Université de Toulouse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Institut National Polytechnique de Toulouse (INP Toulouse)

Discipline ou spécialité :

Génie industriel

Présentée et soutenue par :
Wenxin Mu

le : mardi 20 novembre 2012

Titre :

Caractérisation et logique d'une situation collaborative

Ecole doctorale :
Systèmes (EDSYS)

Unité de recherche :
Centre Génie Industriel - Ecole des Mines d'Albi-Carmaux

Directeur(s) de Thèse :

Hervé Pingaud

Rapporteurs :
Jean-Pierre Bourey
David Chen
Keith Popplewell

Membre(s) du jury :

Hervé Panetto
Jean-Pierre Lorre
Frédérick Bénaben

Abstract: MISE 2.0 (for Mediation Information System Engineering, second iteration) project has been launched in 2009. The MISE 2.0 engineering approach is based on BPM (Business Process Management) and MDE (Model-Driven Engineering). Running a regular BPM approach on a specific organization consists in gathering structural, informational, and functional knowledge in order to design cartography of processes covering the behavior of the modeled organization. Regarding the MISE 2.0 project the BPM approach concerns a set of organizations and MDE helps in automatizing the different steps: i) Knowledge gathering (situation layer): collect information concerning the collaborative situation, ii) Processes cartography design (solution layer): design the processes according to the knowledge gathered and iii) MIS deployment (implementation layer): implement an IT structure able to run the processes cartography.

Both the transitions between these layers are the hard-points of this approach: The first gap is managed at the abstract level of MISE 2.0 while the second one is managed at the concrete level of MISE 2.0. The current PhD is focused on the first issue: designing a relevant processes cartography from the modeled collaborative situation. However, this is usually a manual activity, which requires a large amount of work to draw the processes and their links. The current research works aim at building such collaborative process cartography in an automated manner. Our principles are (i) to gather the essential and minimum initial collaborative knowledge (e.g. partners, shared functions and collaborative objectives) in models, ii) to deduce the missing knowledge with the help of a collaborative metamodel, an associated ontology and transformation rules and iii) to structure the deduced knowledge in a collaborative process cartography thanks to dedicated algorithms.

Keywords: Enterprise Interoperability, MDA, Model Transformation, Ontology, Mediation Information System, Collaborative Process, and SaaS.

Résumé: Initié en 2009, le projet MISE 2.0 (deuxième itération du projet Mediation Information System Engineering) s'articule autour d'une approche BPM (pour Business Process Management) et d'une vision MDE (pour Model-Driven Engineering). La réalisation d'une démarche BPM classique au sein d'une organisation nécessite de recueillir une connaissance couvrant à la fois les aspects structurel, informationnel et fonctionnel afin de définir des modèles de processus caractéristiques du comportement de l'organisation. Concernant le projet MISE 2.0, l'approche BPM considérée concerne un ensemble d'organisations collaboratives. Quant à la composante MDE, elle est destinée à faciliter l'automatisation des différentes étapes de la démarche : i) Recueil de la connaissance (caractérisation de la situation) : Il s'agit de collecter les information concernant la situation collaborative considérée, ii) Déduction de la cartographie de processus collaboratifs (définition de la solution) : il s'agit de définit les processus collaboratifs adaptés à la situation collaboratives caractérisée au niveau precedent and iii) Déploiement du SI de médiation (implémentation de la solution) : il s'agit d'implémenter le SI de médiation sous la forme d'une plateforme informatique capable d'orchestrer les processus collaboratif définis.

