliers, gather information/data from customers, establish target specifications, send RFQ to suppliers, check matching of quotes to specifications, select critical suppliers, establish project plan, approve project plan, establish final specifications, approve final specifications.

Phase 3 : system-level design. The architecture of full suspension system is defined, including the definitions of functional elements, the physical elements (components, modules) of the product and the interactions between the modules and/or components. This phase contains three tasks : specify major sub-systems and interactions, authorize proceed to next phase, and establish product architecture.

Phase 4 : detail design. The specification of all components are determined concerning the parameters, geometry, etc, as well as the technical requirements of sub-systems. This phase contains seven tasks : assess detailed scheduling and risk, establish technical requirements of sub-systems, approve sub-systems requirements, establish design solution to the requirements, review design solution, authorize that the design solution is frozen, estimate manufacturing costs.

Phase 5 : testing and refinement. The test plan, test tool and the prototype are defined. The design solution is finally validated and delivered. This phase contains eight tasks : identify control factors and performance, establish test tool and test plan, verify test tool, approve test plan, establish prototype, approve prototype, validate the design solution, deliver design solution.

Figure 6.3 illustrates the dependencies between tasks. More details including the explanation of each task, the overall arrangement of these tasks and the scheduling of all the tasks can be found in section B.2 of annexe B.



		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
identify market opportunity	1	■	X		X																																
prepare activity/resource plan	2		■	X				X																													
approve resource and plan timing	3			■		X																															
establish technical and economic documents	4				■	X																															
determine project quality objectives	5		X		X	■	X					X																									
approve project mission	6			X	X		■				X					X																					
execute advanced supplier quality audits	7							■	X																												
identify suppliers	8								■	X																											
categorize suppliers	9									■					X																						
gather information/data from customers	10										■	X																									
establish target specifications	11											■	X																								
send RFQ to suppliers	12												■	X																							
check matching of quotes to specifications	13							X				X		■	X	X																					
select critical suppliers	14														■	X																					
establish project plan	15															■	X	X		X																	
approve project plan	16															X	■	X																			
establish final specifications	17																	■	X																		
approve final specifications	18															X		X	■	X			X														
specify major sub-systems and interactions	19																			■	X	X	X														
authorize proceed to next phase	20			X			X														■	X															
establish product architecture	21																					■					X		X								
assess detailed scheduling and risk	22																						■	X	X		X										
establish technical requirements of sub-systems	23																							■	X												
approve sub-systems requirements	24																								■	X											
establish design solution to requirements	25																									■	X										
review the design solution	26																						X		X	X	■	X									
authorize that the design solution is frozen	27																X											■			X						
estimate manufacturing costs	28																												■		X						
identify control factors and performance	29																													■	X						
establish test tool and test plan	30																														■	X	X				
verify test tool	31																															■		X	X		
approve test plan	32																																■	X	X		
establish prototype	33																																	■	X		
approve the design solution	34																													X				X	■	X	
validate design solution	35																																			■	X
deliver design solution	36																																				■

FIGURE 6.3 – Task based DSM

Product management knowledge area

The product evolves along the process of PD project. According to the product evolution process proposed in Chapter 3, the product experiences four state referring to Needs, Requirement, Logical solution and Physical solution. Meanwhile, the four deliverables corresponding to the four states are generated. Figure 6.4 shows the evolution model of the full suspension system.

The needs definition deliverable mainly contains the customer needs about the full suspension system, such as reducing vibration to the hands, allowing easy traversal of difficult terrain, etc. It also contains some requirements in forms of metrics, which are translated from customer needs, for example, "attenuation from dropout to handlebar at $10Hz$" which is from the needs of reducing vibration to the hands. The requirement definition mainly contains target specifications of the full suspension system including function requirements, design constrains and performance. For example, the marginal value of spring preload is between $[480N, 800N]$ while the ideal value is between $[650N, 700N]$. The logical solution deliverable mainly contains the product architecture including the behaviors, functions and structures of the product. The key suppliers are also included such as Y-Focus Company and LAS Industrial Company. The physical solution mainly contains the geometry of the subsytems of the full suspension system and design solution.

The detailed examples for each deliverable is presented in section B.3 of annexe B. They show the details inside the deliverables as well as the transaction and connection from one deliverable to another.

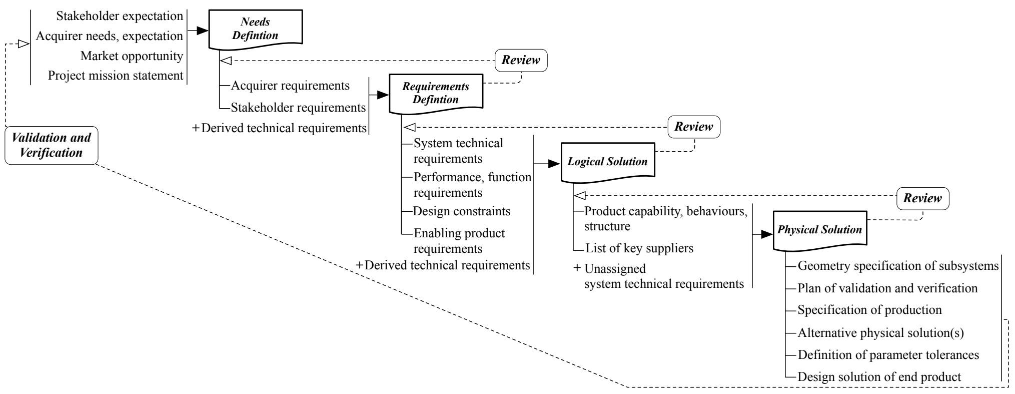

Stakeholder expectation
Acquirer needs, expectation
Market opportunity
Project mission statement
Needs Defintion
Review
Acquirer requirements
Stakeholder requirements
+Derived technical requirements
Requirements Defintion
Review
Validation and Verification
System technical requirements
Performance, function requirements
Design constraints
Enabling product requirements
+Derived technical requirements
Logical Solution
Review
Product capability, behaviours, structure
List of key suppliers
+ Unassigned system technical requirements
Physical Solution
Review
Geometry specification of subsystems
Plan of validation and verification
Specification of production
Alternative physical solution(s)
Definition of parameter tolerances
Design solution of end product

FIGURE 6.4 – Evolution model of full suspension system

During the evolution process, the aimed final product - full suspension system for mountain bike is specified into two major subsystems, i.e., Front fork, and Frameset. Further, these two subsystem are divided into a number of building blocks, i.e., the components or the modules not divided in further. The details of the building blocks are illustrated in Figure 6.5 while the relations between those building blocks are illustrated in Figure 6.6. Those relations are non-effort-based relations because they are reflected by the dependencies between the direct parameters characterizing the building blocks.

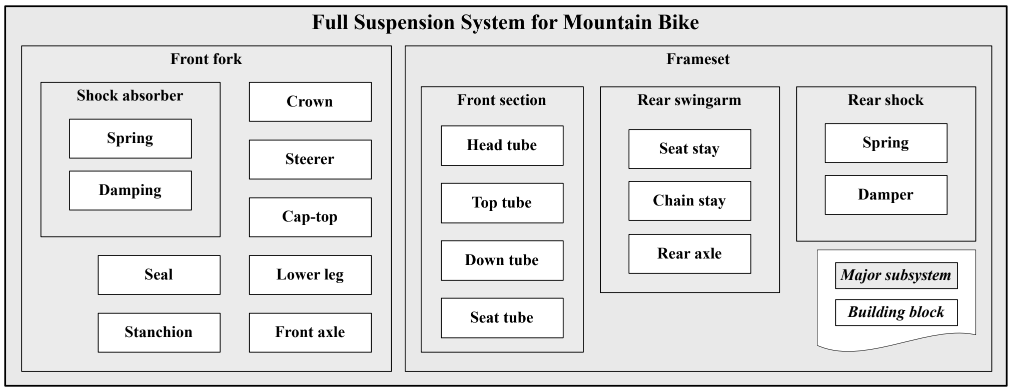
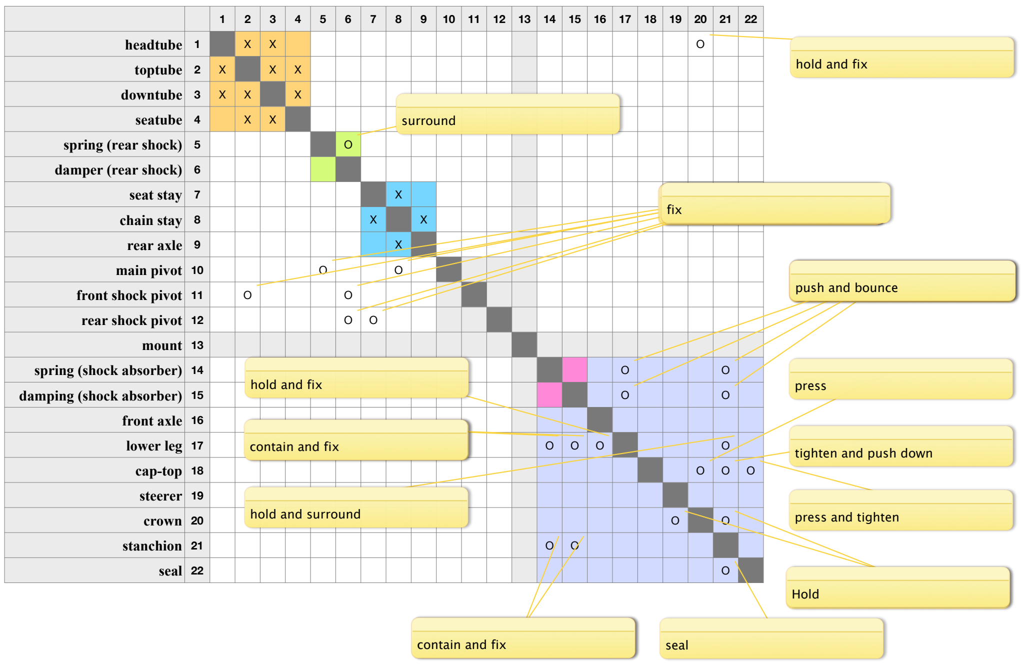


FIGURE 6.5 – Architecture of full suspension system

Figure 6.6 illustrates the component-based DSM that presents both the symmetric (labeled by "×") and the asymmetric relations (labeled by "◯") between the components. In the matrix, the symmetric relations refers to "connecting".



		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
headtube	1		X	X																	O		
toptube	2	X		X	X																		
downtube	3	X	X		X																		
seatube	4		X	X																			
spring (rear shock)	5						O																
damper (rear shock)	6																						
seat stay	7								X														
chain stay	8							X		X													
rear axle	9								X														
main pivot	10					O			O														
front shock pivot	11		O				O																
rear shock pivot	12						O	O															
mount	13																						
spring (shock absorber)	14																	O				O	
damping (shock absorber)	15																	O				O	
front axle	16																						
lower leg	17														O	O	O					O	
cap-top	18																				O	O	O
steerer	19																						
crown	20																			O		O	
stanchion	21														O	O							
seal	22																					O	

FIGURE 6.6 – Component-base DSM

Partnership management knowledge area

We only focus the partners who are in direct business relationships with the company. In other words, we present a simple structured supply chain around the company, in which no any other direct relationships are found between the partners. A.B. Maintains close relationships with Y-Focus Company (a.k.a. YF), LAS Industrial Company (a.k.a. LAS) and BEAUSTEEL Industrial Company (a.k.a. BSTL) during the design and development phase, and it obtains the components/modules from these partners in Engineering-To-Order (ETO) manufacturing class. In other words, these suppliers are end product suppliers.

The supply chain around the company also includes other suppliers which are Alliance Rubber&Plastic Company (a.k.a. Alliance), FORMULA Lubricant Company (a.k.a. FORMULA), HANSON Studio design company (a.k.a. HS) and PEAK Fastener Technologies Cooperation (a.k.a. PEAK). From these four companies, the focal company purchases the supplied items and services in more flexible way. In other words, these four companies supply some standard, low-complexity and highly flexible products/services to the focal company. These suppliers are also end product suppliers.

The responsibilities of those end product suppliers are as follows :

YF : The company supplies the shock absorbers for bicycles and motorbikes.

LAS : The company focuses on the production of front suspension forks for bicycles.

Alliance : The company is a professional and skillful manufacturer at all silicone products including seals, caps used in bicycle components.

BSTL : The company is an experienced producer that specializes in the line of titanium bicycles product, integrated manufacture, sale and service in one.

FORMULA : The company specializes in supplying the fluid solutions for mechanism products including various suspension systems.

HS : The company provides the dissemination solutions for bicycle products including the product outlook design, the customized stickers, the printing solutions and services.

PEAK : The company specializes in high-quality, customized fasteners and other non-standard parts in both small and large quantities.

6.3.2 Identifying constructs

Through highlighting the multiple knowledge areas and analyzing the interactions between them as well as between the building blocks, we are enabled to model the PD project progress and the product evolution process with considering the participation of the partners at the same time. According to the methods used to identify constructs in Chapter 3, we take product evolution as the main clue to gather the building blocks from the three knowledge areas . We follows the guidance below :

*To make the product evolves towards the design solution under the determined design methodology (**product**), what activities/tasks will be executed under what constraints (**project**), and is there any partner involved in and what their responsibilities (**partner**) are ?*

Different constructs are identified in different product states of“front fork” in the full suspension system along the product evolution process. In the tables, “ConstructID” stands for the id of the construct ; “BB_product” stands for the building block from product ; “BB_project” stands for the building block from project ; “BB_partner” stands for partner ; “DP_product” stands for the direct parameters for BB_product ; “DP_project” stands for the direct parameters for BB_project ; “DP_partner” stands for the direct parameters for BB_partner ; “Upstream ConstructID” stands for the construct ID where current construct evolves from. Here we only present the identified constructs during the state of “Physical solution” (see Table 6.2). Other examples can be found in section B.5 in annexe B.

TABLE 6.2 – Identified constructs during the state of "Physical solution"

ConstructID	*BB_product*	*BB_project*	*BB_partner*	*DP_product*	*DP_project*	*DP_partner*	*Upstream ConstructID*
Cst_04_01	front spring	A29 : establish design solution to requirements	YF	travel = 100mm ; material = titanium alloy ; external diameter = 25mm ; free length = 162mm	start time = 29/11/12 end time = 22/01/12 duration = 11w	delivery time before 30/12/12	Cst_03_01
Cst_04_02	lower leg	A29 : establish design solution to requirements	-	length = 370mm ; internal diameter = 45mm ; thickness = 1.2mm ; color = black ; leg distance = 100mm	start time = 29/11/12 end time = 22/01/12 duration = 11w	-	Cst_03_02
Cst_04_03	steerer tube	A29 : establish design solution to requirements	BSTL	length = 150,170,190,210,230 ; diameter = 28.58mm ; material = steel ; tube thickness = 2mm	start time = 29/11/12 end time = 22/01/12 duration = 11w	delivery time before 10/12/12	Cst_03_03
Cst_04_04	crown	A29 : establish design solution to requirements	-	material = alloy ; width = 120mm ; height = 35mm ; internal diameter (stanchion fix) = 30mm ; internal diameter (steerer fix) = 28.58mm	start time = 29/11/12 end time = 22/01/12 duration = 11w	-	Cst_03_08
Cst_04_05	cap-top	A29 : establish design solution to requirements	Alliance	bore diameter = 3.2mm ; threaded race diameter = 26.8mm ; outside diameter = 32mm ; color = black ; material = carbon fiber	start time = 29/11/12 end time = 22/01/12 duration = 11w	delivery time before 05/01/12	Cst_03_09
Cst_04_06	front axle	A29 : establish design solution to requirements	-	diameter = 15mm ; length = 120mm ; material = hi-alloy	start time = 29/11/12 end time = 22/01/12 duration = 11w	-	Cst_03_10
Cst_04_07	stanchion	A29 : establish design solution to requirements	-	material = hi-tensile steel ; stanchion surface material = bronze ; external diameter = 30mm ; thickness = 1.6mm ; length = 200mm ; internal diameter = 26.8mm	start time = 29/11/12 end time = 22/01/12 duration = 11w	-	Cst_03_11

6.4 Verification of the conceptual models

There are two types of conceptual models : change occurrence conceptual model that is used to justify whether an alteration on the construct causes a change, change propagation conceptual model that is used to justify whether a change propagates between two constructs. We have already used simple cases to explain and prove the correctness of these two conceptual models in Chapter 4. Base on these two models, the change propagation trajectories corresponding to the change propagation pattern proposed in Chapter 4 could also be identified.

According to the design solution of the front fork, there exists dependencies between stanchion and cap-top, between stanchion and front spring, between crown and stanchion, between stanchion and lower leg. Readers can refer to section B.1 and B.3 in annexe B for the structure and dependencies between components. In this case, any alteration that can translated into change on one of these building blocks will propagate through the dependencies if the caused change is correspond to the change propagation conceptual model. For example, we take the data in Table 6.2 to show how the change propagation trajectories are found. When the *internal diameter* of lower leg decreases out of the minimum tolerance boundary value, which is estimated as a change, it will cause the *internal diameter* of stanchion to decrease. In further, the *external diameter* of front spring is caused to decease due to the decreased internal diameter of stanchion. Based on this result, the direct parameter oriented change propagation trajectory is from *internal diameter* of lower leg to *internal diameter* of stanchion to *external diameter* of front spring ; the building block oriented change propagation trajectory is from lower leg to stanchion to front spring ; the construct level change propagation trajectory is from Cst_04_02 to Cst_04_07 to Cst_04_01 (see Figure 6.7).

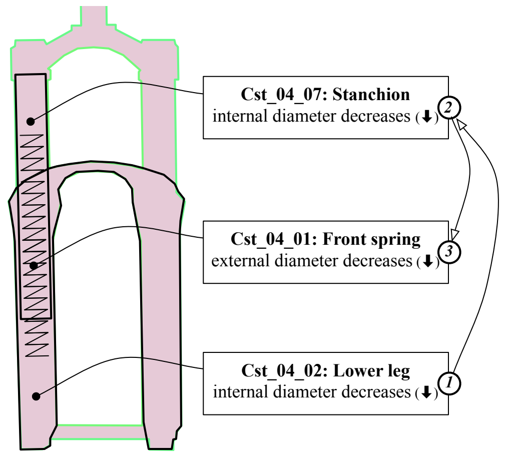

Cst_04_07: Stanchion
internal diameter decreases (⬇)
2
Cst_04_01: Front spring
external diameter decreases (⬇)
3
Cst_04_02: Lower leg
internal diameter decreases (⬇)
1

FIGURE 6.7 – Change propagation between multiple constructs

In some situations, the change first propagates inside one construct or one building block and then propagates outside. When *internal diameter* of stanchion decreases, *external diameter* of stanchion decreases that further cause the *internal diameter* of crown to decrease. Therefore, the direct parameter oriented change propagation trajectory is from *internal diameter* of stanchion to *external diameter* of stanchion to *internal diameter* of crown ; the building block oriented change propagation trajectory is from stanchion to crown ; the construct level change propagation trajectory is from Cst_04_07 to Cst_04_04 (see Figure 6.8).

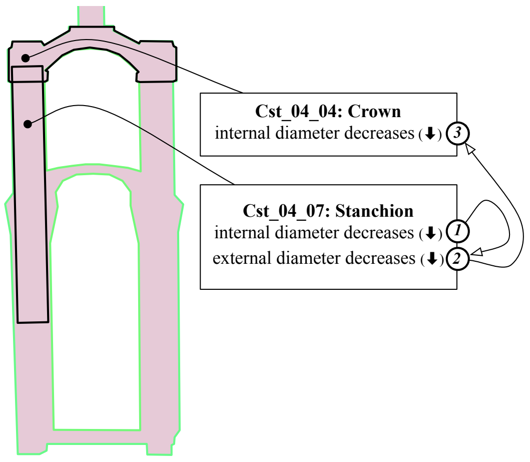

Cst_04_04: Crown
internal diameter decreases (⬇) 3
Cst_04_07: Stanchion
internal diameter decreases (⬇) 1
external diameter decreases (⬇) 2

FIGURE 6.8 – Change propagation within one construct

6.5 Computerized Model Verification

This section focuses on how to verify the computerized model on the illustrative case, which involves in two aspects :

– verifying simulating methodology
– verifying the prototype which is implemented by agent-based technical solution.

Before explaining these two aspect verification, we firstly give an introduction to the user interface. Take Figure 6.9 for example. The right part of the user interface shows the construct network, where the white nodes represent the constructs, and the pink lines connecting the nodes represent the dependencies between the constructs. The id of the construct is labeled on the node. The number of the lines between two nodes means the number of the different dependencies between two constructs. In addition, the percentage of the black area inside each node tells the frequency this construct is involved in. The more the black area is, the more times this construct is involved in during the change propagation.

The left part of the user interface give the information about the change and change propagation. It firstly tells which construct is involved in and the change occurs on which direct parameter. Then the details of change is given, followed by the potential trajectory of change propagation.