La problématique scientifique relève des deux transitions entre ces trois niveaux d'abstractions : la première transition est prise en charge au niveau abstrait de la démarche MISE 2.0 alors que la seconde est traitée au niveau concret. Les travaux de thèse dont il est ici question se focalisent sur le niveau abstrait : déduction d'une cartographie de processus collaboratifs satisfaisant la situation collaborative considérée. Ce type d'objectif relève généralement d'activités entièrement manuelles qui nécessitent une importante quantité de travail afin d'obtenir les modèles de processus escomptés. Les travaux de recherches présentés ambitionnent d'automatiser cette démarche. Le principe est le suivant : (i) recueil, sous la forme de modèles, de la connaissance nécessaire à la caractérisation de la situation collaborative (informations sur les partenaires, les fonctions qu'ils partagent et leurs objectifs), (ii) déduction de la connaissance complémentaire relative à la dynamique collaborative qui pourrait satisfaire ces objectifs selon les moyens disponibles (cette phase s'appuie sur un métamodèle collaboratif, sur l'ontologie associée et sur des règles de transformation) et (iii) structuration de cette connaissance générée sous la forme d'une cartographie de processus collaboratifs (grâce à des algorithmes dédiés).

Mots Clés: Interopérabilité d'Entreprise, MDA, Transformation de Modèle, Ontology, Système d'Information de Médiation, Processus Collaborative, et SaaS.

Acknowledgements

It has been nearly four years since I arrived at the train station in Albi to begin my master internship. I always remember the sunshine of that day and the shining smile on the face of Frédérick. As a foreigner, I am afraid to move to a new place and to meet strangers and new friends. But Frédérick uses his kindness and humor to make me feel much better. It was a nice beginning in Albi with you all. During three years of thesis, he helped and taught me to understand the research work, to write papers, and to present the research work in an understandable way. One thing I learn from him is that: "There is always a solution". I knew Hervé in the class of Petri-net in Bordeaux University 1. He speaks good English with strong French accent, somehow very cute ^0^. I never imagined that he would be my PhD director at that time. He is a professor, who would care about the professional work and the personal feelings of his students. It is a pleasure and honor to work with him and to be his student. The next winner is Sébastien: the "baby face". It is really grateful to be in the same office with him. He gives me lots of helps about coding and developing software tools, joining French culture and speaking French. And he also tells a lot of French jokes in French and in English again. Isabelle is the secretary of our lab. I believe nobody would forget her big round eyes. I love the way she speaks French. The way she talks provides the listener strong emotional feelings. I just like her. She is so kind to help me deal with all the complex paper duties. She also corrects the English faults of my papers, even though this is definitively not her work.

Frédérick Bénaben, Hervé Pingaud, Sébastien Truptil and Isabelle Fournier, thank you for taking care of me and thank you for being around me.

I would thank Mr. Jean-Pierre Bourey, Mr. Keith Popplewell, Mr. David Chen, Mr. Hervé Panetto and Mr. Jean-Pierre Lorré to be my jury of the PhD defense.

In my personal life, I would thank my dear Blanc Joël and Marie-Claire. I remember the garden, the swimming pool, the cakes, the chocolate, the honey, the lavender and the…each word presents a small story, which I would remember in my whole life. When I was ill, thank you for taking care of me. When I was busy, thank you for the cakes beside my door. When I was sad or happy, thank you for listening.

And Qian, she is the first Chinese friend, who I met in Albi. We eat together, chat together, laugh together, and travel together. With her, I am not alone in France. She provides me a place to talk and to rest. And Zhao, the second friend I met. She is a true person, who always talks straightly what she thinks. I am so happy to have two of you in my life. THANK all the people in EMAC, Dolores, Denis, Anne-Marie, Nicolas, Elise, Guillaume, Aurelian, Olfa, François, Sarah… and all I did not mention. Thanks a lot. All of you are so kind and nice to me. I really enjoy the days with all of you. In Chinese: **谢谢**.

最后的最后，爸爸妈妈，我爱你们。拥有你们，我才会如此幸福。

Contents

General Introduction 1

Chapter I Problem Statement 5

I. MISE 1.0 Introduction and Assumptions 6

I.1. Objective of MISE 1.0 6

I.2. MISE 1.0 Introduction 7

I.3. MISE 1.0 Business Level Comparison 9

I.3.1. Comparison of Theoretical Principles 9

I.3.2. Comparison of Engineering Methods 13

I.4. Assumptions and Limits on MISE 1.0 15

I.4.1. Assumptions and Limits on CIM 15

I.4.2. Assumptions and Limits on PIM 16

I.4.3. Assumptions and Limits on MISE 1.0 Model-Driven Approach 16

II. MISE 2.0 Proposal 17

II.1. The Abstract Level of MISE 2.0 18

II.2. Related Works 20

II.2.1. ARIS Method Extension 21

II.2.2. Situation Calculus 23

II.2.3. Business Process Abstraction 25

III. MISE 2.0 Business Level Design 25

IV. Conclusion 28

Chapter II Collaborative Situation Framework 29

I. Introduction 30

II. State of Art 31

II.1. Enterprise Architecture Framework 32

II.1.1. CIMOSA 32

II.1.2. GERA 34

II.1.3. ARIS 36

II.2. Enterprise Interoperability Framework 37

II.2.1. IDEAS 37

II.2.2. AIF 38

II.2.3. EIF 39

II.3. Conclusion 41

III. Collaborative Situation Framework 44

III.1. Framework Introduction 44

III.2. Relations of Collaborative Elements 45

III.3. Relations of Collaborative Lifecycle 46

III.4. Relations of Collaborative Levels 47

IV. Conclusion 48

Chapter III Models 51

I. Introduction 52

II. Collaborative Network Model 54

II.1. Definition of Collaborative Network Model 54

II.2. Example of Collaborative Network Model 59

III. Function Model 60

III.1. Definition of Function Model 60

III.2. Example of Function Model in MISE 2.0 65

IV. Collaborative Process Model 67

IV.1. Definition of Collaborative Process Model 67

IV.2. Example of Collaborative Process Model 70

V. Conclusion 75

Chapter IV Model Transformation 77

I. Introduction 78

II. Reference Ontologies 79

II.1. Definition of Ontology 79

II.2. Ontology of MIT Process Handbook 80

II.3. Collaborative Network Ontology of MISE 1.0 82

III. MISE 2.0 Collaborative Ontology 84

III.1. The Collaborative Concepts 85

III.2. The Mediation Concepts 88

III.3. Transformation Rules 90

IV. Service Selection and Sequence Deduction 99

IV.1. Business Service Selection 100

IV.2. Process Sequence Deduction 105

IV.2.1. Process Deduction 105

IV.2.2. Sequence Deduction 109

V. Conclusion 111

Chapter V Final Result 115

I. Introduction 116

II. Mediator modeling 2ool 117

II.1. Development 118

II.2. Implementation 120

III. Example 123

IV. Conclusion 132

General Conclusion 135

Acronyms 141

References 145

Annex A 153

Annex B 167

List of Figures

Chapter I Problem Statement

Figure I - 1 Mediation Information System 7
Figure I - 2 The global structure of MISE 1.0 8
Figure I - 3 Agility of MISE 1.0 9
Figure I - 4 Model Driven Architecture 10
Figure I - 5 Model Driven Interoperability (Bourey et al., 2007) 11
Figure I - 6 Model Transformation of The BVR (Jouault and Kurtev, 2006) 12
Figure I - 7 Model Transformation of The BST (Truptil et al., 2010) 13
Figure I - 8 Comparison of Business Level Engineering of BVT and BST 14
Figure I - 9 Overview of MISE 2.0 Project 18
Figure I - 10 The Target of MISE 2.0 Abstract Level 19
Figure I - 11 Problems to Solve in MISE 2.0 Abstract Level 20
Figure I - 12 SOA metamodel (Stein et al., 2008) 22
Figure I - 13 ARIS objects grouped according to MDA levels (Stein et al., 2008) 23
Figure I - 14 Automatic Composition and Model Verification 24
Figure I - 15 Allocation of business process model abstraction on MOF levels (Smirnov et al., 2012) 25
Figure I - 16 Global picture of the design of MISE 2.0 business level (UJ picture) 26