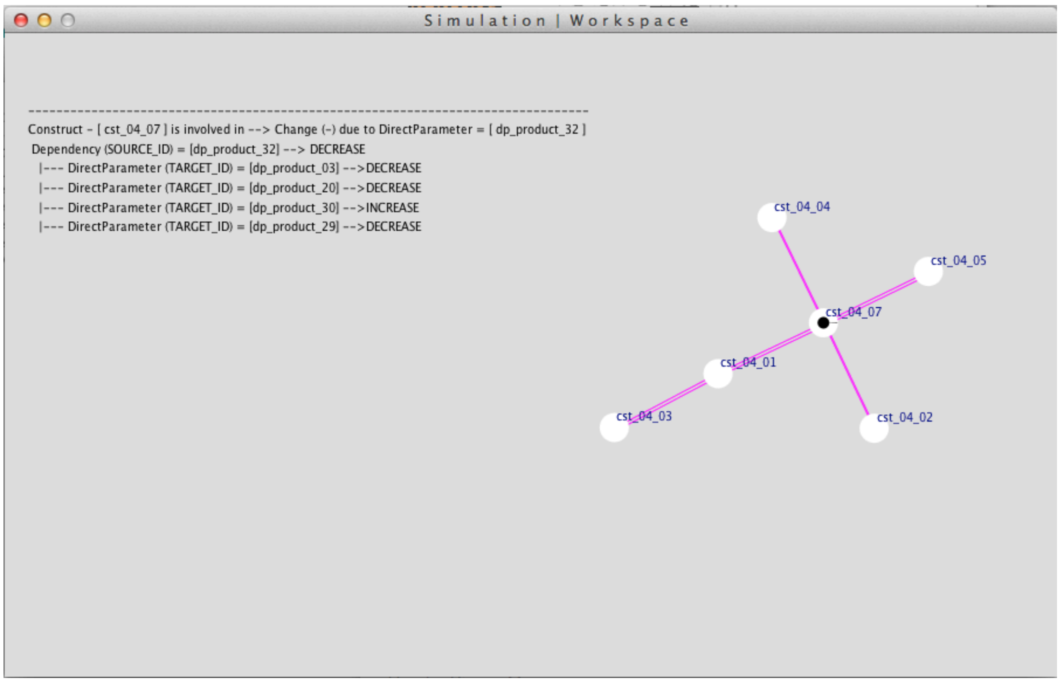
6.5.1 Verification of simulating methodology

The simulating methodology implies the working mechanism behind the prototype including two issues. The first is how the multi-layer structure construct works when an alteration is introduced. The other is how the prototype deals with the simultaneous changes.

Working mechanism of multi-layer structure construct

According to the beginning part of Chapter 5, the constructs and their relations compose a network where constructs are the nodes while the relations are the edges. When an alteration message is introduced to the network, it will be firstly broadcasted to all the constructs(nodes), staying in the construct layer. Then the constructs will forward the alteration message to its building blocks which will look up the contents of the message to find out whether they have the direct parameters related to the alteration. Finally, if having the related direct parameters, the appointed building block distributes this alteration to its specified direct parameters and this alteration will be treated inside its belonging construct. If not, the building block will report to its belonging construct, and this construct will do nothing. During the implementation, all the direct parameters have their unique id.

Take the data from Table 6.2 to test this working mechanism. Constructs Cst_04_01 to Cst_04_07 and their relations compose the network. Now a message "decrease 1mm on direct parameter and the parameter id is dp_product_32" (i.e., internal diameter of stanchion) is broadcasted to the network. Cst_04_01 to Cst_04_07 all receive this message and pass to their building blocks. Then the building blocks start to look up their direct parameters list. The product building block of Cst_04_07 finds out it contains the direct parameter corresponding to the id and executes the operation of decreasing 1 mm. All the rest constructs Cst_04_01 to Cst_04_06 will do nothing because their building blocks does contain the parameter dp_product_32. Figure 6.9 illustrates the above procedure.



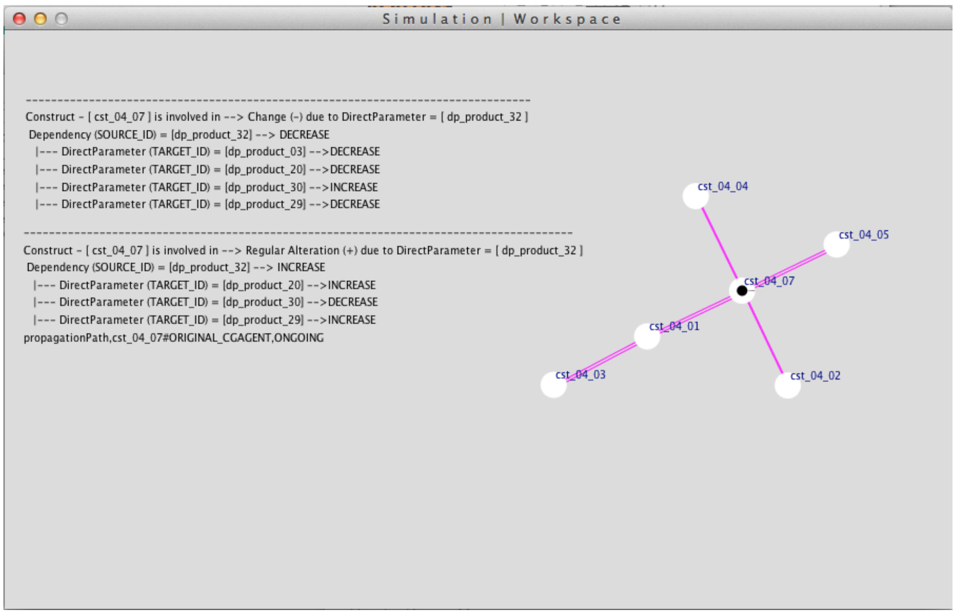
```
------------------------------------------------------------------------------
Construct - [ cst_04_07 ] is involved in --> Change (-) due to DirectParameter = [ dp_product_32 ]
 Dependency (SOURCE_ID) = [dp_product_32] --> DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_03] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_20] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_30] -->INCREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_29] -->DECREASE
```

FIGURE 6.9 – Alteration into construct

Dealing with simultaneous changes

The simultaneous changes are caused by the simultaneous alterations are finally identified within the direct parameter(s). We also use the data from Table 6.2 to test them. The tolerance and absolute tolerance of each parameter can be referred to the Table B.5 in annexe B.

There are two alterations (alt_1 and alt_2) being introduce into direct parameter "dp_product_32" (i.e., internal diameter of stanchion). One change (denoted as $change_1$ due to alt_1) makes the internal diameter of stanchion decrease 1.2mm, and the other one (denoted as $change_2$ due to alt_2) makes the internal diameter increase 1.8mm. Two results will be concluded depending on the order of treating the alterations. If alt_1 is treated first, the result of the simultaneous change is that the internal diameter become 25.6mm and then become 27.4mm (see Figure 6.10).

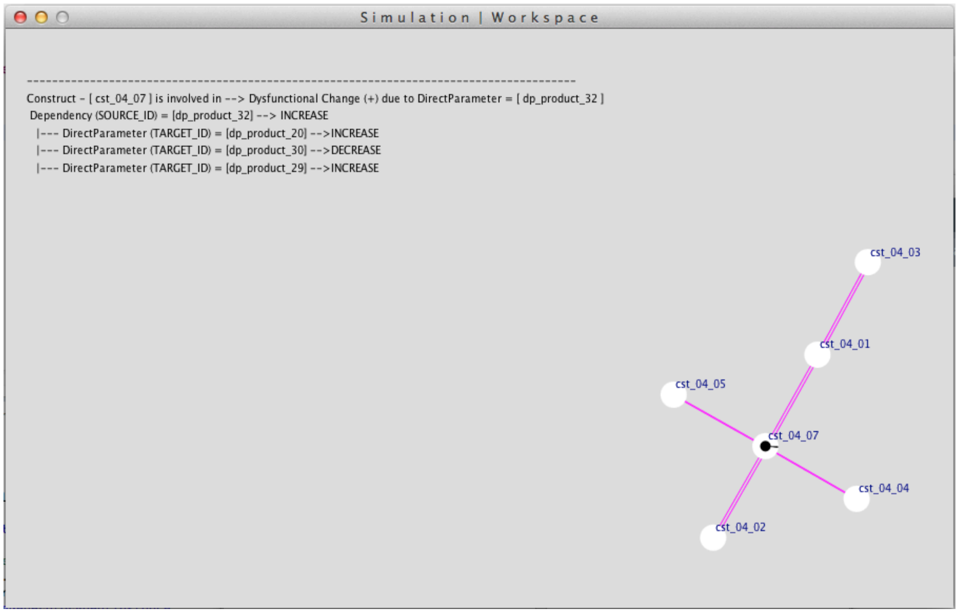


```
--------------------------------------------------------------------------------
Construct - [ cst_04_07 ] is involved in --> Change (-) due to DirectParameter = [ dp_product_32 ]
 Dependency (SOURCE_ID) = [dp_product_32] --> DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_03] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_20] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_30] -->INCREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_29] -->DECREASE

---------------------------------------------------------------------------------------
Construct - [ cst_04_07 ] is involved in --> Regular Alteration (+) due to DirectParameter = [ dp_product_32 ]
 Dependency (SOURCE_ID) = [dp_product_32] --> INCREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_20] -->INCREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_30] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_29] -->INCREASE
propagationPath,cst_04_07#ORIGINAL_CGAGENT,ONGOING
```

FIGURE 6.10 – Simultaneous changes situation 1

However, if alt_2 is treated fist, $change_2$ makes the internal diameter invalid, i.e., a dysfunctional change occurs (see Figure 6.11).



```
---------------------------------------------------------------------------------------
Construct - [ cst_04_07 ] is involved in --> Dysfunctional Change (+) due to DirectParameter = [ dp_product_32 ]
 Dependency (SOURCE_ID) = [dp_product_32] --> INCREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_20] -->INCREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_30] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_29] -->INCREASE
```

FIGURE 6.11 – Simultaneous changes situation 2

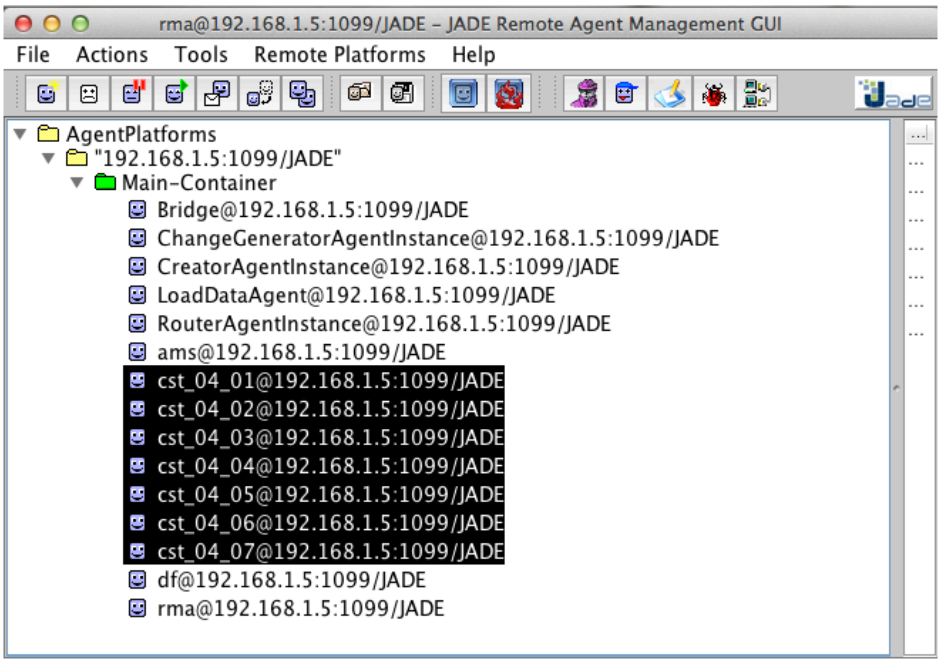
6.5.2 Verification of the prototype

The logical of the prototype is implemented by agent-based technique while the representations of the results is implemented by graphical solution. According to Chapter 5, in the construct network, the nodes(constructs) are implemented by an agent while the edges (relations/dependencies) are represented by the communications between agents. All the calculations and judgement is equipped on the agents, which we call “intelligence”, such as whether an alteration could be translated into a change, whether the change will be propagated, if propagated which construct will receive the impacts, and when a construct receives a propagated change which building block of that construct and further which direct parameter(s) of that building block will be affected.

To make the prototype executing as expected, two issues need to be ensured. The first one is wether the agents are deployed correctly. The other aspects is to check whether the prototype implements our proposed methods and models correctly. We use the data from the illustrative case to test these two issues.

Deployment of agents

According to Chapter 5, there are five types of agents that are created in the early beginning. Those agent are then registered with the container, communicate with each other and execute their tasks. Figure 6.12 show the deployment of those agents.

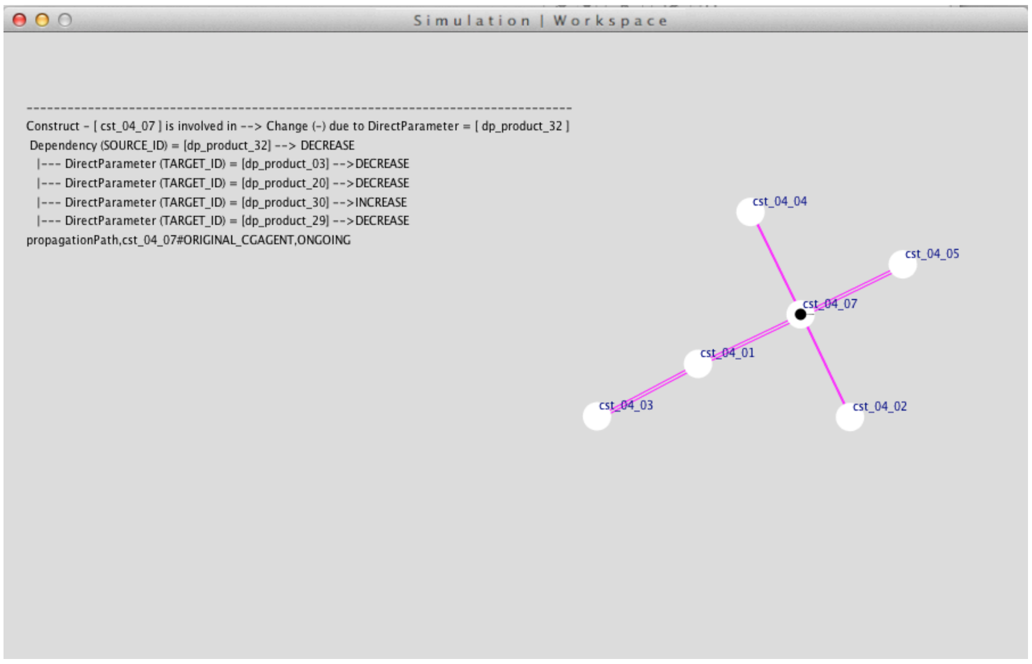

rma@192.168.1.5:1099/JADE – JADE Remote Agent Management GUI
File
Actions
Tools
Remote Platforms
Help

```
▼ AgentPlatforms
  ▼ "192.168.1.5:1099/JADE"
    ▼ Main-Container
        Bridge@192.168.1.5:1099/JADE
        ChangeGeneratorAgentInstance@192.168.1.5:1099/JADE
        CreatorAgentInstance@192.168.1.5:1099/JADE
        LoadDataAgent@192.168.1.5:1099/JADE
        RouterAgentInstance@192.168.1.5:1099/JADE
        ams@192.168.1.5:1099/JADE
        cst_04_01@192.168.1.5:1099/JADE
        cst_04_02@192.168.1.5:1099/JADE
        cst_04_03@192.168.1.5:1099/JADE
        cst_04_04@192.168.1.5:1099/JADE
        cst_04_05@192.168.1.5:1099/JADE
        cst_04_06@192.168.1.5:1099/JADE
        cst_04_07@192.168.1.5:1099/JADE
        df@192.168.1.5:1099/JADE
        rma@192.168.1.5:1099/JADE
```

FIGURE 6.12 – Deployment of agents

Results of implemention

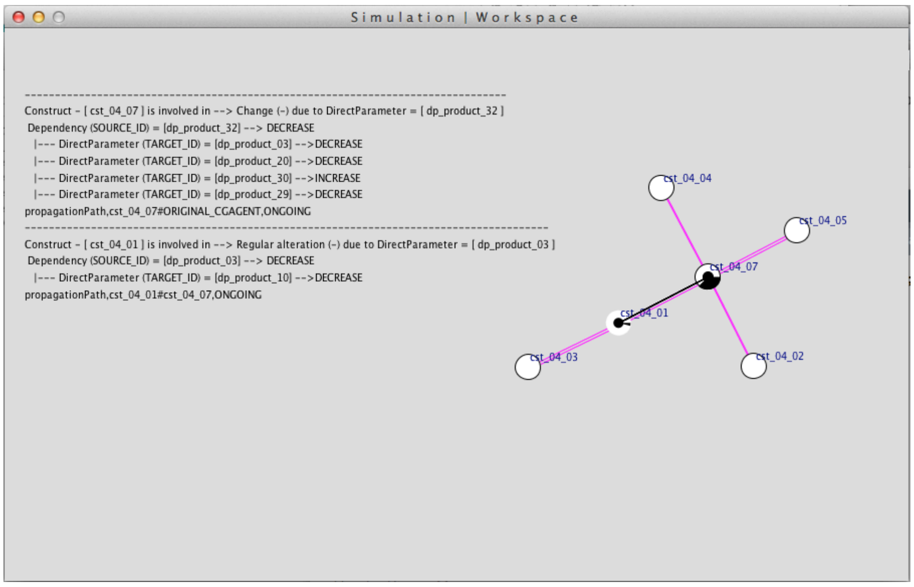
Firstly, the prototype can judge whether a change occurs and the result corresponds to calculation simply through the conceptual model. An alteration of decreasing 1.2mm on internal diameter of stanchion is introduced to the network. The agent implementing Constructs Cst_04_07 receives it and finds this alteration is related to itself. After, the intelligence on the agent (i.e., the implemented conceptual mode of identifying change occurrence) justify this alteration is a change. Figure 6.13 illustrates the result of judging change occurrence.



```
--------------------------------------------------------------------------------
Construct - [ cst_04_07 ] is involved in --> Change (-) due to DirectParameter = [ dp_product_32 ]
 Dependency (SOURCE_ID) = [dp_product_32] --> DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_03] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_20] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_30] -->INCREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_29] -->DECREASE
propagationPath,cst_04_07#ORIGINAL_CGAGENT,ONGOING
```

FIGURE 6.13 – Report of change occurrence

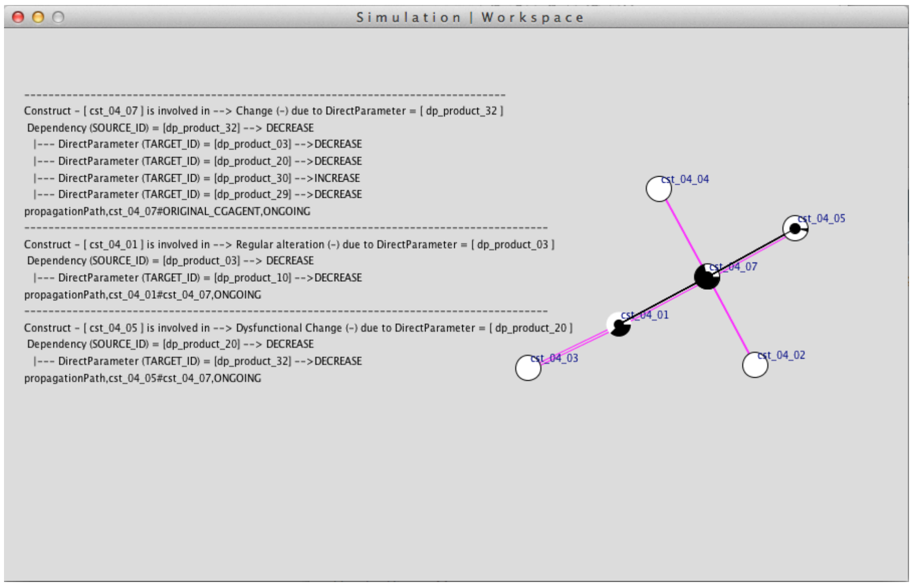
Secondly, the prototype can judge whether a change is propagated and who will be impacted by the propagations. In the same way, the results are correspond to calculation simply through the conceptual model. When the internal diameter of stanchion of Cst_04_07 decreases out of the minimum tolerance boundary value, which is estimated as a change, it causes the external diameter of front spring of Cst_04_01 to decrease, which further cause the external diameter of steerer tube of Cst_04_03 to decrease. Figure 6.14 shows the results of propagation.



```
--------------------------------------------------------------------------------
Construct - [ cst_04_07 ] is involved in --> Change (-) due to DirectParameter = [ dp_product_32 ]
 Dependency (SOURCE_ID) = [dp_product_32] --> DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_03] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_20] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_30] -->INCREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_29] -->DECREASE
propagationPath,cst_04_07#ORIGINAL_CGAGENT,ONGOING
--------------------------------------------------------------------------------------
Construct - [ cst_04_01 ] is involved in --> Regular alteration (-) due to DirectParameter = [ dp_product_03 ]
 Dependency (SOURCE_ID) = [dp_product_03] --> DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_10] -->DECREASE
propagationPath,cst_04_01#cst_04_07,ONGOING
```

FIGURE 6.14 – Result of change propagation

Lastly, the prototype can list all the possible propagation channels when a change occurs. Figure 6.15 shows the complete change propagation due to one change happens to the direct parameter *internal diameter* of stanchion.



```
--------------------------------------------------------------------------------
Construct - [ cst_04_07 ] is involved in --> Change (-) due to DirectParameter = [ dp_product_32 ]
 Dependency (SOURCE_ID) = [dp_product_32] --> DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_03] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_20] -->DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_30] -->INCREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_29] -->DECREASE
propagationPath,cst_04_07#ORIGINAL_CGAGENT,ONGOING
--------------------------------------------------------------------------------------
Construct - [ cst_04_01 ] is involved in --> Regular alteration (-) due to DirectParameter = [ dp_product_03 ]
 Dependency (SOURCE_ID) = [dp_product_03] --> DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_10] -->DECREASE
propagationPath,cst_04_01#cst_04_07,ONGOING
--------------------------------------------------------------------------------------
Construct - [ cst_04_05 ] is involved in --> Dysfunctional Change (-) due to DirectParameter = [ dp_product_20 ]
 Dependency (SOURCE_ID) = [dp_product_20] --> DECREASE
   |--- DirectParameter (TARGET_ID) = [dp_product_32] -->DECREASE
propagationPath,cst_04_05#cst_04_07,ONGOING
```

FIGURE 6.15 – Result of complete change propagations

6.6 Applications of the prototype

The agents platform and user interface communicate through TCP/IP protocol. In this case, the user can operate through user interface remotely. To use the prototype, the first step is to start the user interface which will listen to the messages that are sent by agents. The next step is to start agents. Once they are started, they will communicate, do the calculation and inform the user interface the results. Then the user interface will present the results in a graphical way with additional explanations. The followings shows the usage of the prototype one step by one step.