Chapter II Collaborative Situation Framework

Figure II - 1 Position of Chapter II in UJ picture 30
Figure II - 2 From Integration to Engineering (Kosanke et al., 1999) 33
Figure II - 3 CIMOSA Modeling Framework (Zelm et al., 1995) 34
Figure II - 4 Relationships between lifecycle and life history (Kosanke et al., 1999) 35
Figure II - 5 GERA Modeling Framework 36
Figure II - 6 ARIS Modeling Framework (Idee, 2009) 37
Figure II - 7 IDEAS interoperability framework (IDEAS, 2003) 38
Figure II - 8 ATHENA reference architecture (Berre et al., 2007) 39
Figure II - 9 Enterprise interoperability fraemwork of InterOp (Chen et al., 2008) 40

Figure II - 10 Framework of Collaborative Situation 44

Figure II - 11 Relations among Collaborative Situation Elements 46

Figure II - 12 Relations in Collaborative Situation Lifecycle 47

Figure II - 13 Relations among Collaborative Situation Levels 48

Figure II - 14 Position of MISE 1.0 abstract level in Collaborative Situation Framework 49

Figure II - 15 Position of MISE 2.0 abstract level in Collaborative Situation Framework 49

Chapter III Models

Figure III - 1 Organization model, function model and collabroative process model 53

Figure III - 2 Position of Chapter III in UJ 53

Figure III - 3 Example of Chapter II 54

Figure III - 4 Modeling process of an example (Jiang et al., 2011) 55

Figure III - 5 Collaboration diagram between typical roles (Bouslimi et al., 2009) 56

Figure III - 6 The definition of the organization model 58

Figure III - 7 Level 1 and Level 2 of the collaborative network model 59

Figure III - 8 Level 2 and Level 3 of the collaborative network model 60

Figure III - 9 Modeling unit of IDEF0 (PUBS, 1993) 61

Figure III - 10 Decomposition of Function (PUBS, 1993) 62

Figure III - 11 Modeling Unit of Function Model 63

Figure III - 12 Results of Transformation Rule (1) 64

Figure III - 13 completed A0: network 65

Figure III - 14 Completed A2: Sell Product 65

Figure III - 15 Function Models of Supplier 1, 2 and 3 66

Figure III - 16 Function Models of Assembler and Client 67

Figure III - 17 Collaborative Process Model of MISE 1.0 68

Figure III - 18 Additional Process Cartography of MISE 2.0 69

Figure III - 19 Relations of Collaborative Process Cartography and Collaborative Process Model 70

Figure III - 20 Process cartography of A0 Network 71

Figure III - 21 Process Cartography of A2 Sell product 72

Figure III - 22 Collaborative process of Choose Partner 73

Figure III - 23 Collaborative process of Send Payment 74

Figure III - 24 Collaborative process of Deliver Product......75

Figure III - 25 Positions of Models in MISE 2.0 Framework 76

Chapter IV Model Transformation

Figure IV - 1 Position of Chapter IV in UJ 78

Figure IV - 2 Graphical representation of the MIT Process Handbook ontology schema 82

Figure IV - 3 The collaborative ontology of MISE 1.0 (Rajsiri et al., 2010)......84

Figure IV - 4 Collaborative Ontology of MISE 2.0......85

Figure IV - 5 Partner, Network and Objectives 86

Figure IV - 6 Same As and Near By......87

Figure IV - 7 Main Function, Function and Business Message......87

Figure IV - 8 Relations among Organizational and Functional......88

Figure IV - 9 Mediator and Mediator Relationship 89

Figure IV - 10 Mediator Function and Inter Mediator Function......90

Figure IV - 11 Example of Group 1......92

Figure IV - 12 Example of Group 2......95

Figure IV - 13 Example of Group 3......96

Figure IV - 14 Example of Group 4......98

Figure IV - 15 Example of Group 5......99

Figure IV - 16 Example of chapter IV......100

Figure IV - 17 Instances of collabroative ontology and models......101

Figure IV - 18 Results of deduction 102

Figure IV - 19 Deduction of collaborative process cartography......106

Figure IV - 20 Deduction of collaborative process 107

Figure IV - 21 Deduction of partner pool......108

Figure IV - 22 Deduction of mediator pool......109

Figure IV - 23 Deduce sequences by input/output messages 110

Figure IV - 24 Link functions with objectives......110

Figure IV - 25 Example of process deduction......111

Figure IV - 26 Comparisons of collaborative ontologies of MISE 1.0 and MISE 2.0 112

Chapter V Final Result

Figure V - 1 Position of chapter V in the UJ 116

Figure V - 2 Main use case diagram of mediator modeling 2ool......118

Figure V - 3 Main flows of “Define models” 119

Figure V - 4 Main flows of “Transfer model” 119

Figure V - 5 Main interface (palette and property) 121

Figure V - 6 Main interface (file and ontology) 122

Figure V - 7 Collaborative ontology in protégé......123

Figure V - 8 Create new project 124

Figure V - 9 Added new folder and files 124

Figure V - 10 Level 0 collaborative network model 125

Figure V - 11 Define same as/near by instance for “design product” 125

Figure V - 12 Define sub model......126

Figure V - 13 Sub network of “design product” 126

Figure V - 14 Sub network of “Sell product” 127

Figure V - 15 Sub network of “Design offer strategy” 127

Figure V - 16 Define function model......128

Figure V - 17 Define same as/near by instances for function......128

Figure V - 18 “Transfer model” menu......129

Figure V - 19 .xml file 129

Figure V - 20 XML file of the selection result......130

Figure V - 21 .bpmn file of collabroative process cartography 130

Figure V - 22 Import .bpmn to Petals BPM 131

Figure V - 23 The collabroative process cartography in Petals BPM......131

Figure V - 24 .bpmn files of collabroative processes 132

Figure V - 25 collabroative process of “Design offer strategy” 132

Figure V - 26 Theory of ontology definistion tool 133

Figure V - 27 Main interface of Ontology definition tool 134

General Conclusion

Figure 1 Global view of MISE 2.0 138

Figure 2 Global picture of MISE project 139

List of Tables

Chapter I Problem Statement

Table I - 1 Current documentation of processes (N=127, Na=3) (Patig et al., 2010)....21

Chapter II Collaborative Situation Framework

Table II - 1 Summary of Enterprise Architecture....43

Chapter III Models

Table III - 1 Summary of IOCM, OMOA and OMCK....57
Table III - 2 Transformation Rule 1: objective to function....63

Chapter IV Model Transformation

Table IV - 1 Transformation Rules....91
Table IV - 2 Group 1 Create Mediator....91
Table IV - 3 Group 2 Create Mediator Relationship....93
Table IV - 4 Group 3 Create Generated Mediator Function....96
Table IV - 5 Group 4 Link Generated Mediator Function to Mediator....97
Table IV - 6 Group 5 Create Inter Mediator Function....99

Chapter V Final Result

Table V - 4 Summaries of packages....120

General Introduction

In the world marketing, to seek the maximum of the benefits and to reduce the costs, the enterprises search for supplier, manufacturer, assembler and clients all over the world. The enterprises are obliged to deal with enterprises with different culture background, with people who speaks different language, with information systems in different software environment, etc. In order to cooperate in an efficient manner, the enterprises have to find a way to understand each other concepts, to understand each other languages and to make interoperable each other information systems. In order to achieve such objectives, the whole working process could be changed; organizations and information systems could be turned upside down. This addresses a problem: how to work with other partners without costing and changing so much? This is the research topic of enterprise interoperability.