1. Start the user interface. Figure 6.16 is the initial user interface, which illustrate the network composed by constructs and their the effort-based relations.

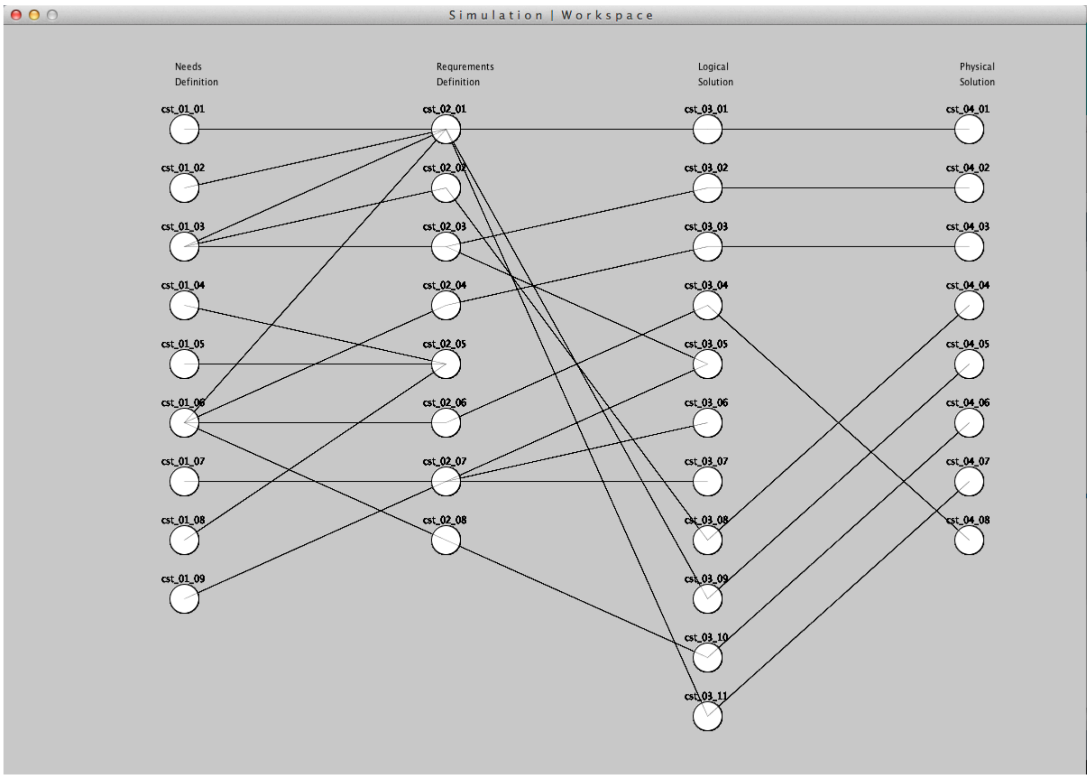
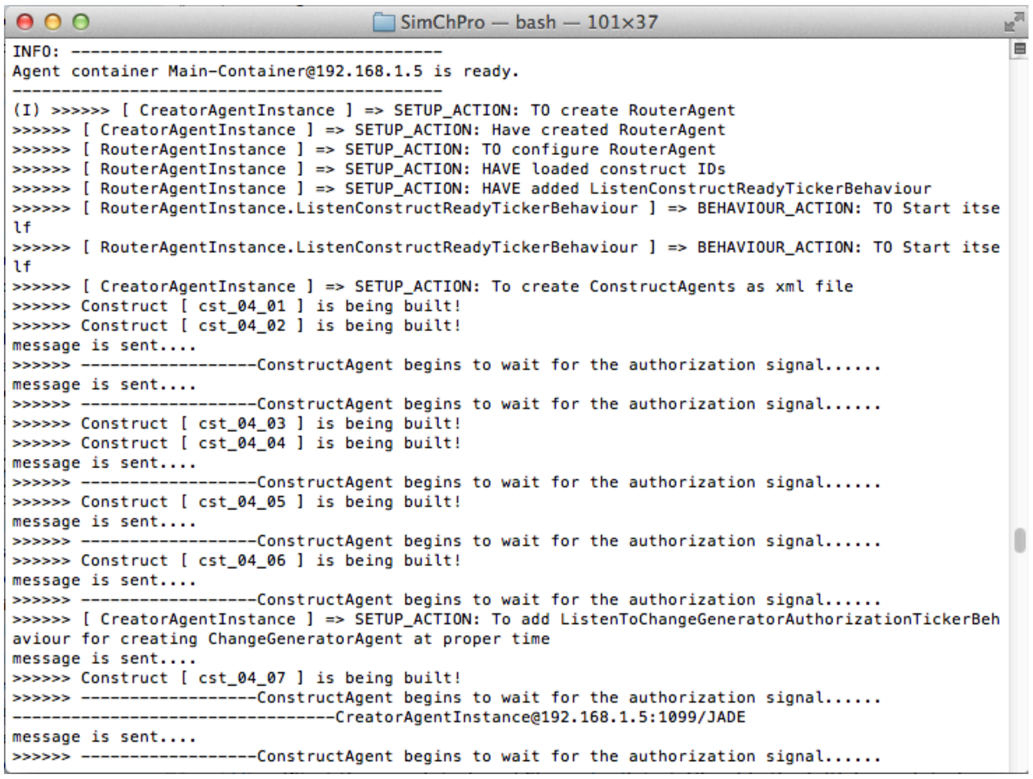


FIGURE 6.16 – Initialization of user interface

2. Start the agents. Figure 6.17 shows the interface of started agents.



```
INFO: --------------------------------------
Agent container Main-Container@192.168.1.5 is ready.
--------------------------------------------
(I) >>>>>> [ CreatorAgentInstance ] => SETUP_ACTION: TO create RouterAgent
>>>>>> [ CreatorAgentInstance ] => SETUP_ACTION: Have created RouterAgent
>>>>>> [ RouterAgentInstance ] => SETUP_ACTION: TO configure RouterAgent
>>>>>> [ RouterAgentInstance ] => SETUP_ACTION: HAVE loaded construct IDs
>>>>>> [ RouterAgentInstance ] => SETUP_ACTION: HAVE added ListenConstructReadyTickerBehaviour
>>>>>> [ RouterAgentInstance.ListenConstructReadyTickerBehaviour ] => BEHAVIOUR_ACTION: TO Start itse
lf
>>>>>> [ RouterAgentInstance.ListenConstructReadyTickerBehaviour ] => BEHAVIOUR_ACTION: TO Start itse
lf
>>>>>> [ CreatorAgentInstance ] => SETUP_ACTION: To create ConstructAgents as xml file
>>>>>> Construct [ cst_04_01 ] is being built!
>>>>>> Construct [ cst_04_02 ] is being built!
message is sent....
>>>>>> ------------------ConstructAgent begins to wait for the authorization signal......
message is sent....
>>>>>> ------------------ConstructAgent begins to wait for the authorization signal......
>>>>>> Construct [ cst_04_03 ] is being built!
>>>>>> Construct [ cst_04_04 ] is being built!
message is sent....
>>>>>> ------------------ConstructAgent begins to wait for the authorization signal......
>>>>>> Construct [ cst_04_05 ] is being built!
message is sent....
>>>>>> ------------------ConstructAgent begins to wait for the authorization signal......
>>>>>> Construct [ cst_04_06 ] is being built!
message is sent....
>>>>>> ------------------ConstructAgent begins to wait for the authorization signal......
>>>>>> [ CreatorAgentInstance ] => SETUP_ACTION: To add ListenToChangeGeneratorAuthorizationTickerBeh
aviour for creating ChangeGeneratorAgent at proper time
message is sent....
>>>>>> Construct [ cst_04_07 ] is being built!
>>>>>> ------------------ConstructAgent begins to wait for the authorization signal......
---------------------------------CreatorAgentInstance@192.168.1.5:1099/JADE
message is sent....
>>>>>> ------------------ConstructAgent begins to wait for the authorization signal......
```

FIGURE 6.17 – Start of agents

3. Introduce alterations. To introduce an alteration, the user should input the alteration contents in the program (cf. Figure 6.9).
4. Show the results. Figure 6.13, Figure 6.15 are also examples of change propagation results.

Before start the prototype, the data about constructs, building blocks and direct parameters should be input and stored into XML file. Figure 6.18 is part of the XML file that stores the information of Table 6.2.

6.7 Chapter Summary

We verified our simulation models and methods basically following the verification and validation process encompassing the conceptual model and the computerized model proposed by [Sargent2010]. We also introduced the illustrative case background and showed how to apply the case into our COCA methodology including specifying the three knowledge areas and identifying constructs. In addition, we also introduced how to use the prototype.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<dependencySet>

        <dependency id="dep_product_01">
                <KA>PRODUCT</KA>
                <source id="dp_product_06">
                        <action>DECREASE</action>
                        <sourceAlterationMinVal>15</sourceAlterationMinVal>
                        <sourceAlterationMaxVal>45</sourceAlterationMaxVal>
                </source>
                <target id="dp_product_29">
                        <action>DECREASE</action>
                        <targetAlterationValue>1*SRV-15</targetAlterationValue>
                        <targetAlterationMinVal>0</targetAlterationMinVal>
                        <targetAlterationMaxVal>30</targetAlterationMaxVal>
                </target>
        </dependency>

        <dependency id="dep_product_02">
                <KA>PRODUCT</KA>
                <source id="dp_product_17">
                        <action>DECREASE</action>
                        <sourceAlterationMinVal>0</sourceAlterationMinVal>
                        <sourceAlterationMaxVal>30</sourceAlterationMaxVal>
                </source>
                <target id="dp_product_29">
                        <action>DECREASE</action>
                        <targetAlterationValue>1*SRV</targetAlterationValue>
                        <targetAlterationMinVal>0</targetAlterationMinVal>
                        <targetAlterationMaxVal>30</targetAlterationMaxVal>
                </target>
        </dependency>

        <dependency id="dep_product_03">
                <KA>PRODUCT</KA>
                <source id="dp_product_17">
                        <action>INCREASE</action>
                        <sourceAlterationMinVal>0</sourceAlterationMinVal>
                        <sourceAlterationMaxVal>9999</sourceAlterationMaxVal>
                </source>
                <target id="dp_product_29">
                        <action>DECREASE</action>
                        <targetAlterationValue>1*SRV</targetAlterationValue>
                        <targetAlterationMinVal>0</targetAlterationMinVal>
                        <targetAlterationMaxVal>9999</targetAlterationMaxVal>
                </target>
        </dependency>

        <dependency id="dep_product_04">
                <KA>PRODUCT</KA>
                <source id="dp_product_29">
                        <action>INCREASE</action>
                        <sourceAlterationMinVal>15</sourceAlterationMinVal>
                        <sourceAlterationMaxVal>9999</sourceAlterationMaxVal>
                </source>
                <target id="dp_product_06">
                        <action>INCREASE</action>
                        <targetAlterationValue>1*SRV-15</targetAlterationValue>
                        <targetAlterationMinVal>0</targetAlterationMinVal>
                        <targetAlterationMaxVal>9999</targetAlterationMaxVal>
                </target>
        </dependency>