The enterprise interoperability can be seen as the capacity of enterprises to structure, formalize, and present their knowledge and know-how in order to be able to exchange or share it. To enhance the interoperability of the enterprises, there are numerous methods, tools or languages, which are developed. In (Nicolle et al.), various architectures for the interoperation of information systems are introduced, summarized and compared. These architectures are the following Peer-to-Peer (Milojicic et al., 2002), Standardization[1], Federation[2], Multi-base[3], Ontology (Guarino and Giaretta, 1995) and Mediation (Wiederhold, 1992). Considering the weak point of adding a new partner (and its IS[4]) which requires many translators, Peer-to-Peer and Standardization could not be the first choice. Considering the difficulty of building common standard and language, Federation and Multi-base are removed. Although mediation information system (MIS) requires the difficult task of constructing automatically collaborative process, MIS still is a credible and pertinent way of supporting ISs interoperability.

In order to build and run the collaborative process automatically, MDA (Miller et al., 2003) and ESB[5] have been selected in MISE 1.0 project (2004-2010) as a development approach and implementation tool. Yet, one strong hypothesis we base our research work on, is that partners' IS are supposed to follow the same conceptual logical architectural style: Service Oriented Architecture (SOA) (Vernadat, 2007). This leads to a context where each partner is able to contribute to the collaboration space through interfaces from different levels (computer independent or platform independent). The MIS is able to deal with the three functions identified below among a set of SOA partners' ISs. It can i) gather knowledge of partners' data, ii) build a repository of partners' services and iii) deduce a collaborative process model that can run and a workflow engine that enables to run it.

Dr. Jihed Touzi wrote the first PhD thesis of MISE 1.0 (2004-2007) completed by a post-doc. His work addresses the conceptualization of the logic and technological models of MIS. Dr. Vatcharaphun Rajsiri did the second PhD thesis of MISE 1.0 (2006-2009). Her work is the input of the first one and aims at deducing a business collaborative process by gathering necessary collaborative knowledge. Dr. Sébastien Truptil takes the work the Dr. Jihed Touzi as input and deduces the BPEL based technical collaborative process, which is deployed and executed on an ESB.

The collaborative process developed by Dr. Vatcharaphun Rajsiri, is a single but complex process. The process covers strategy, operation and support level. This leads to the first problem: the partners of the collaborative network come from different apartments of enterprises. They have different major and education background. This means that the partners cannot understand the whole collaborative process but a part of it. Thus the collaborative process is so complex that it is difficult for partners to find the part, which they understand. The collaborative process cartography is very necessary.

The definition of such cartography of collaborative process became the goal of abstract level of MISE 2.0 project (2009-2013) The collaborative process cartography represents the collaborative process in to three

[1] Standardization uses pivot, canonical model or metamodel to reduce the number of translators (similar to Peer-to-Peer).
[2] Federation derives from standardization and uses a global, static federated schema.
[3] Multi-base uses a single language for many ISs.
[4] Information System
[5] Enterprise Service Bus

levels: strategy, operation and support. If the partner's role is manager, designer or marketing, they only needs to understand the collaborative processes in the strategy level and the communications with other levels. If the partner's role is producer, deliverer or assembler, they are interested in the collaborative processes in the operation level. If the partner's role is cleaner, driver or cooker, they are concerned by the collaborative processes in the support level. The goal of the research works in this thesis is to build the collaborative process cartography as automatically as possible by using MDA and SOA.

In order to define the collaborative process cartography, the first thing to know is: what is the collaborative knowledge that can be gathered? And how to model such collaborative knowledge? In a collaborative situation, partners define the collaborative network (organizational). They come with the functions or business services to share (functional) and the collaborative objectives to achieve (complementary of organizational). In the knowledge-gathering phase, the vision is focused on the organizational and functional knowledge. The organizational knowledge includes the collaborative network, the partners, the partner relationships, and the collaborative objectives. The organizational knowledge is gathered and defined by the collaborative network model. The functional knowledge is considered as the functions of partners and input/output messages of the functions. The function model is provided to model this part of knowledge. To answer the question, the collaborative network model and the function model is designed to gather the minimum collaborative knowledge.