        <dependency id="dep_product_05">
                <KA>PRODUCT</KA>
                <source id="dp_product_29">
                        <action>DECREASE</action>
                        <sourceAlterationMinVal>0</sourceAlterationMinVal>
                        <sourceAlterationMaxVal>30</sourceAlterationMaxVal>
                </source>
                <target id="dp_product_17">
                        <action>DECREASE</action>
                        <targetAlterationValue>1*SRV</targetAlterationValue>
                        <targetAlterationMinVal>0</targetAlterationMinVal>
                        <targetAlterationMaxVal>30</targetAlterationMaxVal>
                </target>
        </dependency>
```

FIGURE 6.18 – XML data for constructs, building blocks and parameters

7 Conclusion and Further Work

▷ *This chapter concludes this thesis and summarizes the contributions of this research. Limitations of this research work are discussed along with suggestions for future works.* ◁

Plan of Chapter

7.1 Main Conclusions

In Chapter 1, five central research questions were stated that should be answered in this dissertation. The chapters answering these questions will be discussed here.

1. *What exactly is meant by change and change propagation ?*

This research question was answered in Chapter 2, where a literature review was done about change and change propagation in Product Development project. We sorted out the literature with a set of dimensions in considering the characteristics of change management and clarified the concepts of change and change propagation in our research.

2. *What is the context of change occurrence and propagation ?*

This research question was answered in Chapter 3, where the COCA framework was proposed and discussed. This framework models the context of change occurrence and propagation by taking into account the three knowledge areas at the same time (i.e., project management, product management and partnership management knowledge areas).

3. *How to identify change and change propagation ?*

This research question was answered in Chapter 4, where the conceptual models of change occurrence and change propagation were proposed. Those conceptual models provided a qualitative method to identify change and change propagation.

4. *How to simulate change propagation in an efficient way ?*

This research question was answered in Chapter 5, where the methodology of simulation and the techniques were presented. We explained how we implemented the methods and models to the prototype by using agent-based technical solution.

5. *How to represent and verify the research results ?*

This research question was answered in Chapter 5 and 6. The prototype was designed and developed with graphical representations to illustrate our research results. An illustrative case was applied to verify our methods, models and prototype.

All in all, the research described in this thesis provides a way of simulating change and change propagations in product development project. Table 7.1 summarizes the contributions of each chapter.

TABLE 7.1 – Chapter Contributions

Chapter	Contributions
Chapter 2	Literature review on change management ; Introduction of product development project modeling, agent-based techniques and computer-aided solution.
Chapter 3	Methods to model product development project ; Product evolution model to help understand how the information/data is aggregated with the project going on ; Construct identification to find out the elements where change is analyzed and propagated.
Chapter 4	Conceptual models to identify change occurrence and change propagation ; Analysis of direct parameters ; Hierarchical method used to analyze change propagation patterns in the propagation network.
Chapter 5	Analysis of the propagation network ; Methodology of simulating change propagations ; Introduction the agent-based system that translate the methodology into prototype.
Chapter 6	Introduction of how to use the prototype ; Description of the case where the methods and prototypes are verified and applied.

7.2 Research Contributions

In general, our research filled some gaps found during the literature review in change management, which simulated change propagations by considering three knowledge areas in product development project and implemented the simulation methodology in the prototype. To be more specific, four central research contributions were made.

Modeling the context of change occurrence and change propagation

We proposed COCA framework that models simultaneously product management, project management, and partnership management knowledge areas as well as the interrelations between them. Under the framework, methods to analyse the knowledge areas at different granularity levels were suggested. Meanwhile, by analyzing the interactions between the three knowledge areas, the change propagation channels were revealed. Also, under the framework, the basic elements where changes could occurre and be ananlyzed were identified.

Conceptual models of change occurrence and change propagations

The conceptual models described the procedure of identifying the occurrence of change and change propagation. The conceptual model of change occurrence enabled us to justify whether an alteration could result in change. While, according to the conceptual model of change propagation, the phenomena of change propagation could be perceived between two related direct parameters, building blocks and constructs. Therefore, a hierarchical method to identify and analyze change propagations patterns was proposed. The patterns were at three different levels, i.e., direct parameter oriented change propagation patterns, building block oriented change propagation patterns, and construct network of change propagation.

Simulation methodology of change propagation

We established a change propagation network based on constructs and their relations. In the network, each node (construct) was equipped with the intelligence of identifying, estimating and coping with changes according to conceptual models of change occurrence and change propagation. The simulation solution started with introducing the alterations. Through analyzing the alterations, whether a change emerges or not was judged. If a change emerges, the coping actions will be determined and decided whether the change would be propagated along the edges (relations between constructs). Solutions of treating simultaneous changes were also suggested during the simulation.

Prototype implementing the methods and models

We implemented the simulation methodology in the prototype by using agent-based technical solution. The nodes were realized by agents as well as the conceptual models and coping actions equipped on the nodes. The relations between nodes were realized by the communications between agent to establish the propagation channels. In a word, agent-based technical solution implemented the logics behind the prototype. In addition, to present the simulation result clearly, the prototype provided graphical user interfaces. It clearly illustrated the product evolution process, the propagation network composed by constructs and their relations, the change propagation channels when change occurs, infected constructs along the propagation channels, and propagation result on each construct.

7.3 Limitation and further Work

This research provided some limitation and it is hoped that future research will build on the findings described in this thesis. This section describes the areas of research that might be done in the future.

High complexity problem

With the complexity of problem increasing, the amount of the parameters to be treated by our approach also turns significantly higher, and then both the time and the space complexity for resolving the problem become quite high. In this situation, the performance of the technical solution might be poor. During executing the approach, we would turn to the experts for modelling the PD project. However, the performance of the approach would be affected by the quality of the PD project modelling procedures. In other words, whether the change propagation could be simulated as precisely as possible partly depends on the results provided through the modelling work from the experts.

There would exist some changes that our approach could not identify. According to our approach, we focus on the change propagations concerning the KIDs in the operational decision-making level. While referring to the change concerning the KIDs in the strategic and the tactical level, such as the organisational change, our conceptual models would not be applied.

Solving loops in change propagation channels

In Chapter 5, when dealing with simultaneous changes, we mentioned that any loop in the graph of change propagation channels might cause a resonance-like effect. The effect could amplify the influence of occurred changes during the change propagations and also reflect some too-high couplings existing the elements of the PD project. In this thesis, we eliminated loops in change propagation channels. To deal with loops requires more efforts in understanding the reasons causing loops, the dependencies between constructs/building blocks/direct parameters, the propagated influences along the change propagation channel of loop. Therefore, one aspect of the future work could be to do a deep research on loops in change propagation channels and to provide a solution to solve this problem.

Case studies

The research results and the prototype have been applied into an illustrative case and have been verified during the application. However, there is a lack of validation on the results and the prototype through case studies. Limited to timescale and resources during the research, it is difficult for us to collect real cases from industrial fields.

Therefore, we hope in the future the research results and the prototype can be applied and validated through case studies. On one hand, this enable us to collect the feedbacks about how the results help the companies during their change management process. On the other hand, user tests regarding the prototype can reflect whether the user interfaces are friendly or not, and whether the operations on the prototype are easy to execute or not. To achieve this purpose, it needs our efforts to contact the companies and to persuade them to use our methods and prototype.

Visualization comparison

During the visualization of the research results, what we considered was just to provide a graphical representation and we did not do a comparison between different types of representations. Therefore, the algorithm of representing the results in our research only provided a single view of the change propagation network. In fact, there are different ways to present the same result, which bring different vision effects. Taking tree visualisation techniques for example, there are standard trees, tree-map, radial tree layout and sunburst.

Therefore, the future work can do a research on visualization techniques and compare them to find out a more proper way of representing the research results. In addition, the future work can incorporate a multiple view strategy.

User interface improvement

In this thesis, we provided user interfaces to show the simulation results. However, there was no graphical interface for other users to input the information about the product development project such as the direct parameters, building blocks, constructs, and the dependencies. We recorded those information in XML files, which requires the users have some basic knowledge about XML. Therefore, to provide graphical user interface to input the information could be one aspect of the future work.

A
Highlighted Reviewed Literature

TABLE A.1: Reviewed on literature on change management

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
Shiau & Wee(2008)	Product	Proc	- develop a distributed change control workflow change handling	- closed-loop change control process ; -distributed routing algorithm	article
Singh & Shoura (2006)	Project	Proc	- provide a approach to analyze the process of change ; - make managers better understanding their responsibilities in managing change	- Organizational Change (OC) model ; - questionaire	article
Ainscough et al. (2003)	Product, Project	Pre	- suggest how to deploy concurrent engineering within the organization through the management of change	- concurrent engineering self-assessment model ; - Implementation Strategy Tool ; - Generic Planning and Guidance Tool	article ; tool validated in a company
Kocar & Akgunduz (2010)	Product	Proc	- propose an approach for processing engineering changes smart user support for predicting engineering changes model engineering change management process	-Active Distributed Virtual Change Environment ; - Sequential Pattern Mining Techniques ; - Change Propagation mechanisms	article ; computer support system
Oliverira& Pinheiro (2009)	Project	Pre	- present best practices with respect to the management of organizational change due to the implementation of ISO 14001	- best practices for the implementation of ISO 14001 ; - ISO 14001	article ; case study
Chen & Thimm (2008)	Product	Pre	- model associative engineering relations between changes ; - propose the information consistency control algorithm	- method of modeling change relations ; - change propagation algorithm ; - JTMS-based dependency network	article

Continued on next page (*IKA : Involved Knowledge Area, FM : Focus Moment, Diss : Dissemination*)

TABLE A.1 – continued from previous page

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
Eckert et al. (2004)	Product	Proc	- analyze the formal and informal process used to handle change ; - categorize changes as emergent change and initiated change ; - list the causes for problems with change ; - categorize types of change propagations ; - contribute strategies to cope with changes	case study in a focal company ; - organization of change processes in the focal company	article ; case study
Wolfgang et al. (2007)	Product, Project	Pre	- propose criteria in change decisions ; - suggest decision-making approach in change decisions ; - identify the alternative patterns of decision making	- multiple-case study ; - operative and strategic analysis of change management	article
Rouibah & Caskey (2003)	Product, Project, Partnership	Proc	- propose a parameter-based approach of identify the impact of design changes in early phase during the collaborative product development ; - identify the lack of engineering change management between companies	- literature review to discover the research gap ; - parameter value evolution reflecting the design team interaction and the parameter maturity ; - formalized parameter network	article ; product data management system
Fei et al. (2011)	Product	Proc, Post	- present a methodology to trace, analyze and evaluate engineering changes in product design ; - adopt a modeling method to enhance the traceability of potential design changes ; - develop a matrix to analyze change propagation based on functional and physical models of design ; - develop a reasoning methodology based on knowledge to resolve design conflicts	- Matrix-based analysis ; - ontology	article

Continued on next page (*IKA : Involved Knowledge Area, FM : Focus Moment, Diss : Dissemination*)

TABLE A.1 – continued from previous page

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
Arnaud et al. (2002)	Product	Pre	- present an overview of engineering change management for complex products ; - identify potential causes and consequences of engineering change ; - propose an approach to design and implement a measurement plan for engineering change processes ; - propose indicators for engineering change processes	- classification of engineering change kinds ; - generic engineering change process	article
Raineri (2009)	Project	Pre	- analyze the differences between the managerial practices of organizational change processes ; - conclude that : 1. more change strategists than change receptors use change management practices ; 2. firms use the practices related to change propagation stage more frequently ; 3. using change management practices has a significant impact on the accomplishment of the change program objectives and deadlines	-case study in 90 organizations ; - statistical hypothesis test methodology	article
Gareis (2010)	Project	Pre	- categorize change types, change objects ; - develop change management models and the interpretations	- hypotheses ; - case study in four organizations	article
Huang & Mak (1998)	Product	Pre	- identify the gap between what is used and what is available for product change management ; - establish a set of characteristics of computer-aids for change management to assess existing systems ; - guide developing new systems	- survey ; - CAECC systems	article

Continued on next page (*IKA : Involved Knowledge Area, FM : Focus Moment, Diss : Dissemination*)

TABLE A.1 – continued from previous page

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
Lehmann (2010)	Project	Pre	- identify the gap between conceptualizations in change management and in project management ; - structure the fields of project and change management ; - propose a guide to investigate management of changes as project	- “mixed model” approach to study how management evolved in the project management and the change management fields	article
Fricke et al. (2000)	Product, project, partner	Pre, Proc	- describe the causes and reasons for changes ; - propose five strategies to cope with change as well as the related methods	- five attributes to implement a better change management	article
Huang et al. (2003)	Product	Pre	- investigate the current state of engineering change problems ;- examine the present industrial practices in managing engineering changes ;- report that : 1. engineering change is a noticeable problem ; 2. engineering change management is unsatisfactory in the surveyed companies ; - state that : 1. there are a number of computer software packages supporting engineering change management ; 2. several national and international standards are available for engineering change management and configuration management	- interviews with four companies	article
Habhouba et al. (2011)	Project	Proc	- propose a collaborative tool to assist designers and experts to manage change ; ensure the coherence of data between the various disciplines ; - assist experts in making decisions by proposing alternative solutions	- Multi-Agent System	article ; software prototype

Continued on next page (*IKA : Involved Knowledge Area, FM : Focus Moment, Diss : Dissemination*)

TABLE A.1 – continued from previous page

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
Fricke & Schulz (2005)	Product	Pre	- incorporate changeability into a system architecture ; - define four key aspects of changeability ; - propose the design principles to enable the four key aspects within systems	- Design Structure Matrix	
Rios et al. (2007)	Project	Pre	- propose a methodology to analyze the cost impact for change in requirements	- axiomatic design principles matrices ; -Work Breakdown Structure (WBS)	article
Huang et al. (2001)	Product	Pre, Proc	- propose methods to design, develop and implement a web-based framework supporting engineering change management	- web-based software and hardware architecture	article ; software prototype
Tavcar & Joze (2007)	Project	Proc	- analyze the complexity and the design level of product ; - optimize the engineering change process	- four characteristic levels of design during PD ; - generalized model of engineering change process ; questionnaire for assessing engineering change process	article
Tang et al. (2010)	Product	Pre	- trace the design knowledge to improve engineering change management with DSM-based methodology ; - predict change propagations through capturing knowledge ; - identify type and level of engineering change	- Design Structure Matrix (single-domain and multiple-domain DSMs) ; - knowledge classification	article ; software prototype
Gerst et al. (2001)	Product	Pre	- analyze possible change on the existing product and their impact ; - suggest to consider a trade-off between fulfilling requirements and the impact from changes	- stable balance model between product properties and product requirements ; - classification of requirements ;	article

Continued on next page (*IKA : Involved Knowledge Area, FM : Focus Moment, Diss : Dissemination*)

TABLE A.1 – continued from previous page

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
Mohan et al. (2008)	Product, Project	Proc	- integrate the traceability and software configuration management to support change management ; - provide guideline to project manager and developers for documenting changes and their impact	- a traceability model representing knowledge element ; - a tools supporting the integrated practice of software configuration management and traceability	article ; software tool
Terwiesch & Loch (1999)	Project	Pre	- identify five key contributors to long engineering change order lead time ; - classify four principles of engineering change order management ; - outline strategies to reduce engineering change order time	- Engineering change order process ; - on site case study for over four months	article
Rutka et al. (2006)	Product, project, partner	Pre	- simulate and analyze the change propagations in an engineering product, process and/or organization ; -capture dependencies between multiple viewpoints (time, cost and resources)	- Change Propagation Analysis ; - Decision criteria impact analyses	article
Dvir & Lechler (2004)	Project	Pre	- study the interactions between planning variables and their influences on project success ; - identify contextual variables affecting the planning process ; - report that goal changes and plan-changes bring more effects to the project success than the quality of planning	-structural equation model	article
Clarence & Hoon (2001)	Project	Proc	- introduce a comprehensive project change management systems ; - identify five principles for project change management	- two-level process model of change management system	article

Continued on next page (*IKA : Involved Knowledge Area, FM : Focus Moment, Diss : Dissemination*)

TABLE A.1 – continued from previous page

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
Eckert et al. (2005)	Product	Pre	- identify the problems in predicting change propagation ; - identify and analyze the predictability of design change ; - propose a model distinguishing the static background for design from the actions to effect design change ; - propose a change propagation model ; - identify strategies for change prediction	- empirical studies ; -Design Structure Matrix	article
Clarkson et al. (2001)	Product	Pre	- analyze change behavior by case study ; - outline and evaluate a change prediction method ; - propose the mathematical models to predict the risk of change propagation	- risk Design Structure Matrix ; - empirical study ; - risk assessment ; - change propagation tree	article ; software prototype
Keller et al. (2007)	Product	Pre	- compare two product models ; - analyze two approaches of predicting product change and suggest complements ; - propose the combined strategies of using the two approaches	- CPM product model ; - C&CM product model	article
Do et al. (2008)	Product	Proc	- propose a procedure for engineering change propagation ; - analyze engineering changes propagation to product data views ; - propose an integrated product data model for change propagation procedure	- product data model to support engineering change propagation ; - engineering change propagation procedure	article ; software prototype
Jordan et al. (2006)	Product, Project	Pre	- explore changes in technical, political, or economic environment, which necessitate design modifications or upgrades ;- conclude that flexibility is a key attribute to the design of many complex engineering systems with long design lifetime	- later-freeze of requirements ; -case study	article

Continued on next page (*IKA : Involved Knowledge Area, FM : Focus Moment, Diss : Dissemination*)

TABLE A.1 – continued from previous page

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
You & Chao (2008)	Product	Proc, Post	- adopt the Macro Type Passing Data (MTPD) and neutral format files to record design procedures ; - use similarity comparison to identify the differences between the respective models from industrial design and mechanical design ; - identify the design change propagations through discovering the differences between the models	- MTPD ; - similarity comparison	article
Eckert et al. (2006)	Product	Pre	- characterize product change by case study ; - introduce tools to help understand the potential change effects with probabilistic prediction and visualization of change propagation	- case study ; - change propagation tree ; - DSM	article
Saeedipour & Stevenson (1998)	Product	Proc	- analyze the interaction between design parameters ; - present a linear method to formulate the effect of changes to the specification		article
Keller et al. (2005)	Product	Pre	- introduce different views on change propagation ; - show how the data is visualized using CPM tool ; - suggest the suitable graphs to display different change propagations data	- CPM tool ; - DSM ; - Propagation networks and trees ; - Change Risk Plot	article
Reddi & Moon (2009)	Product	Proc	- present a framework to manage engineering change propagation ; - identify the affected components due to engineering change	- DSM ; - Object-Oriented concepts	article ; software prototype

Continued on next page (*IKA : Involved Knowledge Area, FM : Focus Moment, Diss : Dissemination*)

TABLE A.1 – continued from previous page

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
Do et al. (2002)	Product	Proc	- represent product data to express and enforce integrity constraints on product structure during engineering changes ; - extend an approach to manage engineering change data ; - describe a prototype product data management systems	- structure-oriented approach ; - web-based PDM system	article ; software prototype
Mehta et al. (2010)	Product	Proc, Post	- present an approach to compute similarity between engineering changes ; - define engineering changes by disparate attributes ; - identify the semantics associated with the attribute ; - aggregate the similarities between attribute values to compute the similarity	- features of similarity ; - semantics analysis	article
Conrad et al. (2007)	Product	Pre	- present an approach to support to analyze and assess the change effects in product development process	- CPM/PDD theory ; - Failure Modes and Effects Analysis method	article
Eger et al. (2007)	Product, project, partner	Pre	- discuss the links between the product, process and people during PD ; - list the factors causing risky change implementation and higher change cost ; - present a tool to evaluate change proposals during design processes	- CPM ; - design freeze	article ; software prototype
Ariyo et al. (2006)	Product	Pre	- identify a model accommodating different types of component interactions to generate prediction of change propagation ; - list the conditions in which change will propagate between components	- CPM ; - FBS	article

Continued on next page (*IKA : Involved Knowledge Area, FM : Focus Moment, Diss : Dissemination*)

TABLE A.1 – continued from previous page

Citation	*IKA*	*FM*	*Main Contributed Research Issue*	*Key Technology*	*Diss*
Ariyo et al. (2006)	Product	Pre, Proc	- describe an algorithm of predicting change propagation ; - propose a multilevel approach of computing change propagation likelihood considering the change probability levels	- product connectivity model ; - DSM ; - CPM	article
Ouertani (2008)	Project	Proc	- propose a solution to assess the impact of change through data dependency network ; - quantify key issues to enable better design process management ; - re-organize the execution of design activities	- DEPNETproduct specification DEPendencies NETwork identification and qualification ; - data dependencies network	article

B
Illustrative case

▷ *The complementary data and information of the illustrative case used for the verification and application are presented in this annexe.* ◁

Plan of Chapter

B.1 Components of Full Suspension System

We introduce here each of the modules/components of the full suspension system.

Front fork In the case, we specify the spring and the damping as two end component, though they could be still taken as systems including sets of smaller components. Thus, the components for adjusting the customized performance of front suspension are excluded from our consideration. In this way, the front fork consists of following components (see Figure B.1).

– Steerer
– Crown
– Cap-top
– Seal
– Stanchion
– Lower leg
– Damping
– Spring
– Front axle

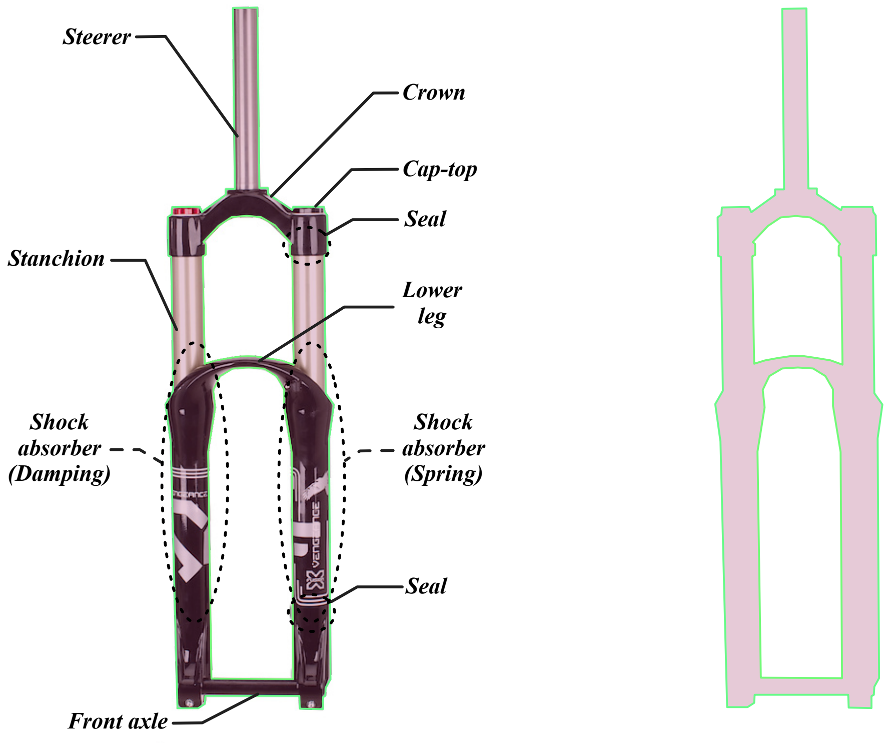


FIGURE B.1 – Front fork

Front section

The front section is one of the modules in frameset and in a triangle layout (also called main triangle), and the front wheel and the front fork are fitted on it (see Figure B.2). The front section consists of four tubes :

- Top tube
- Seat tube
- Down tube
- Head tube

Rear swingarm

The rear swingarm controls the range in which the suspension can move, and it is connected with the front section (see Figure B.2). The rear wheel is fitted on it. Compared with the triangle layout (also called paired rear) within traditional bike frame that is without suspension system, we can also find the formal components mentioned in the traditional triangle layout.

- Seat stay
- Chain stay
- Rear axle

Rear shock

The rear shock is installed in line with the seat stay and connects the swingarm with the front section (see Figure B.2). It can help to reduce vibration and small bumps during riding. It usually contains a spring and damper.

- Spring
- Damper

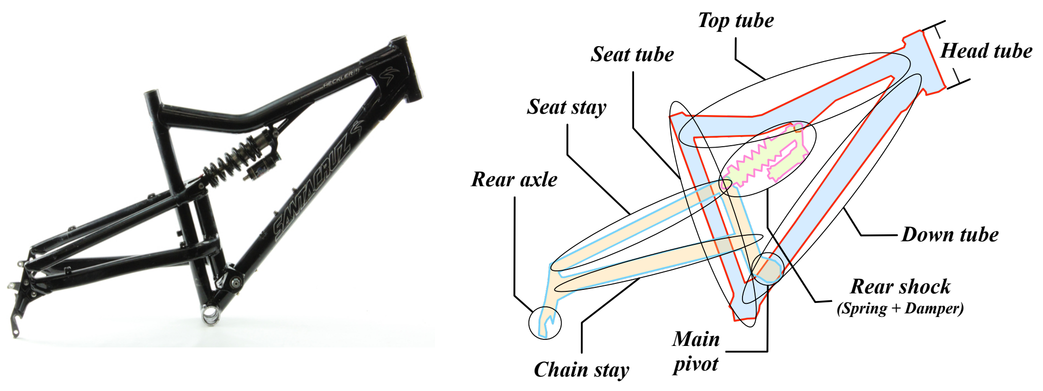


FIGURE B.2 – Frameset

B.2 Complementary Data for Project Management Knowledge Area

Hereafter, we give a description of the PD project. This is not a typical project decomposition. There are many interactions and back/forward actions in order to progress in the project. Not all of the interactions are shown in order to maintain the project simple enough for a good understanding.

B.2.1 Task description

Phase 1 : planning

Task 1 : identify market opportunity

The focal company begins a PD project with investigating the market to collect and identify the potential business opportunity for design a product.

Upstream Task : null

Downstream Task : Task 2, Task 4

Task 2 : prepare Task/resource plan

With the identified business opportunity, the focal company prepares the plan for organizing the relevant resources (such as human month, estimated budget, etc.) and the schedule of the concerned activities. The plan is to be considered by the focal company to decide whether the corresponding project would be established as planned. With the prepared plan, the focal company then starts to consider some supply chain management issues (i.e., concerned in Task 7) in further.

Upstream Task : Task 1, Task 5

Downstream Task : Task 3, Task 7

Task 3 : approve resource and plan timing

With the prepared plan of arranging activities and assigning the resources, the focal company considers it and then publish the official agreement on it.

Upstream Task : Task 2, Task 6, Task 20

Downstream Task : Task 5

Task 4 : establish technical and economic documents

Given the identify business opportunity of the potential product and the project objectives/mission, the technical and economic documents are composed to analyze the feasibility of design the product through the project from multiple aspects (mainly concerning technical and economic issues). The documents are to support the focal company determine the project quality objectives.

Upstream Task : Task 1, Task 5, Task 6

Downstream Task : Task 5

Task 5 : determine project quality objectives

Based on the approved Task/resource plan, the project quality objectives are created and determined, in which the project management issues are concerned, such as the constraints from time, cost and/or quality aspects. The result of this Task is to be considered by the focal to decide whether the project mission is accepted. Through this Task, the possible iterations of adjusting Task/resource plan (i.e., Task 2) and the technical/economic documents (i.e., Task 4) would be executed.

Upstream Task : Task 3, Task 4

Downstream Task : Task 2, Task 4, Task 6, Task 11

Task 6 : approve project mission

The focal company considers the project mission and published the official agreement on it. During the consideration, the resource/Task plan (i.e., concerned in Task 3) and the technical/economic documents (i.e., concerned in Task 4) would be adjusted iteratively. Once the project mission is approved, the further Task of gathering information/data from the potential customers (i.e., Task 10) is started. The project mission also supports the focal company to establish the project plan (i.e., Task 15).

Upstream Task : Task 5, Task 20

Downstream Task : Task 3, Task 4, Task 10, Task 15

Phase 2 : concept development

Task 7 : execute advanced supplier quality audits

After preparing the Task/resource plan, the focal company execute advanced supplier quality audits supporting to create the potential supply chain for the product development project. Through the audition, the potential suppliers are discovered for the further identification.

Upstream Task : Task 2, Task 13

Downstream Task : Task 8

Task 8 : identify suppliers

With the discovered potential suppliers, the focal company identifies some of them for building up the further collaboration in the project.

Upstream Task : Task 7

Downstream Task : Task 9

Task 9 : categorize suppliers

The identified suppliers are categorized into a set of groups according to the focal companys criteria, which could include the criticality of the suppliers, the products they provided, the experienced relationships, etc.

Upstream Task : Task 8

Downstream Task : Task 14

Task 10 : gather information/data from customers

Once the project mission is approved, the focal company also gather the information/data from the customers in order to learn their expectations/needs onto the final product.

Upstream Task : Task 6

Downstream Task : Task 11

Task 11 : establish target specifications

With the project objectives as well as the customers expectations/needs, the focal company creates the target specifications as the answer though neither has the product concept been establish nor is the target specification too arbitrary to be technically feasible. The target specifications will also used to compose RFQ (Request For Quotation).

Upstream Task : Task 5, Task 10, Task 13

Downstream Task : Task 12

Task 12 : send RFQ to suppliers

Based on the target specifications, the focal company sends RFQ to the identified suppliers in order to check the matching of quotes to the specifications (i.e., concerned in Task 13).

Upstream Task : Task 11

Downstream Task : Task 13

Task 13 : check matching of quotes to specifications

With the reply of answering the RFQ from the suppliers, the focal company checks matching of quotes by the suppliers to the specifications. The obtained result would support the focal company to select the critical suppliers (i.e., concerned in Task 14) in the project and establish the project plan (i.e., concerned in Task 15). If necessary, such as not all the specifications can be matched with the quotations from the concerned suppliers, the focal company would either adjust the specifications or re-execute the supplier quality audition.

Upstream Task : Task 12

Downstream Task : Task 7, Task 11, Task 14, Task 15

Task 14 : select critical suppliers

From the category of the identified suppliers (i.e., concerned in Task 9) as well as based on the checked matching of quotes to the specifications (i.e., concerned in Task 13), the focal company selects a set of critical suppliers according to the its evaluation criteria, such as the timing of participation of suppliers, the quality/performance of the supplied product and the suppliers, whether sharing risk/revenue, etc. The result of selecting the critical suppliers would support the focal company to establish the project plan (i.e., Task 15).

Upstream Task : Task 9, Task 13

Downstream Task : Task 15

Task 15 : establish project plan With the selected critical suppliers, the checked matching of quotes to the specifications, and the approved project mission, the focal company establishes the project plan to guide the the execution of project. Through this Task, the objectives of the project are documented with the plan concerning what the goals would be achieved, who would be involved, what would be the responsibilities of the stakeholders, when a task would be started and completed, what are the meaningful decision points, etc.

Upstream Task : Task 6, Task 13, Task 14, Task 16, Task 18

Downstream Task : Task 16, Task 17, Task 19

Task 16 : approve project plan

With the project plan, the focal company evaluates it and then publishes the official agreement on it. After approving the project plan, the focal company then begins to establish the final specifications of the product (i.e., concerned in Task 17). If the project plan would not be accepted, then it would be re-established (i.e., concerned in Task 15).

Upstream Task : Task 15, Task 27

Downstream Task : Task 15, Task 17

Task 17 : establish final specifications

After approving the project plan, the focal company re-considers the specifications (i.e., released through Task 11) and finalizes them with the various trade-offs (such as technological constraints,production costs, etc.), i.e., establishes the final specifications.

Upstream Task : Task 15, Task 16, Task 18

Downstream Task : Task 18

Task 18 : approve final specifications

With the final specifications of the product, the focal company evaluates it and then makes the official agreement on it. The approved final specifications support the focal company to establish the product architecture (i.e., concerned in Task 19) and assess some detailed scheduling and risk in further (i.e., concerned in Task 22). If the final specifications would not be accepted, then it would be re-composed (i.e., concerned in Task 17) or even lead to re-considering the project plan.

Upstream Task : Task 17

Downstream Task : Task 15, Task 17, Task 19, Task 22

Phase 3 : system-level design

Task 19 : specify major sub-systems and interactions

Based on the final specifications as well as the constraints from the project plan, the major sub-systems of the product are specified in order to establish the product architecture and assessed with the detailed scheduling and risks (i.e., concerned in Task 22) in further. Meanwhile, the interactions between the sub-systems are also identified.

Upstream Task : Task 15, Task 18

Downstream Task : Task 20, Task 21, Task 22

Task 20 : authorize proceed to next phase

After establishing the product architecture, the focal company evaluates the obtained results and makes the decision whether the project would proceed to the next phase or iterate to the previous activities. If the project is agreed on being continued to the next phase, the authorization would be made as a explicit decision point.

Upstream Task : Task 19

Downstream Task : Task 3, Task 6, Task 21

Task 21 : establish product architecture

Based on the specified sub-systems as well as their interactions, the focal company starts to assign the functional elements to the physical elements (components, modules) of the product and define the interactions between the modules and/or components, i.e., establishing the product architecture. The established product architecture is critical for the focal company to make decisions concerning the product changes, market strategies, and product development management, etc. It also supports the focal company to estimate the manufacturing costs (i.e., concerned in Task 28) in further.

Upstream Task : Task 19, Task 20

Downstream Task : Task 26, Task 28

Phase 4 : detail design

Task 22 : assess detailed scheduling and risk

With the specified sub-systems, the focal company assesses the detailed scheduling and the risks of implementing each of them. The assessment is used to establish and evaluate the technical requirements of the sub-systems (i.e., concerned in Task 23, Task 24) as well as review the design solution (i.e., concerned in Task 26) in further.

Upstream Task : Task 18, Task 21, Task 26

Downstream Task : Task 23, Task 24, Task 26

Task 23 : establish technical requirements of sub-systems

Based on the specified sub-systems and the assessment, the technical requirements of the sub-systems support the focal company to established the implementations in further.

Upstream Task : Task 22

Downstream Task : Task 24

Task 24 : approve sub-systems requirements

With the technical requirements of the sub-systems as well as the assessment of them, the focal company evaluates them and then makes the official agreement.

Upstream Task : Task 22, Task 23, Task 26

Downstream Task : Task 25

Task 25 : establish design solution to the requirements

Referring to the approved requirements, the focal company establishes the corresponding design solutions as the implementations.

Upstream Task : Task 24

Downstream Task : Task 26

Task 26 : review design solution

The proposed design solutions are reviewed in association with the product architecture, the assessment of designing sub-systems by the focal company to evaluate the design performance and determine whether the design process is complete. Through the review, the focal company would adjust the assessment of scheduling/risk (i.e., concerned in Task 22), re-analyze the sub-system requirements (i.e., concerned in Task 24), and/or re-work onto the design solutions (i.e., concerned in Task 25).

Upstream Task : Task 19, Task 22, Task 25

Downstream Task : Task 22, Task 24, Task 25, Task 27

Task 27 : authorize that the design solution is frozen

According to the result of reviewing the design solution, the focal company authorizes the design solution is frozen which implies that any change to the design would be rejected. With the authorization, the focal company would continue the project and turn to consider testing the product in further. If the authorization could not be made, then the project plan would be adjusted correspondingly.

Upstream Task : Task 26

Downstream Task : Task 16, Task 30

Task 28 : estimate manufacturing costs

Based on the frozen design solution, the focal company estimates the manufacturing costs for the further manufacturing process. The estimation also supports the focal company to establish the test plan of the product prototype.

Upstream Task : Task 22

Downstream Task : Task 30

Phase 5 : testing and refinement

Task 29 : identify control factors and performance

In order to test the prototype of the product, the focal company identifies a set of control factors in order to measure the performance of the prototype.

Upstream Task : Task 34

Downstream Task : Task 30

Task 30 : establish test tool and test plan

With the identified control factors and performance as well as the estimation of manufacturing costs, the focal company establishes the test tool and test plan when the design solution is frozen.

Upstream Task : Task 27, Task 28, Task 29

Downstream Task : Task 31, Task 32

Task 31 : verify test tool

With the established test tool, the focal company verifies the test tool in order to support to establish the prototype.

Upstream Task : Task 30

Downstream Task : Task 33, Task 34

Task 32 : approve test plan

With the established test tool and the test plan, the focal company considers the test plan and releases the official agreement on it.

Upstream Task : Task 30

Downstream Task : Task 33, Task 34

Task 33 : establish prototype

Based on the test tool and the test plan, the focal company establishes the prototype in advanced which would be used to test, validate and verify the design solution.

Upstream Task : Task 31, Task 32, Task 34

Downstream Task : Task 34

Task 34 : approve prototype

With the established prototype, the focal company evaluates the prototype with the test tool according to the test plan. If the prototype performs as the focal company specified, then the prototype is approved. Through this Task, the focal company would identify some other control factors and/or establish other prototype(s) to resolve some issues and/or improve the performance closer to the final product.

Upstream Task : Task 31, Task 32, Task 33

Downstream Task : Task 29, Task 33, Task 35

Task 35 : validate the design solution

Through adopting the prototype(s), the focal company validates the design solution of the product.

Upstream Task : Task 34

Downstream Task : Task 36

Task 36 : deliver design solution

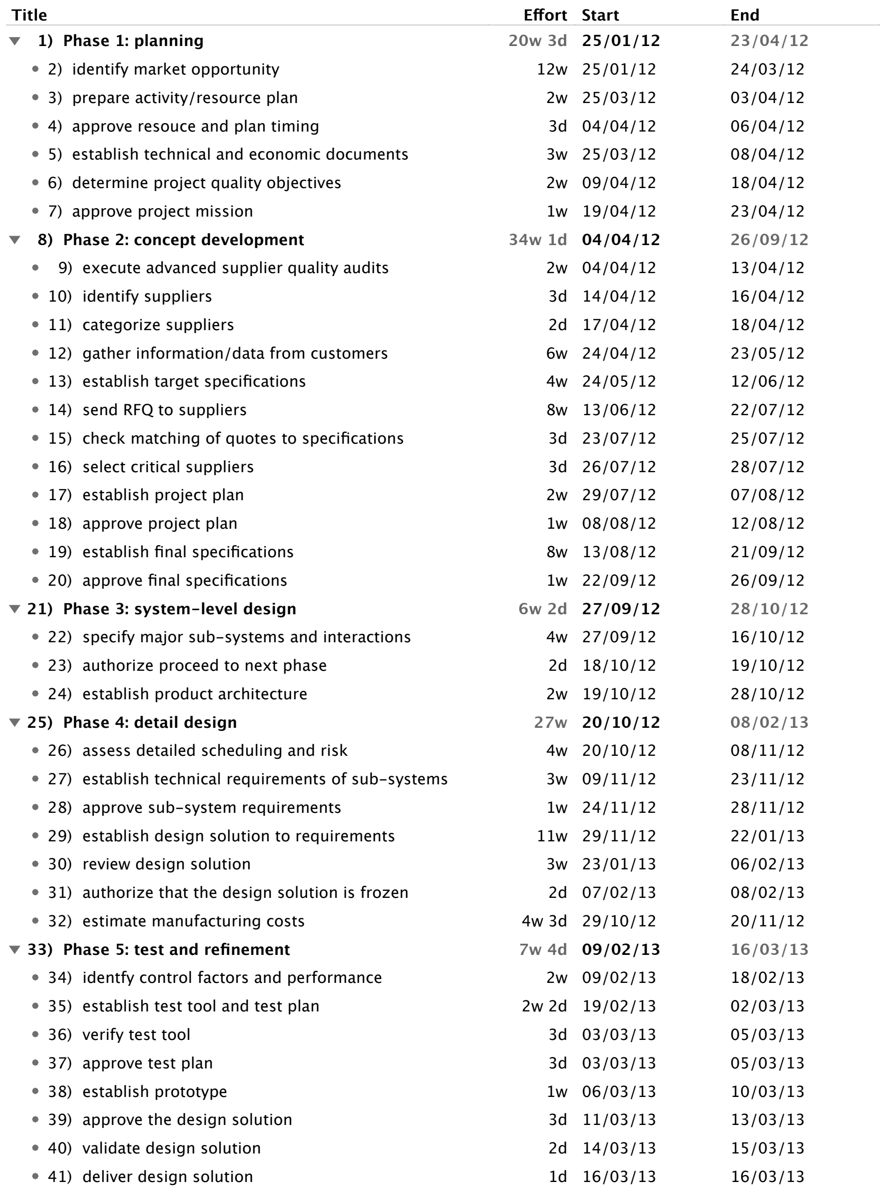
After validating the design solution, it is delivered to the next process for manufacturing.

Upstream Task : Task 35

Downstream Task : null

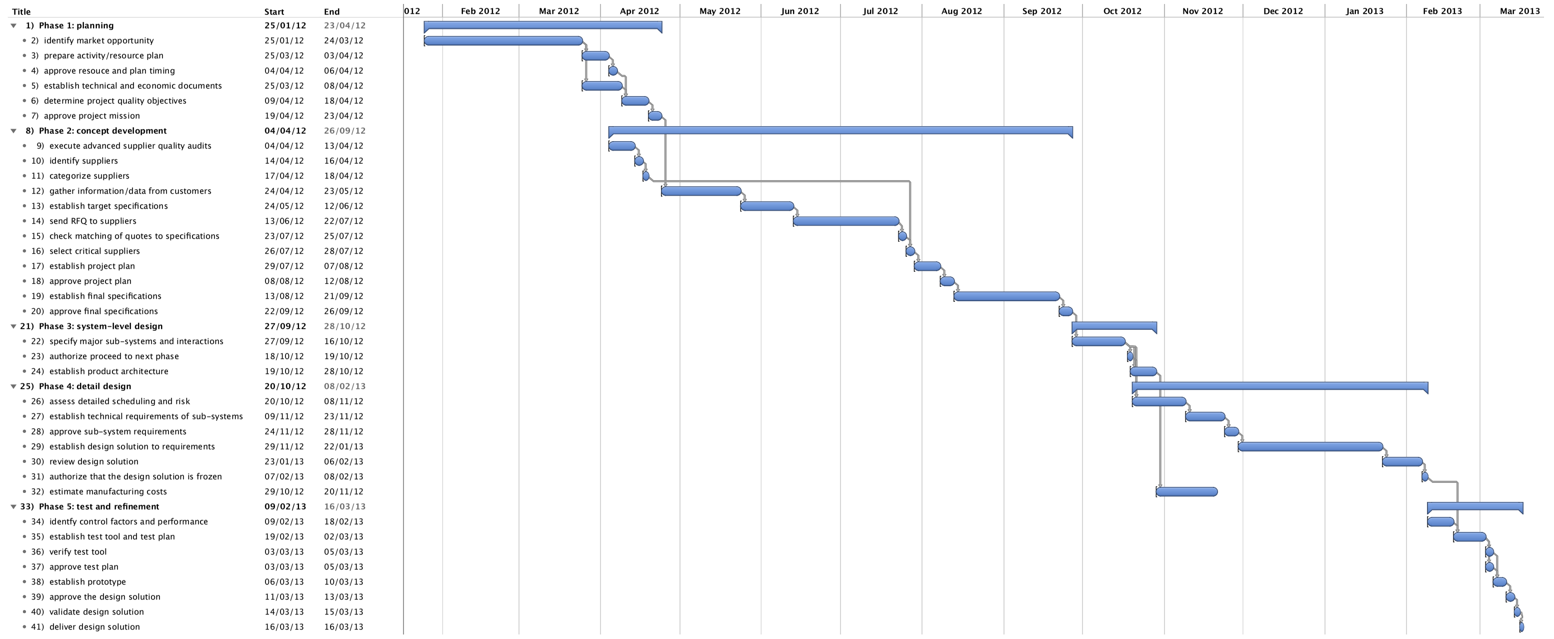
B.2.2 Task arrangement

See Figure B.3, Figure B.4 and Figure B.5.



Title	Effort	Start	End
▼ **1) Phase 1: planning**	20w 3d	**25/01/12**	23/04/12
• 2) identify market opportunity	12w	25/01/12	24/03/12
• 3) prepare activity/resource plan	2w	25/03/12	03/04/12
• 4) approve resouce and plan timing	3d	04/04/12	06/04/12
• 5) establish technical and economic documents	3w	25/03/12	08/04/12
• 6) determine project quality objectives	2w	09/04/12	18/04/12
• 7) approve project mission	1w	19/04/12	23/04/12
▼ **8) Phase 2: concept development**	34w 1d	**04/04/12**	26/09/12
• 9) execute advanced supplier quality audits	2w	04/04/12	13/04/12
• 10) identify suppliers	3d	14/04/12	16/04/12
• 11) categorize suppliers	2d	17/04/12	18/04/12
• 12) gather information/data from customers	6w	24/04/12	23/05/12
• 13) establish target specifications	4w	24/05/12	12/06/12
• 14) send RFQ to suppliers	8w	13/06/12	22/07/12
• 15) check matching of quotes to specifications	3d	23/07/12	25/07/12
• 16) select critical suppliers	3d	26/07/12	28/07/12
• 17) establish project plan	2w	29/07/12	07/08/12
• 18) approve project plan	1w	08/08/12	12/08/12
• 19) establish final specifications	8w	13/08/12	21/09/12
• 20) approve final specifications	1w	22/09/12	26/09/12
▼ **21) Phase 3: system-level design**	6w 2d	**27/09/12**	28/10/12
• 22) specify major sub-systems and interactions	4w	27/09/12	16/10/12
• 23) authorize proceed to next phase	2d	18/10/12	19/10/12
• 24) establish product architecture	2w	19/10/12	28/10/12
▼ **25) Phase 4: detail design**	27w	**20/10/12**	08/02/13
• 26) assess detailed scheduling and risk	4w	20/10/12	08/11/12
• 27) establish technical requirements of sub-systems	3w	09/11/12	23/11/12
• 28) approve sub-system requirements	1w	24/11/12	28/11/12
• 29) establish design solution to requirements	11w	29/11/12	22/01/13
• 30) review design solution	3w	23/01/13	06/02/13
• 31) authorize that the design solution is frozen	2d	07/02/13	08/02/13
• 32) estimate manufacturing costs	4w 3d	29/10/12	20/11/12
▼ **33) Phase 5: test and refinement**	7w 4d	**09/02/13**	16/03/13
• 34) identfy control factors and performance	2w	09/02/13	18/02/13
• 35) establish test tool and test plan	2w 2d	19/02/13	02/03/13
• 36) verify test tool	3d	03/03/13	05/03/13
• 37) approve test plan	3d	03/03/13	05/03/13
• 38) establish prototype	1w	06/03/13	10/03/13
• 39) approve the design solution	3d	11/03/13	13/03/13
• 40) validate design solution	2d	14/03/13	15/03/13
• 41) deliver design solution	1d	16/03/13	16/03/13

FIGURE B.3 – Task scheduling



Title	Start	End
1) Phase 1: planning	**25/01/12**	23/04/12
2) identify market opportunity	25/01/12	24/03/12
3) prepare activity/resource plan	25/03/12	03/04/12
4) approve resouce and plan timing	04/04/12	06/04/12
5) establish technical and economic documents	25/03/12	08/04/12
6) determine project quality objectives	09/04/12	18/04/12
7) approve project mission	19/04/12	23/04/12
8) Phase 2: concept development	**04/04/12**	26/09/12
9) execute advanced supplier quality audits	04/04/12	13/04/12
10) identify suppliers	14/04/12	16/04/12
11) categorize suppliers	17/04/12	18/04/12
12) gather information/data from customers	24/04/12	23/05/12
13) establish target specifications	24/05/12	12/06/12
14) send RFQ to suppliers	13/06/12	22/07/12
15) check matching of quotes to specifications	23/07/12	25/07/12
16) select critical suppliers	26/07/12	28/07/12
17) establish project plan	29/07/12	07/08/12
18) approve project plan	08/08/12	12/08/12
19) establish final specifications	13/08/12	21/09/12
20) approve final specifications	22/09/12	26/09/12
21) Phase 3: system-level design	**27/09/12**	28/10/12
22) specify major sub-systems and interactions	27/09/12	16/10/12
23) authorize proceed to next phase	18/10/12	19/10/12
24) establish product architecture	19/10/12	28/10/12
25) Phase 4: detail design	**20/10/12**	08/02/13
26) assess detailed scheduling and risk	20/10/12	08/11/12
27) establish technical requirements of sub-systems	09/11/12	23/11/12
28) approve sub-system requirements	24/11/12	28/11/12
29) establish design solution to requirements	29/11/12	22/01/13
30) review design solution	23/01/13	06/02/13
31) authorize that the design solution is frozen	07/02/13	08/02/13
32) estimate manufacturing costs	29/10/12	20/11/12
33) Phase 5: test and refinement	**09/02/13**	16/03/13
34) identfy control factors and performance	09/02/13	18/02/13
35) establish test tool and test plan	19/02/13	02/03/13
36) verify test tool	03/03/13	05/03/13
37) approve test plan	03/03/13	05/03/13
38) establish prototype	06/03/13	10/03/13
39) approve the design solution	11/03/13	13/03/13
40) validate design solution	14/03/13	15/03/13
41) deliver design solution	16/03/13	16/03/13

FIGURE B.4 – Gantt chart of the project

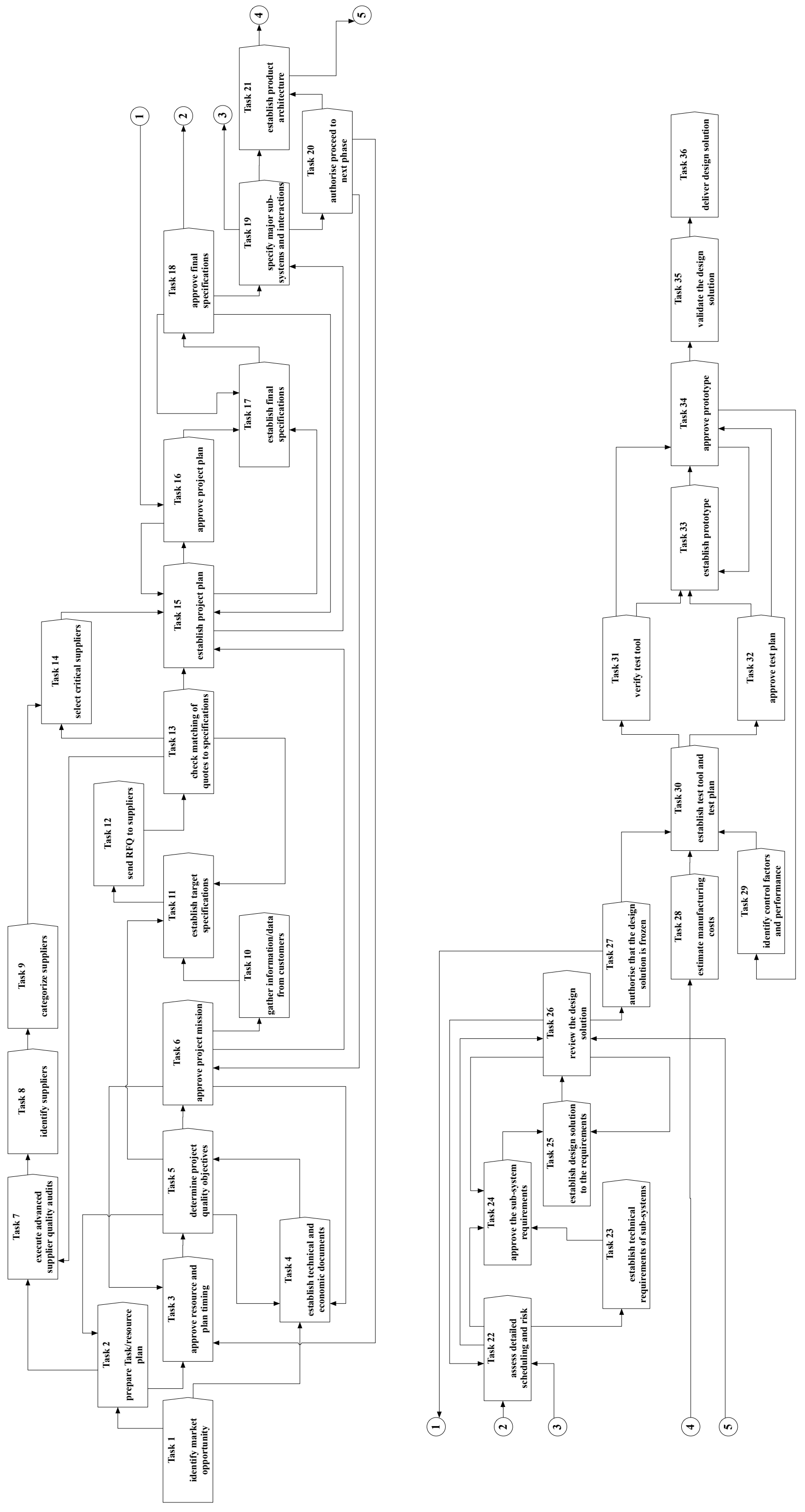

Task 1 identify market opportunity
Task 2 prepare Task/resource plan
Task 3 approve resource and plan timing
Task 4 establish technical and economic documents
Task 5 determine project quality objectives
Task 6 approve project mission
Task 7 execute advanced supplier quality audits
Task 8 identify suppliers
Task 9 categorize suppliers
Task 10 gather information/data from customers
Task 11 establish target specifications
Task 12 send RFQ to suppliers
Task 13 check matching of quotes to specifications
Task 14 select critical suppliers
Task 15 establish project plan
Task 16 approve project plan
Task 17 establish final specifications
Task 18 approve final specifications
Task 19 specify major sub-systems and interactions
Task 20 authorise proceed to next phase
Task 21 establish product architecture
Task 22 assess detailed scheduling and risk
Task 23 establish technical requirements of sub-systems
Task 24 approve the sub-system requirements
Task 25 establish design solution to the requirements
Task 26 review the design solution
Task 27 authorise that the design solution is frozen
Task 28 estimate manufacturing costs
Task 29 identify control factors and performance
Task 30 establish test tool and test plan
Task 31 verify test tool
Task 32 approve test plan
Task 33 establish prototype
Task 34 approve prototype
Task 35 validate the design solution
Task 36 deliver design solution
1
2
3
4
5

FIGURE B.5 – Task network

B.3 Complementary Data for Product Management Knowledge Area

B.3.1 Customer needs

The customer needs are organized hereafter (see Table B.1). In the table, 5 is the highest importance and 1 is the lowest. To make this table, we rely partly on the example developed by Ulrich & Eppinger (2007).

TABLE B.1 – Deliverable example in need state : customer needs

No.	Need	Importance
1	Reduces vibration to the hands	3
2	Allows easy traversal of slow, difficult terrain	2
3	Enables high-speed descents on bumpy trails	5
4	Allows sensitivity adjustment	3
5	Preserves the steering characteristics of the bike	4
6	Remains rigid during hard cornering	4
7	Is lightweight	4
8	Provides stiff mounting points for the brakes	2
9	Fits a wide variety of bikes, wheels, and tires	5
10	Is easy to install	1
11	Works with fenders	1
12	Instills pride	5
13	Is affordable for an amateur enthusiast	5
14	Is not contaminated by water	5
15	Is not contaminated by grunge	5
16	Can be easily accessed for maintenance	3
17	Allows easy replacement of worn parts	1
18	Can be maintained with readily available tools	3
19	Lasts a long time	5
20	Is safe in a crash	5

B.3.2 List of metrics for the suspension

Through generating "Needs Definition" deliverable, the customer needs are translated into some requirements in forms of "metrics" (see Table B.2).

TABLE B.2 – Deliverable example in need state : metrics

Metric No.	Need Nos.	Metric	Importance
1	1, 3	Attenuation from dropout to handlebar at 10 Hz	3
2	2, 6	Spring preload	3
3	1, 3	Maximum value from the Monster	5
4	1, 3	Minimum descent time on test rack	5
5	4	Damping coefficient adjustment range	3
6	5	Maximum travel (26-in. Wheel)	3
7	5	Rake offset	3
8	6	Lateral stiffness at the tip	3
9	7	Total mass	4
10	8	Lateral stiffness at brake pivots	2
11	9	Headset sizes	5
12	9	Steertube length	5
13	9	Wheel sizes	5
14	9	Maximum tire width	5
15	10	Time to assemble to frame	1
16	11	Fender compatibility	1
17	12	Instills pride	5
18	13	Unit manufacturing cost	5
19	14	Time in spray chamber without water entry	5
20	15	Cycles in mud chamber without contamination	5
21	16, 17	Time to disassemble/assemble for maintenance	3
22	17, 18	Special tools required for maintenance	3
23	19	UV test duration to degrade rubber parts	5
24	19	Monster cycles to failure	5
25	20	Japan Industrial Standards test	5
26	20	Bending strength (frontal loading)	5

B.3.3 Target specification

The preliminary version of product specifications that specifies the target specifications according to the customer needs (see Table B.3).

B.3.4 Final specification

By considering the constraints, executing benchmarking, etc., the final specifications are composed and released.

TABLE B.3 – Deliverable example in requirement state : target specification

Metric No.	Need Nos.	Metric	Import-ance	Unit	Marginal value	Ideal value
1	1, 3	Attenuation from dropout to handlebar at 10 Hz	3		>10	>15
2	2, 6	Spring preload	3		480-800	650-700
3	1, 3	Maximum value from the Monster	5		<3.5	<3.2
4	1, 3	Minimum descent time on test rack	5		<13.0	<11.0
5	4	Damping coefficient adjustment range	3		0	>200
6	5	Maximum travel (26-in. Wheel)	3		33-50	45
7	5	Rake offset	3		37-45	38
8	6	Lateral stiffness at the tip	3		>65	>130
9	7	Total mass	4		<1.4	<1.1
10	8	Lateral stiffness at brake pivots	2		>325	>650
11	9	Headset sizes	5		1.000 1.125	1.000 1.125 1.250
12	9	Steertube length	5		150 170 190 210	150 170 190 210 230
13	9	Wheel sizes	5		26in.	26in. 700C
14	9	Maximum tire width	5		>1.5	>1.75
15	10	Time to assemble to frame	1		<60	<35
16	11	Fender compatibility	1		None	All
17	12	Instills pride	5		>3	>5
18	13	Unit manufacturing cost	5		<85	<65
19	14	Time in spray chamber without water entry	5		>2300	>3600
20	15	Cycles in mud chamber without contamination	5		>15	>35
21	16, 17	Time to disassemble/assemble for maintenance	3		<300	<160
22	17, 18	Special tools required for maintenance	3		Hex	Hex
23	19	UV test duration to degrade rubber parts	5		>250	>450
24	19	Monster cycles to failure	5		>300k	>500k
25	20	Japan Industrial Standards test	5		Pass	Pass
26	20	Bending strength (frontal loading)	5		>7.0	>10.0

B.3.5 Product architecture

See Figure B.6, FigureB.7.

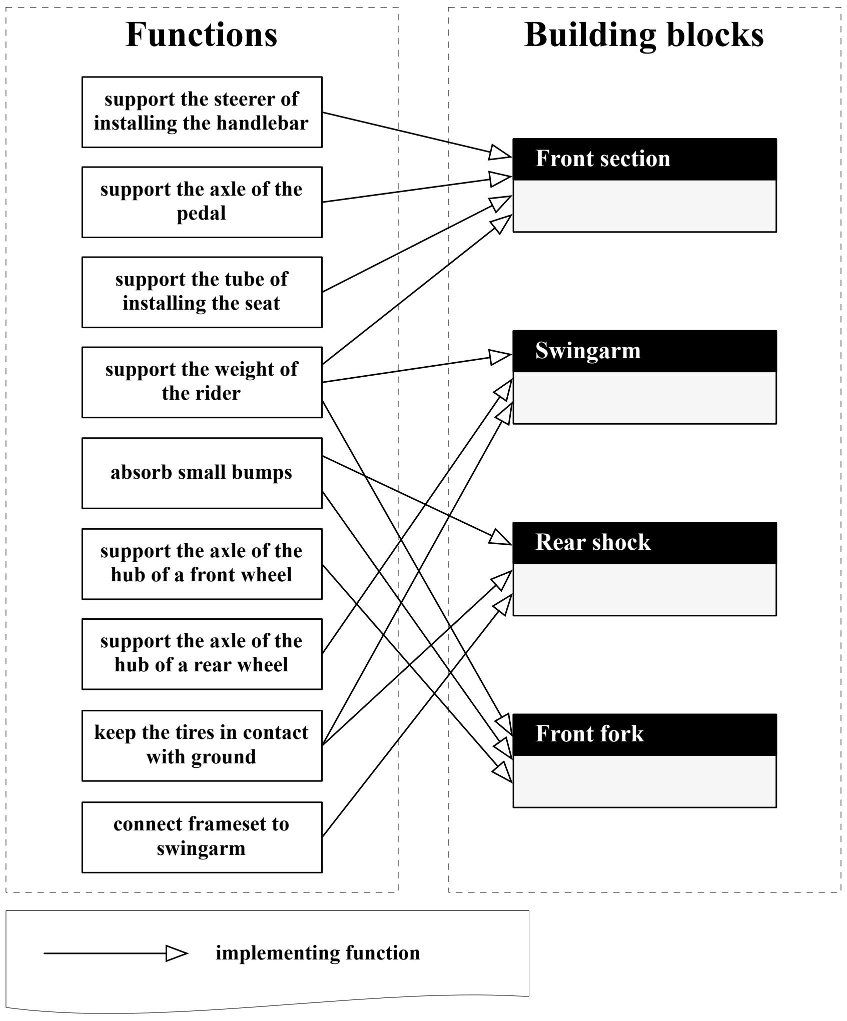

Functions
Building blocks
support the steerer of installing the handlebar
support the axle of the pedal
support the tube of installing the seat
support the weight of the rider
absorb small bumps
support the axle of the hub of a front wheel
support the axle of the hub of a rear wheel
keep the tires in contact with ground
connect frameset to swingarm
Front section
Swingarm
Rear shock
Front fork
implementing function

FIGURE B.6 – Product functions

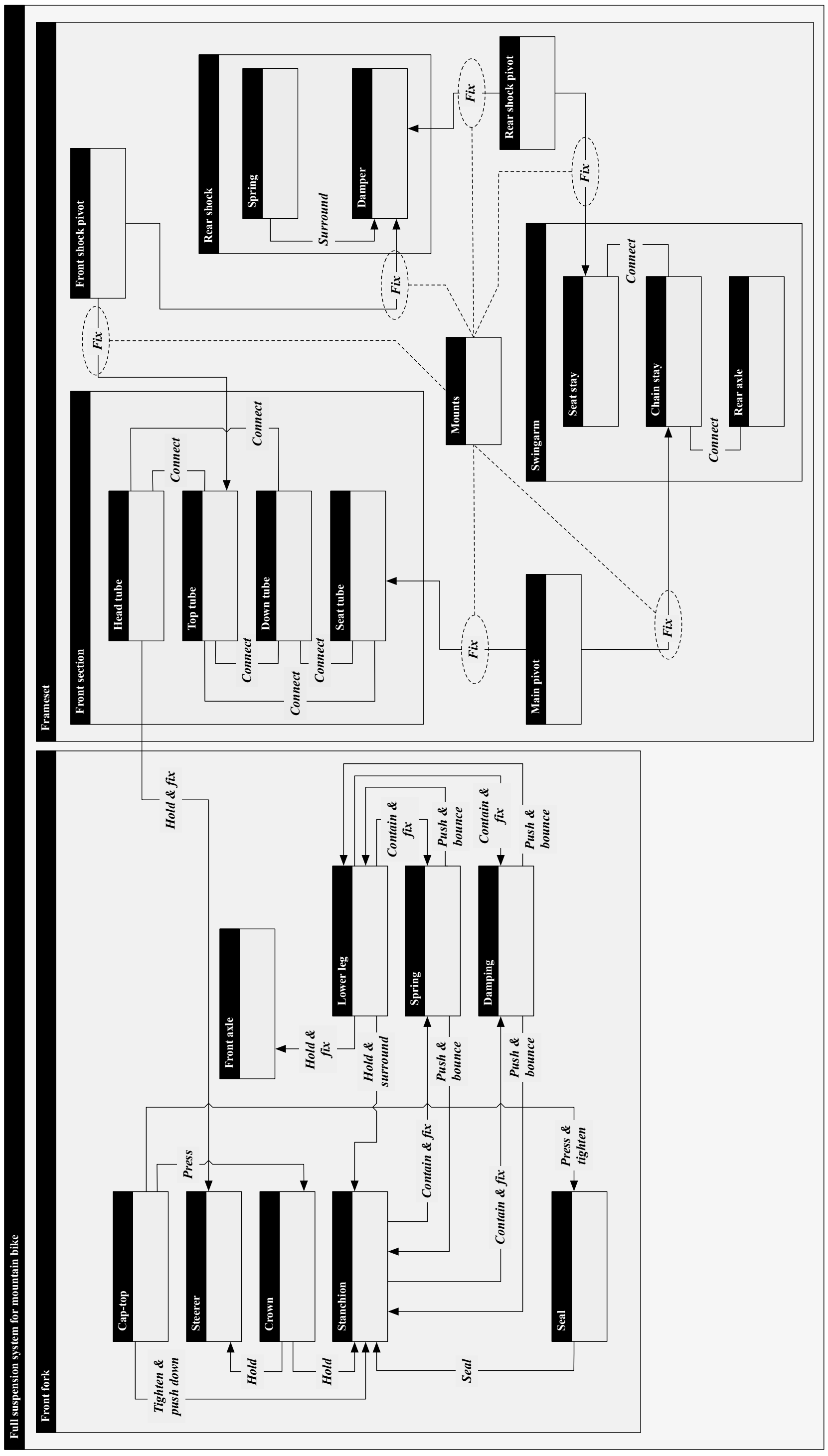


FIGURE B.7 – Interactions between building blocks

B.3.6 Bill-Of-Materials (BOM)

See Figure B.8.

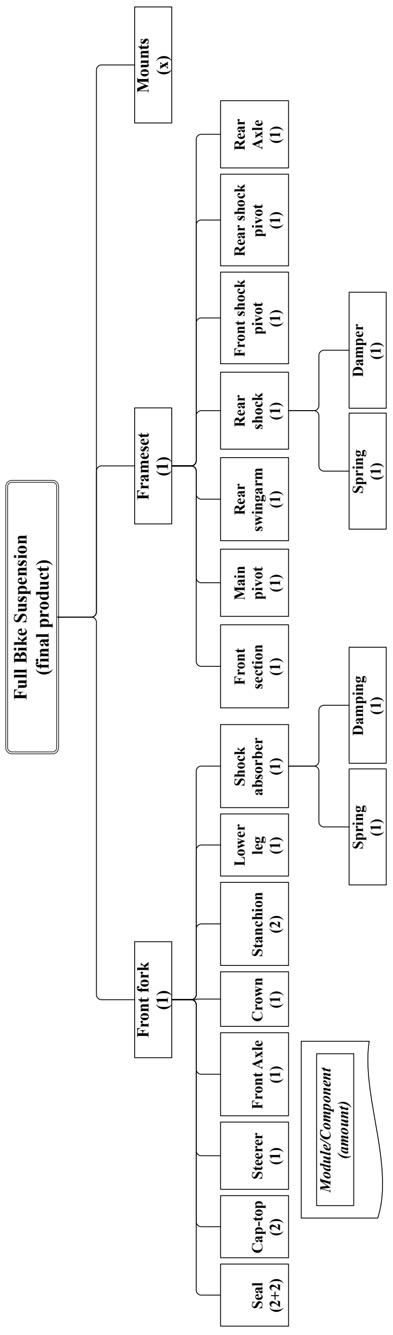


FIGURE B.8 – Bill-Of-Materials

B.3.7 Design solution

See Figure B.9, Figure B.10.

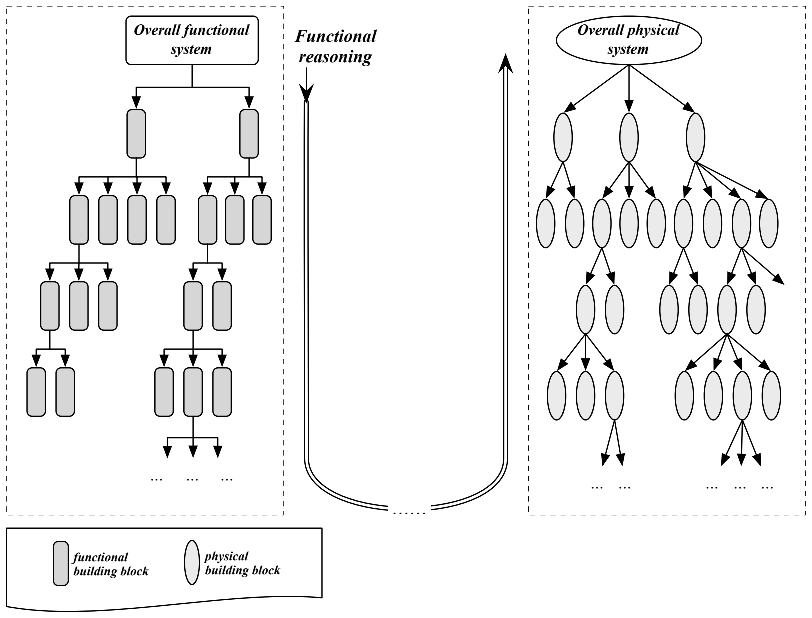


FIGURE B.9 – Design methodology (Functional reasoning methodology)

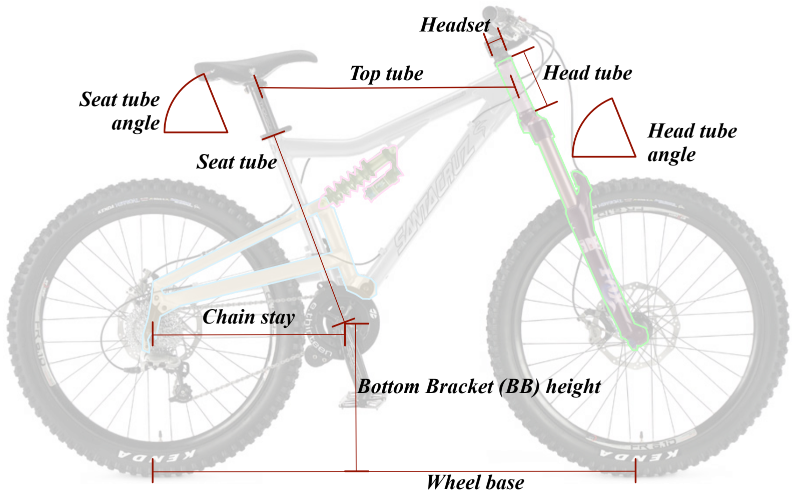


FIGURE B.10 – Frameset geometry

B.4 Supply chain of A.B.

See Figure B.11.

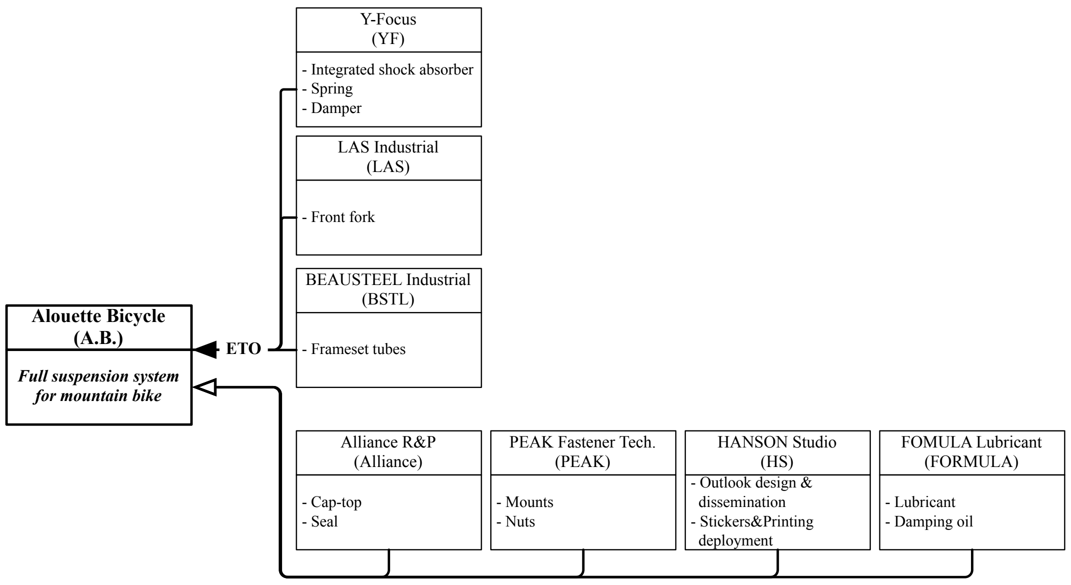


FIGURE B.11 – Supply chain of A.B.

B.5 Identified constructs

Table B.4 shows the example identified constructs during the state of "Logical solution".

Table B.5 adds two additional fields for the identified constructs during the state of "Physical solution".

TABLE B.4 – Identified constructs during the state of "Logical solution"

ConstructID	*BB_product*	*BB_project*	*BB_partner*	*DP_product*	*DP_project*	*DP_partner*	*Upstream ConstructID*
Cst_03_01	front spring	A22 : specify major sub-systems and interactions	YF	travel = 100mm ; free length = 162mm	start time = 27/09/12 end time = 16/10/12 duration = 4w	-	Cst_02_01
Cst_03_02	lower leg	A22 : specify major sub-systems and interactions	-	length = 370mm ; thickness = 1.2mm ; leg distance = 100mm	start time = 27/09/12 end time = 16/10/12 duration = 4w	-	Cst_02_03
Cst_03_03	steerer tube	A22 : specify major sub-systems and interactions	BSTL	length = 150,170,190,210,230 ; material = steel	start time = 27/09/12 end time = 16/10/12 duration = 4w	-	Cst_02_04
Cst_03_04	headset	A22 : specify major sub-systems and interactions	-	headset size = 1.000, 1.125, 1.250	start time = 27/09/12 end time = 16/10/12 duration = 4w	-	Cst_02_06
Cst_03_05	lower leg paint	-	-	color = black	-	-	Cst_02_07 Cst_02_03
Cst_03_06	crown paint	-	-	color = black	-	-	Cst_02_07
Cst_03_07	stanchion paint	-	-	-	-	-	Cst_02_07
Cst_03_08	crown	A22 : specify major sub-systems and interactions	-	material = alloy ; width = 120mm ; height = 35mm	start time = 27/09/12 end time = 16/10/12 duration = 4w	-	Cst_02_02
Cst_03_09	cap-top	A22 : specify major sub-systems and interactions	Alliance	color = black ; material = carbon fiber	start time = 27/09/12 end time = 16/10/12 duration = 4w	-	Cst_02_01
Cst_03_10	front axle	A22 : specify major sub-systems and interactions	-	length = 120mm ; material = hi-alloy	start time = 27/09/12 end time = 16/10/12 duration = 4w	-	Cst_02_08
Cst_03_11	stanchion	A22 : specify major sub-systems and interactions	-	material = hi-tensile steel ; stanchion surface material = bronze	start time = 27/09/12 end time = 16/10/12 duration = 4w	-	Cst_02_01

TABLE B.5 – Identified constructs during the state of "Physical solution"

ConstructID	**DP_product(mm)**	**tolerance(mm)**	**Abs. tolerance(mm)**
cst_04_01	travel = 100	[-3 , +3]	[-3.5 , +3.5]
	material = titanium alloy		
	external diameter = 25	[-1 , +2]	[-1.5 , +2.5]
	free length = 162	[-5 , +5]	[-6 , +7]
cst_04_02	length = 370	[-5 , +7]	[-7 , +9]
	internal diameter = 45	[-1.5 , +3]	[-1.8 , +3.5]
	thickness = 1.2	[-0.1 , +0.3]	[-0.2 , +0.5]
	color = black		
	leg distance = 100	[-2 , +2]	[-4 , +4]
cst_04_03	length = 150, 170, 190, 210, 230	[-2 , +2]	[-5 , +5]
	external diameter = 28.58	[-0.3 , +0.3]	[-0.5 , +0.5]
	material = steel		
	tube thickness = 2	[-0.1 , +0.3]	[-0.2 , +0.5]
cst_04_04	material = alloy		
	width = 120	[-5 , +5]	[-6 , +6]
	height = 35	[-1 , +2.5]	[-3 , +4]
	internal diameter = 30	[-0.3 , +0.3]	[-0.4 , +0.4]
	internal diameter = 28.58	[-0.3 , +0.3]	[-0.4 , +0.4]
cst_04_05	bore diameter = 3.2	[-0.1 , 0.1]	[-0.15 , 0.15]
	threaded race diameter = 26.8	[-0.3 , +0.3]	[-0.35 , +0.35]
	outside diameter = 32	[-1 , +3]	[-1.5 , +4]
	color = black		
	material = carbon fiber		
cst_04_06	diameter = 15	[-0.5 , +0.5]	[-0.6 , +0.6]
	length = 120	[-0.5 , +5]	[-0.6 , +7]
	material = hi-alloy		
cst_04_07	material = hi-tensile steel		
	stanchion surface = bronze		
	external diameter = 30	[-0.3 , +0.3]	[-0.5 , +0.5]
	thickness = 1.6	[-0.3 , +0.3]	[-0.4 , +0.4]
	length = 200	[-5 , +7]	[-8 , +9]
	internal diameter = 26.8	[-1 , +1]	[-1.5 , +1.5]

C
Key Concepts

Alteration

The statement of the phenomenon that something is made different.

Attribute

A quality or feature regarded as a characteristic or inherent part of system.

Building block

The intermediate system obtained during decomposing a PD project, product or partner.

Construct

An artefact perceived and determined by its owned single particular functionality. The functionality is specified by some direct parameters characterizing the building blocks from the product management, project management and/or partnership management knowledge areas.

Contribution

The relationship between two adjacent deliverables reflects the plan/solution in the downstream deliverable contributes in achieving the corresponding goal created in the upstream deliverable through supplying effort.

Coupling

The act of joining two system together.

Deliverable

A set of generated documents that indicates the obtained results from aggregating the efforts, the specified objectives of designing and developing the product, and the means used to determine whether the objectives are achieved.

Dependency

The effect of the change in one system's KID on another.

Direct parameter

A measure item determining the response, the characteristics and/or the behaviour of a system obtained through decomposing the knowledge areas.

Generation

The relationship between two adjacent deliverables reflects the corresponding plan(s)/solution(s) in the downstream deliverable is/are created or produced by the upstream deliverable.

Knowledge area

A scope of specialization.

PD project

An endeavor comprised of the myriad, multi-functional activities done between defining a technology or market opportunity and starting production.

Tolerance

An allowable amount of variation of the corresponding value belong to the direct parameter.

D
Data Entities

▷ *The data entities created and used in the Agent-based technical solution are explained in this annexe.* ◁

TABLE D.1 – Specifications of construct entity

Entity	*Property*	*Description*	*External relation*
Construct	ID	The identification of entity : construct	-
	Class	The type of entity : construct	-
	Building block ID (Product)	The identification of the composite building block from product management knowledge area	Building block entities
	Building block ID (Project)	The identification of the composite building block from project management knowledge area	Building block entities
	Building block ID (Partnership)	The identification of the composite building block from partnership management knowledge area	Building block entities
	Change occurrence	The record of the occurrences of changes within the construct	Change entities

TABLE D.2 – Specifications of building block entity

Entity	*Property*	*Description*	*External relation*
Building block	ID	The identification of entity	-
	Class	The type of entity : building block	-
	Involved knowledge area	The involved knowledge area : product, project, partnership knowledge areas	-
	Building block description	The description of the building block	-
	Direct parameter list ID	The list identification of the involved direct parameters characterizing the building block	Direct parameter entities
	Alteration list ID	The list identification of the led alterations in the building block	Alteration entities
	Change list ID	The list identification of the occurred changes in the building block	Change entities

TABLE D.3 – Specifications of direct parameter entity

Entity	*Property*	*Description*	*External relation*
Direct parameter	ID	The identification of entity : direct parameter	-
	Class	The type of entity : direct parameter	-
	Involved knowledge area	The involved knowledge area : product, project, partnership knowledge areas	-
	Name	The direct parameter name	-
	Attribute	The property name of the building block characterized by the current direct parameter	-
	Value	The quantitative value of the direct parameter	-
	Tolerance	The boundary of the direct parameter	-
	Absolute tolerance	The extreme boundary of the direct parameter and it is optional	-
	Description	The description of the direct parameter	-
	Dependency list ID	The list identification of the dependencies concerning the current direct parameter	Dependency entities

TABLE D.4 – Specifications of dependency entity

Entity	*Property*	*Description*	*External relation*
Dependency	ID	The identification of entity : dependency	-
	Class	The type of entity : dependency	-
	Source	The upstream direct parameter ID	Direct parameter entity
	Target	The downstream direct parameter ID	Direct parameter entity
	Dependency type	The potential type of dependency : generation, contribution, parameter coupling	-
	Description	The description of the dependency	-
	Effect	The formalized specifications of the effect transferred by the dependency	-

TABLE D.5 – Specifications of alteration entity

Entity	*Property*	*Description*	*External relation*
Alteration	ID	The identification of entity : alteration	-
	Class	The type of entity : alteration	-
	Involved direct parameter ID	The identification of the direct parameter where the alteration is led in	Direct parameter entity
	Description	The description of the alteration	-
	Shifting effect	The formalized specifications of the shifting effect led in by the alteration	-
	Format	The data format of the involved direct parameter	-

TABLE D.6 – Specifications of alteration list entity

Entity	*Property*	*Description*	*External relation*
Alteration list	ID	The identification of entity : alteration list	-
	Class	The type of entity : alteration list	-
	Included alteration IDs	The set of the alterations corresponding to the related building block	Alteration entity

TABLE D.7 – Specifications of change entity

Entity	*Property*	*Description*	*External relation*
Change	ID	The identification of entity : change	-
	Class	The type of entity : change	-
	Involved direct parameter ID	The identification of the direct parameter where the change occurs	Direct parameter entity
	Description	The description of the change	-
	Due alteration ID	The identification of the alteration causing the change	Alteration entities

TABLE D.8 – Specifications of change propagation entity

Entity	*Property*	*Description*	*External relation*
Change propagation	ID	The identification of entity : change propagation	-
	Class	The type of entity : change propagation	-
	Involved direct parameter ID (upstream)	The identification of the upstream direct parameter where the initial change occurs and is propagating	Direct parameter entity
	Involved direct parameter ID (downstream)	The identification of the downstream direct parameter where the propagated change is caused by the propagation	Direct parameter entity
	Due change ID	The identification of the initial change occurring in the upstream direct parameter	Change entity
	Involved dependency ID	The identification of the dependency transferring the effect of the intitial change	Dependency entity
	Description	The description of the change propagation	-

TABLE D.9 – Specifications of change list entity

Entity	*Property*	*Description*	*External relation*
Change list	ID	The identification of entity : change list	-
	Class	The type of entity : change list	-
	Included change IDs	The set of the changes corresponding to the related building block	Change entity

TABLE D.10 – Specifications of direct parameter list entity

Entity	*Property*	*Description*	*External relation*
Direct parameter list	ID	The identification of entity : direct parameter list	-
	Class	The type of entity : direct parameter list	-
	Included direct parameter IDs	The set of the direct parameters corresponding to the related building block	Direct parameter entity

TABLE D.11 – Specifications of dependency list entity

Entity	*Property*	*Description*	*External relation*
Dependency list	ID	The identification of entity : dependency list	-
	Class	The type of entity : dependency list	-
	Included dependency IDs	The set of the dependencies corresponding to the related direct parameters	Dependency entity

E
Extended Abstract

Contribution to Engineering Change Management in Product Development Projects

With the business context growing, companies are facing more and more challenges from product management and supply chain management simultaneously during Product Development (PD). On one hand, products are expected to be designed and developed in a shorter period to satisfy the requirements as far as possible, whereas on the other hand the concerned partners should establish an effective and efficient communication to achieve their own objectives and take the profit through the product development project. When considering the above aspects, one of important issues is change management. During a PD project, changes reveal multiple senses. Changes could bring the opportunity to a company for innovations. However, changes could also enhance the risk of failing to release the final product within the constraints (lead time, requirements, cost, etc.). Therefore, the change management during the PD project generally aims at supporting companies to cope with alterations through the PD project in order to achieve the objectives. Moreover, one change can cause the occurrence of other changes, and it is to say that the change is propagated through the potential relations between the parts, roles, activities and any other element involved in the product development project.

In this Ph.D. research, we aim at providing the contributions with the methodologies and tools to improve the performance in engineering change management. The

research achievement is disseminated by a multi-agent based technical solution of simulating change propagations in Product Development (PD) projects.

In a PD project, change can occur at any time and propagate through the potential relations between the multiple knowledge areas of the product development project. In other words, a change occurring in one element belonging to one knowledge area could cause other change(s) in other area(s). However, the research on engineering change considering multiple knowledge areas are limited. To file this gap, we firstly establish a general framework presenting our research point of view in engineering change management, i.e., Co-evolution Oriented Change Analysis (COCA) framework.

The COCA framework enables us to :

– Model product design and development process by indicating how the product evolves considering those three knowledge areas simultaneously ;
– Acquire the knowledge/information/data derived from the multiple knowledge areas. The knowledge/information/data can be aggregated to reflect the product functions ;
– Identify the dependencies between the knowledge/information/data belonging to the same/different knowledge area(s) ;
– Identify the potential change propagation channels, and analyze the mechanism of change propagations.

It provides a simultaneous modeling consideration in three knowledge areas as well as their interrelations :

1. Project management : this knowledge area manages product design and development activities with a set of milestones and makes sure they are under the time, quality and cost constraints ;
2. Product management : this knowledge area covers the process during which product model evolves from customer needs to design solution, and then the product solution is released to achieve the expected performance and expected needs ;
3. Partnership management : this knowledge area considers the activities and roles of the partners participating in the PD project along the evolution of the product.

Corresponding to the project management knowledge area, we employ the generic product design and development process proposed by Ulrich et al. (2007) to model the process of project. The generic product design and development process stresses an endeavor from the viewpoint of the project evolution, during which the six phases imply a serial of critical activities that are executed to achieve the prescribed milestones. We propose a hierarchical model of analyzing project and obtain the three groups of building blocks as : project, phase and task building blocks.

Corresponding to the product management knowledge area, we propose a four-state product evolution model that reflects the procedures of efforts to generate the product design solution. In the product evolution model, the four states refer to Needs, Requirements, Logical solution and Physical solution. These states capture a set of milestones in the evolution of the product model. There would exist lots of back and forth between each couple of the adjacent states, which reflects the possible iterations during the PD project and implies the mutual relations between the states. The product evolution process enables to model the product data with investigating the relations between the states and the relations between the building blocks belonging to the same state. Four groups of product building blocks are obtained as : need, requirement, functional and physical building blocks.

Corresponding to the partnership management knowledge area, we divide all the partners into the classes according to their contributions. We then identify the partner individuals (the companies or business units) in each class according to their own specific business scopes. During the product evolution process, the partners participate in the project with performing their responsibilities. Three groups of partner building blocks are obtained as : partner class, partner individual and partner responsibility building blocks.

When analyzing the three knowledge areas simultaneously, we take use of product evolution process (product management knowledge area) as the main clue to connect the other two knowledge areas. Then we are enabled to model the PD project progress and the product evolution process with considering the participation of the partners at the same time. In this way, we structure the PD project through aggregating the knowledge/information/data as the structured representations which are denoted as "constructs".

As follows, to analyze change and change propagation in the PD project, we continue to present the conceptual models of change occurrence and propagation. The conceptual model of change occurrence enables us to justify whether an alteration could result in change, and it reveals a set of characteristics of changes. While, with the conceptual model of change propagation, the channels through which the impact of occurred change is transferred are identified.

With the modeled constructs and their relationships, a change propagation network is established, in which constructs are designated as the nodes and their relationships are the edges. By adopting the conceptual models of change occurrence and change propagation, an alteration introduced into one construct can be identified and analyzed to determine whether a change is occurring or not. If a change occurs, its influence can be transferred to other node(s) through the edge(s). A construct is an artefact obtained through aggregating the building blocks from the three knowledge areas respectively. Moreover, each building block is characterized by a set of direct parameters. Relying on the hierarchical structure of constructs, we are enabled to treat the change propagations among the constructs as the multi-layer communication. According to the construct structure, we build up three communication layers through encapsulating their internal behaviours respectively. The three communication layers are :

1. *Construct layer* : this is the scope within which the constructs receive alterations, forward the alterations to other procedures and determine the response according

to the preset plans. The introduced alterations into one construct would be treated in local to be qualified as causing the consequences (i.e., regular alteration, change occurrence, dysfunctional change occurrence), and they are quantified to invoke the change propagations to other constructs.

2. *Building block layer* : this layer is the scope within which the building blocks search the involved direct parameters in term of the query from their located construct(s) and look up the relations (cf. Table 3.7 in section 3.6) for supporting to execute the potential change propagations.
3. *Parameter layer* : this layer is the scope where the alterations introduced into the involved direct parameters are treated according to the conceptual model of change occurrence and propagation. In this layer, the relations between building blocks are embodied as the dependencies between the direct parameters.

One realistic problem in engineering change management is how to handle the changes efficiently and effectively. This constitutes the main problem which our work is answering. To answer this problem, we provide the methodology and tool of simulating change propagation. We firstly build up a multi-layer structured change propagation network, which is composed with the constructs as nodes and their relations as edges. At the same time, we propose the rationale of simulating change propagations based on the identified change propagation mechanism. Each node is equipped with the procedures of identifying, estimating and coping with changes according to conceptual models of change occurrence and propagation. The simulation solution starts with introducing the alterations. Through analyzing the consequence due to the alterations, whether a change occurs or not is judged. If a change is perceived, the coping actions will be taken and the change would be propagated along the edges. Meanwhile, the solutions to treat simultaneous changes encountered during the occurrence are also executed.

Lastly, we select the multi-agent-based technique to implement the simulation methodology in view of the structure of the change propagation network and the constructs equipped with the distributed procedures. The implemented tool clearly illustrates the product evolution process, the change propagation network, the change propagation channels when change occurs, the infected constructs along the propagation channels, and their propagation parameters.

Keyword : Product Development project, product evolution process, engineering change, change propagation, multi-agent system, Co-evolution Oriented Change Analysis framework, change propagation simulation, product-project-partnership.

F
Resume etendu en francais

Contribution à l'ingénierie du changement dans les projets de développement de produits : modèle de référence et simulation par système multi-agents

Avec le contexte d'affaires en croissance, les entreprises sont de plus en plus confrontées à des défis la gestion des produits et la gestion de la chaîne d'approvisionnement simultanément pendant le développement du produit (PD). D'une part, les produits devraient être conçus dans un délai plus court pour satisfaire aux exigences autant que possible, tandis que d'autre part les partenaires concernés devraient établir une communication efficace et efficiente pour atteindre leurs propres objectifs et prendre les bénéfices à travers le produit projet de développement. Lorsque l'on considère les aspects ci-dessus, l'une des questions importantes est la gestion du changement.

Durant la durée du PD projet, les changements révèlent plusieurs sens. Des changements pourraient apporter la possibilité à une entreprise pour des innovations. Toutefois, les changements pourraient aussi augmenter le risque de ne pas libérer le produit final dans les contraintes (de délai, les exigences, les coûts, etc.) Par conséquent, la gestion du changement au cours du PD projet vise généralement à soutenir les entreprises à faire face aux changements à travers le PD projet afin d'atteindre les objectifs. En outre, un changement peut entraîner l'apparition d'autres changements. Cela signifie que la modification est propagée à travers les relations possibles entre les parties, les rôles, les activités et tout autre élément impliqué dans le PD projet.

Dans mes travaux de thèse, je mintéresse aux méthodes et outils visant à améliorer la gestion du changement en ingénierie. Je propose ainsi une solution technique basée sur une modélisation multi-agents permettant de simuler la propagation du changement dans les PD projets.

Dans un projet de PD, le changement peut survenir à tout moment et se propager à travers les relations possibles entre les domaines de connaissances multiples du PD projet. En d'autres termes, un changement survenu dans un élément appartenant à un domaine de la connaissance pourrait causer d'autres changements (s) dans une autre domaine(s). Cependant, la recherche sur les changements d'ingénierie compte tenu des domaines de connaissances multiples sont limitées. Pour déposer cette lacune, nous établissons d'abord un cadre général de présenter notre point de vue de la recherche dans la gestion du changement en ingénierie , nommé COCA (Co-evolution Oriented Change Analysis).

COCA associe les domaines produit, projet et partenaires et nous permet de :

– Modéliser processus de conception et développement de produits en indiquant comment le produit évolue compte tenu de ces trois domaines de connaissances simultanément ;
– Acquérir les connaissances/informations/données provenant des domaines de connaissances multiples. Les connaissances/informations/données peuvent être agrégées pour refléter les fonctions du produit ;
– Identifier les dépendances entre les connaissances/informations/données appartenant au(x) même/différente domaine(s) même/différente de la connaissance ;
– Identifier les canaux potentiels de la propagation du changement, etă analyser le mécanisme de la propagation du changement.

COCA fournit une méthode de modélisation en compte les trois domaines de connaissances ainsi que leurs interrelations :

– Gestion de projet : ce domaine de connaissances gère les activités de conception et de développement de produits avec un ensemble de jalons et s'assure qu'ils sont sous des contraintes de temps, de qualité et de coût ;
– Gestion de produit : ce domaine de connaissance couvre le processus au cours duquel le modèle de produit évolue des besoins du client aux solutions de conception, puis la solution de produit est libéré pour atteindre les performances attendues et les besoins prévus ;
– Gestion de partenariat : ce domaine de la connaissance considère les activités et les rôles des partenaires participant au PD projet sur l'évolution du produit.

Correspondant à le domaine de la connaissance de la gestion de projet, nous employons le processus générique de la conception et le développement du produit, proposé par Ulrich et al. (2007) pour modéliser le processus de projet. Ce processus souligne un effort du point de vue de l'évolution du projet, au cours de laquelle les six phases impliquent une série d'activités critiques qui sont exécutées pour atteindre les jalons prévus. Nous proposons un modèle hiérarchique de l'analyse de projet et nous obtenons les trois groupes de « building blocks » comme : « project building block »,

« phase building block » et « task building block ».

Correspondant à le domaine de la connaissance de la gestion de produit, nous proposons un modèle d'évolution de produit qui reflète les procédures d'efforts pour générer la solution de conception de produits. Le modèle d'évolution du produit a quatre états qui se réfèrent à « Needs », « Requirements », « Logical solution » and « Physical solution ». Ces états de capturer un ensemble de jalons dans l'évolution du produit. Il existerait beaucoup de va-et- vient entre chaque couple des états adjacents, ce qui reflète les itérations possibles au cours du PD projet et implique les relations mutuelles entre les etats. Le processus de l'évolution du produit permet de modéliser les données du produit et d'analyser les relations entre les etats et les relations entre les building blocks appartenant à le même état. Quatre groupes de « building blocks » de produit sont obtenus sous la forme : « need building block », « requirement building block », « functional building block » et « physical building block ».

Correspondant à le domaine de la connaissance de la gestion des partenariats, nous divisons tous les partenaires dans les classes en fonction de leurs contributions. Nous identifions alors les individus de partenaires (les entreprises ou unités d'affaires) dans chaque catégorie en fonction de leurs propres champs d'activité spécifiques. Pendant le processus de l'évolution des produits, les partenaires participent au projet avec l'exercice de leurs responsabilités. Trois groupes de « building blocks » sont obtenus comme : « partner class building block », « partner individual building block » et « partner responsibility building block ».

Lors de l'analyse des trois domaines de connaissances simultanément, nous utilisons des processus d'évolution du produit (domaine de connaissance de la gestion des produits) que l'indice principal de relier les deux autres domaines de connaissances. Ensuite, nous sommes capables de modéliser l'évolution du PD projet et le processus d'évolution de produit compte tenu de la participation des partenaires au même moment. Cette méthode sappuie sur des éléments appelés « constructs » pour procéder à lanalyse ; ces éléments associent de la connaissance issue des trois domaines cités cidessus. Ils constituent les éléments de base dun réseau sur lequel sappuiera lanalyse de loccurrence et de la propagation du changement.

Comme suit, pour lidentification des changements et leur propagation dans le PD projet, nous établissons les modèles conceptuels. Le modèle conceptuel des changements nous permet de justifier de savoir si une « alteration » peut entraîner des changements, et il révèle un ensemble de caractéristiques des changements. La notion d« altération » est une dénomination originale de lauteur pour désigner une modification sur le projet à lorigine dun changement. Avec le modèle conceptuel de la propagation des changements, les canaux par lesquels l'impact du changement survenu est transféré sont identifiés. Avec les constructions modélisés et leurs relations, nous établissons un réseau de propagation du changement, dans lequel les nuds sont les « constructs », via lesquels le changement se propage, et les liens les relations entre « constructs ». En utilisant les modèles conceptuels de la apparition et la propagation des changements, une

« altération » peut rester dans le domaine de tolérance de lattribut ou non. La modélisation de la propagation des modifications est alors présentée. Les scénarios conduisant à une « altération » sont alors présentés. Si un changement intervient, son influence peut être transférée à un autre noeud (s) par le bord (s). Une « construct » est un artefact obtenu par l'agrégation des « building blocks » des trois domaines de connaissances respectivement. De plus, chaque « building block » est caractérisé par un ensemble de paramètres directs. S'appuyant sur la structure hiérarchique de constructions, nous sommes capables de traiter les propagations de changement parmi les constructions que la communication multi-couche. Selon la structure de « construct », nous construisons trois couches de communication par encapsulation leurs comportements internes respectivement. Les trois couches de communication sont les suivants :

– Construct layer : c'est le cadre dans lequel les « constructs » reçoivent « altération », transmettre les « altération » à d'autres procédures et déterminer la réponse selon les plans préétablis. Les « altération » introduites dans une « construct » seraient traités en local pour être qualifié comme causant les conséquences (« altération » régulière, changement occurrence, dysfonctionnel changement occurrence), et ils sont quantifiées à invoquer les propagations des changements à d'autres « constructs » ;
– Building block layer : cette couche est le cadre dans lequel la recherche les « building blocks » les paramètres directs concernés en terme de la requête de leur « construct » situé et se tournent vers les relations de soutien pour exécuter les propagations des changements potentiels ;
– Parameter layer : cette couche est du domaine dans lequel les « altération » introduites dans les paramètres directs concernés sont traités selon le modèle conceptuel de changement apparition et la propagation. Dans cette couche, les relations entre les « building blocks » sont réalisés en tant que les dépendances entre les paramètres directs.

Un problème réaliste dans la gestion du changement de l'ingénierie est de savoir comment gérer les changements efficace et efficiente. Pour répondre à ce problème, nous proposons la méthodologie et un outil de simulation de la propagation des changements. Nous construisons d'abord un réseau structuré changement propagation multi-couche, qui est composé avec les « construct » comme des nuds et leurs relations que les bords. Dans le même temps, nous proposons la logique de simuler les changements propagations sur la base du mécanisme de changement de propagation identifié. Chaque nud est équipé avec les procédures d'identification, l'estimation et l'adaptation aux changements selon les modèles conceptuels de changement apparition et la propagation. La solution de simulation commence par l'introduction de ces « altération ». En analysant la conséquence due aux « altération », si un changement intervient ou pas est jugé. Si un changement est perçu, les actions d'adaptation seront prises et le changement seraient propagées le long des bords. Pendant ce temps, les solutions pour traiter les changements simultanés rencontrés lors de l'événement sont également exécutées.

Enfin, on sélectionne la technique à base de multi-agent-de mettre en uvre la méthodologie de simulation en vue de la structure du réseau de propagation du changement et les « construct » équipées des procédures distribués. L'outil mis en place illustre clairement le processus de produit de l'évolution, le réseau de changement de propagation, les canaux de propagation du changement quand le changement intervient, les « construct » infectés le long des canaux de propagation, et leurs paramètres de propagation.

Mots clés : les projets de developpement de produits, processus de l'évolution des produits, l'ingenierie du changement, la propagation des modifications, systeme multi-agents, Co-evolution Oriented Change Analysis framework, la simulation de propagation de changement, produit-projet-réseau de partenaires.

References

ALBERS, A., MATTHIESEN, S. & OHMER, M. (2003). An innovative new basic model in design methodology for analysis and synthesis of technical systems. In *INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'03*. 23

ALEXANDER, I. & BEUS-DUKIC, L. (2009). *Discovering Requirements*. How to Sepcify Products and Services, WILEY. 44

ANDREASEN, M. & HEIN, L. (1987). *Integrated product development*. IFS (Publications). 38

ANDREW, K. & JUITE, W. (1995). Dependency Analysis in Constraint Negotiation. *IEEE TRANSATIONS ON SYSTEM, MAN, AND CYBERETICS*, **25**, 1301–1313. 57, 59

BAUER, B., MÜLLER, J.P. & ODELL, J. (2001). Agent uml : A formalism for specifying multiagent software systems. *International Journal of Software Engineering and Knowledge Engineering*, **11**, 1–24. 28

BHATTACHARYA, S., KRISHNAN, V. & MAHAJAN, V. (1998). Managing new product definition in highly dynamic environments. *Management Science*, **44**, 50–64. 3

BOEHM, B.W. & DEMARCO, T. (1997). Software risk management. *IEEE software*, **14**, 17–19. 127

BONABEAU, E. (2002). Agent-based modeling : Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, **99**, 7280–7287. 28

BRALLA, J.G. (1999). *Design for Manufacturability Handbook*. The McGraw-Hill Companies, 2nd edn. 15

BROWNING, T.R. (2001). Applying the design structure matrix to system decomposition and integration problems : A review and new directions. *IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT*, **48**, 292–306. 57, 59

BROWNING, T.R., FRICKE, E. & NEGELE, H. (2006). Key concepts in modeling product development processes. *Systems Engineering*, **9**, 104–128. 33, 59

CARLEY, K.M. & KRACKHARDT, D. (1999). A Typology for C2 Measures . In *The 1999 International Symposium on Command and Control Research and Technology*, 1–12. 59

CHAKRABARTI, A. & BLIGH, T.P. (2001). A scheme for functional reasoning in conceptual design. *Design Studies*, **22**, 493 – 517. 46

CHU, C.H. & TRETHEWEY, M.W. (1998). RAPID STRUCTURAL DESIGN CHANGE EVALUATION WITH AN EXPERIMENT BASED FEM. *Journal of Sound and Vibration*, **211**, 335–353. 22

CLARK, K.B. & FUJIMOTO, T. (1991). *Product Development Performance : Strategy, Organization and Management in the World Auto Industry*. Harvard Business School Press, Boston, Massachusetts, USA. 3, 15, 26

CLARKSON, J., SIMONS, C. & ECKERT, C. (2001). Predicting Change Propagation in Complex Design. In *ASME 2001 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 1–10. 19, 23, 24

CLARKSON, P., SIMONS, C. & ECKERT, C. (2004). Predicting change propagation in complex design. *ASME Journal of Mechanical Design*, **126**, 765–797. 23

CLAUSING, D. (1994). *Total quality development : a step-by-step guide to world class concurrent engineering*. ASME Press series on international advances in design productivity, ASME Press. 15

DANILOVIC, M. & BÖRJESSON, H. (2001). Managing the Multiproject Environment. In *The Third Dependence Structure Matrix (DSM) International Workshop*, 1–17. 59

DE WECK, O.L. & SUH, E.S. (2006). Flexible Product Platforms : Framework and Case Study. *Proceedings of IDETC/CIE 2006 ASME 2006 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, 1–17. 16

DEPARTMENT OF DEFENSE, U. (1988). MILITARY STANDARD CONFIGURATION CONTROL- ENGINEERING CHANGES, DEVIATIONS AND WAIVERS. 20

DOWLATSHAHI, S. (1997). The role of product design in designer-buyer-supplier interface. *Production planning & control*, **8**, 522–532. 26

ECKERT, C., CLARKSON, P. & ZANKER, W. (2004). Change and customisation in complex engineering domains. *Research in Engineering Design*, **15**, 1–21. 4, 5, 16, 19, 20, 21, 24

ECKERT, C., CLARKSON, J. & EARL, C. (2005). PREDICTABILITY OF CHANGE IN ENGINEERING : A COMPLEXITY VIEW. In *ASME 2005 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, 1–10. 20, 22, 24

ECKERT, C.M. & STACEY, M.K. (2003). The spiral of applied research : a methodological view on integrated design research. In *International Conference on Engineering Design ICED 03*, 1–10. vii, 7, 8

ECKERT, C.M., KELLER, R., EARL, C. & CLARKSON, P.J. (2006). Supporting change processes in design : Complexity, prediction and reliability. *Reliability Engineering and System Safety*, 1521–1534. 19

EGER, T., ECKERT, C. & CLARKSON, P.J. (2005). The role of design freeze in product development. In *International Conference on Engineering Design ICED 05*, 1–11. 3, 15, 42, 82

FEI, G., GAO, J., OWODUNNI, D. & TANG, X. (2011). A Model-driven and Knowledge-based Methodology for Engineering Design Change Management. *Computer-Aided Design and Applications*, **8**, 373–382. 19

FRICKE, E. & SCHULZ, A.P. (2005). Design for changeability (DfC) : Principles To Enable Changes in System Throughout Their Entire Lifecycle. *System Engineering*, **8**, 342–359. 4, 17, 19, 23

FRICKE, E., GEBHARD, B., NEGELE, H. & IGENBERGS, E. (2000). Coping with Changes : Causes, Findings, and Strategies. *System Engineering*, **3**, 169–179. 3, 15, 16, 87

GERO, J.S. & KANNENGIESSER, U. (2004). The situated function–behaviour–structure framework. *Design Studies*, **25**, 373–391. 44, 45

GONZALEZ-ZUGASTI, J.P., OTTO, K.N. & BAKER, J.D. (2000). A Method for Architecting Product Platforms. *Research in Engineering Design*, **12**, 61–72. 16

HAMRAZ, B., CALDWELL, N.H.M. & JOHN CLARKSON, P. (2012). A Multidomain Engineering Change Propagation Model to Support Uncertainty Reduction and Risk Management in Design. *Journal of Mechanical Design*, **134**, 100905. 19

HAMRAZ, B., CALDWELL, N.H. & CLARKSON, P.J. (2013). A Matrix-Calculation-Based Algorithm for Numerical Change Propagation Analysis. *IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT*, **60**, 186–198. 19

HANF, J.H. & PALL, Z. (2009). New dimension in retailing- retailer as focal company. In *4th Aspects and Visions of Applied Economics and Informatics*, 349–355. 3

HARTLEY, J., MEREDITH, J., MCCUTCHEON, D. & KAMATH, R. (1997). Suppliers' contributions to product development : an exploratory study. *Engineering Management, IEEE Transactions on*, **44**, 258–267. vii, 26

HIRTZ, J., B STONE, R., A MCADAMS, D., SZYKMAN, S. & L WOOD, K. (2002). A functional basis for engineering design : Reconciling and evolving previous efforts. *Research in Engineering Design*, **13**, 65–82. 44

HÖLTTÄ, V., EISTO, T. & MAHLAMÄKI, K. (2009). Benefits for cast product development through early supplier involvement. In *15th International Conference on Concurrent Enterprising (ICE 2009)*. 26

HUANG, G., YEE, W. & MAK, K. (2001). Development of a web-based system for engineering change management. *Robotics and Computer-Integrated Manufacturing*, **17**, 255 – 267. 19

HUANG, G.Q. & MAK, K.L. (1998). Computer aids for engineering change control. *Journal of Materials Processing Technology*, 187–191. 19

HUANG, G.Q., HUANG, J. & MAK, K.L. (2000). Agent-based workflow management in collaborative product development on the Internet. *Computer-Aided Design*, **32**, 133–144. 3, 24, 28

HUANG, G.Q., YEE, W.Y. & MAK, K.L. (2003). Current practice of engineering change management in Hong Kong manufacturing industries. *Journal of Materials Processing Technology*, **139**, 481–487. 19

IBBS, C.W., WONG, C.K. & KWAK, Y.H. (2001). Project change management system. *Journal of management in engineering*, **17**, 159–165. 23

IBRAHIM, M.A. (2003). A Systematic Approach to Modelling Change Processes in Construction Projects. *The Australian Journal of Construction Economics and Building*, **5**, 1–7. 20

IEEE (2011). Ieee guide–adoption of the project management institute (pmi(r)) standard a guide to the project management body of knowledge (pmbok(r) guide)–fourth edition. 127

JARRATT, T.A.W. & CLARKSON, P.J. (2005). *Engineering change. Design process improvement : a review of current practice*. Springer, London, UK. 23, 24

JENNINGS, N.R. & SYCARA, K. (1998). A roadmap of agent research and development. *Journal of Autonomous Agents and Multi-Agent Systems*, **1**, 7–36. 28

JIAO, J., TSENG, M.M., MA, Q. & ZOU, Y. (2000). Generic Bill-of-Materials-and-Operations for High-Variety Production Management. *Concurrent Engineering*, **8**, 297–321. 45

KELLER, R., EGER, T., ECKERT, C.M. & CLARKSON, P.J. (2005). Visualising Change Propagation. In *INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN ICED 05*, 1–12. 19

KELLER, R., ALINK, T., PFEIFER, C., ECKERT, C.M., CLARKSON, P.J. & ALBERS, A. (2007). PRODUCT MODELS IN DESIGN : A COMBINED USE OF TWO MODELS TO ASSESS CHANGE RISKS. In *INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'07*, 1–12. 23, 24

KERZNER, H. (2009). *Project Management*. A system approach to planning, scheduling and controlling, John Wiley & Sons, Inc., 10th edn. 21, 38, 74

KOCAR, V. & AKGUNDUZ, A. (2010). Advice : A virtual environment for engineering change management. *Computers in Industry*, **61**, 15 – 28. 19, 24

KORNGOLD, J. & LUSCHER, T. (2000). *Product Design for Ease of Assembly-DFA/DFM Manual*. 15

KUHN JR, J.R., COURTNEY, J.F., MORRIS, B. & TATARA, E.R. (2001). Multiagent systems engineering. *International Journal of Software Engineering and Knowledge Engineering*, **11**, 231–258. 28

KUHN JR, J.R., COURTNEY, J.F., MORRIS, B. & TATARA, E.R. (2010). Agent-based analysis and simulation of the consumer airline market share for frontier airlines. *Knowledge-Based Systems*, **23**, 875–882. 28

LAWRENCE, D. & DAVID, O. (1999). Decision support system for the management of systems change. *Technovation*, **19**, 483–493. 16, 23

LI, H. & AZARM, S. (2002). An approach for product line design selection under uncertainty and competition. *Journal of Mechanical Design*, **124**, 385–392. 15

MA, Y., CHEN, G. & THIMM, G. (2008). Change propagation algorithm in a unified feature modeling scheme. *Computers in Industry*, 110–118. 17

MACAL, C.M. & NORTH, M.J. (2001). Tutorial on agent-based modeling and simulation part 2 : How to model with agents. Proceedings of the 2006 Winter Simulation Conference. xi, 28, 29

MAJESKE, K.D., LYNCH-CARIS, T. & HERRIN, G. (1997). Evaluating product and process design changes with warranty data. *International Journal of Production Economics*, **50**, 79–89. 22

MARTIN, M.V. & ISHII, K. (2002). Design for variety : developing standardized and modularized product platform architectures. *Research in Engineering Design*, **13**, 213–235. 15

MIKE, D. & BROWNING, T.R. (2007). Managing complex product development projects with design structure matrices and domain mapping matrices. *International Journal of Project Management*, 300–314. 57, 59

MORGAN, J.M. & LIKER, J.K. (2006). *The Toyota product development system*. Productivity press New York. 26

NICHOLS, K. (1990). Getting engineering changes under control. *Journal of Engineering Design*, **1**, 5–15. 3

OUERTANI, M.Z. & GZARA, L. (2008). Tracking product specification dependencies in collaborative design for conflict management. *Computer-Aided Design*, **40**, 828–837. 59

PICH, M.T., LOCH, C.H. & DE MEYER, A. (2002). On uncertainty, ambiguity, and complexity in project management. *Management Science*, 1008. 33

REDDI, K.R. & MOON, Y.B. (2009). A framework for managing engineering change propagation. *International Innovation and Learning*, **6**, 1–17. 19

RIVIERE, A., DACUNHA, C. & TOLLENAERE, M. (2002). PERFORMANCES IN ENGINEERING CHANGES MANAGEMENT. In *IDMME 2002*, 1–10. 16, 19, 20, 21

ROUIBAH, K. & CASKEY, R.K. (2003). Change management in concurrent engineering from a parameter perspective. *Computers in Industry*, **50**, 15–34. 20, 21

RUTKA, A., GUENOV, M.D., LEMMENS, Y., SCHMIDT-SCHÄFFER, T., COLEMAN, P. & RIVIÈRE, A. (2006). Methods for engineering change propagation analysis. In *25TH INTERNATIONAL CONGRESS OF THE AERONAUTICAL SCIENCES*, 1–8. 22

SARGENT, R.G. (2012). Verification and validation of simulation models. *Journal of Simulation*, **7**, 12–24. viii, 127

SCHULZ, A.P., CLAUSING, D.P., NEGELE, H. & FRICKE, E. (1999). Shifting the view in systems development-technology development at the fuzzy front end as a key to success. In *1999 ASME DETC, 11th International Conference on Design Theory and Methodology*. 15

SHIAU, J.Y. & WEE, H.M. (2008). A distributed change control workflow for collaborative design network. *Computers in Industry*, 119–127. 19

SMITH, P.G. & REINERTSEN, D.G. (1997). *Developing Products in Half the Time : New Rules, New Tools*. Wiley, 2nd edn. 15

SUH, N.P. (1990). *The principles of design*, vol. 990. Oxford University Press New York. 44, 46

TANG, D., ZHU, R., TANG, J., XU, R. & HE, R. (2010). Product design knowledge management based on design structure matrix. *Advanced Engineering Informatics*, **24**, 159–166. 19

ULRICH, K.T. & EPPINGER, S.D. (2007). *Product Design and Development*. Mc Graw-Hill, 4th edn. 3, 35, 45, 59, 184

UMEDA, Y. & TOMIYAMA, T. (1997). Functional Reasoning in Design. *IEEE Expert*, **12**, 42–48. 46

WIE, M.V., BRYANT, C.R., BOHM, M.R., MCADAMS, D.A. & STONE, R.B. (2005). A model of function-based representations. *AIE EDAM*, **19**. 45

WILLIAMS, T. (2000). Safety regulation changes during projects : the use of system dynamics to quantify the effects of change. *International Journal of Project Management*, **18**, 23 – 31. 22

ZANGWILL, W.I. (1993). *Lightning strategies for innovation : how the world's best companies create new products*. Lexington Books, New York, USA. 3

ZHANG, X., ZOLGHADRI, M., LECLAIRE, P. & GIRARD, P. (2013). A Co-evolution Oriented Change Analysis Framework in Product Development Project. In *6th International Conference on Management and Control of Production and Logistics MCPL2013*, 8. 17

ZOLGHADRI, M., ECKERT, C., ZOUGGAR, S. & GIRARD, P. (2009). A TAXONOMY OF COLLABORATION IN SUPPLY CHAINS. In *INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'09*, 1–12. 26

ZOLGHADRI, M., BARON, C. & GIRARD, P. (2010). Modelling mutual dependencies between products architecture and network of partners. *International Journal of Product Development*, **10**, 62–86. 59

ZOUGGAR, S., ZOLGHADRI, M. & GIRARD, P. (2009). Modelling Product and Partners Network Architectures to Identify Hidden Dependencies. In *Proceedings of the 19th CIRP Design Conference*, 32–39. 59