If the research work of this thesis is considered as a system, the inputs are the organizational and functional models, then what are the outputs? Or, on a more concrete point of view, how to represent the collaborative process cartography as a model? The representation of the collaborative process cartography is divided in two parts: the process and the cartography. For the process, in the thesis of Dr. Vatcharaphun Rajsiri, the BPMN (Business Process Management Notation) based collaborative process model represents the process. The work of Dr. Vatcharaphun Rajsiri is reused in this thesis to define the collaborative process. For the cartography, the goal is to divide the collaborative process into strategy, operation and support levels. The cartography can be considered as the main collaborative process, which classifies the main tasks by strategy, operation and support. The main tasks can be detailed represented by collaborative processes. To summarize, the collaborative process cartography is defined by the main collaborative process (the cartography) and the sub collaborative process (the processes).

The third thing to discuss is how to play between the input models and the output models to make the system get the input and produce the output? Here, the ontologies and model transformations appear on the board. The ontology organizes and defines the concepts of the input and output models. The model transformation rules can be applied among these concepts to transfer the input models to the output models. But the transformation rule is not the multi-function key. There exist several small black holes, which cannot be filled by the ontology and model transformation. For example, the collaborative process is based on BPMN. The modeling elements of BPMN contain the events and gateways. How can the events and gateways be deduced and inserted to the collaborative process? Further more, there is a gap between objectives and functions. How to select the correct functions to achieve the objectives?

All the above-discussed questions are about the design of the research work. Another essential part is the implementation. The software tool, which can define the input models, implement the transformation rules and transfer the output model, is needed. The software engineering is a domain, which changes and improves every second. The techniques, which are used today, would be disappeared tomorrow. To choose a technique, which can follow the footstep of the main flow and can meet the requirements of the design work, is the main goal of the implementation. Dr. Sébastien Truptil defined the agility management of MISE 1.0 in his thesis. The management requires all the development software tools to be web services, which can be deployed on the ESB. But the developing tools of MISE 1.0 are GMF, ATL and Protégé, which are really difficult to adapt as web service. With the experience of MISE 1.0, it has been chosen that the software tools of MISE 2.0 should be migrated to the web services. In recent years, SaaS (Software as a Service) becomes a hot word in the software engineering world. The SaaS is a good choice for the development of MISE 2.0. But there are a lot of SaaS venders, which vender to choose? This is another question, which should be answered in this thesis.

Until now, the discussion of MISE 2.0 abstract level is accomplished. This thesis is the first part of MISE 2.0, which deduces the collaborative process cartography. The second part of MISE 2.0 is to take the collaborative process cartography and to deduce the technical collaborative workflows (the PhD subject of Nicolas Boissel-Dallier). The third part is the agility management of MISE 2.0 (the PhD subject of Anne-Marie Berth-Delanoë). Besides MISE 2.0, there is MISE 1.0 (2004-2010) and MISE 3.0 coming (2011-?). The whole MISE project is a complex system with numerous knowledge, methods and tools. To well present the work of MISE project, a framework is designed.

The above discussion and identified questions reveals that each topic covers a wide range of aspects and issues which are related to each other. They are studied in detail to gain a clear understanding of them and find a way to manage them. Thus, the thesis is organized as follows:

- Chapter I focuses on describing the scope and objectives of the thesis. This chapter gives a short introduction of MISE 1.0 and the assumptions and limits of MISE 1.0. The MISE 2.0 is presented to show how to fix the limits and improve the previous work in MISE 1.0. Finally, the global design of abstract level of MISE 2.0 (the research work of this thesis) is introduced.
- Chapter II is the collaborative situation framework. This first part is the state of the art of enterprise architecture, enterprise integration framework and enterprise interoperability framework. The second part is the collaborative situation framework of MISE, which presents the main factors and elements and the relations of the factors in the MISE project.
- Chapter III presents the input models and output models of the abstract level of MISE 2.0. This chapter helps the reader to clearly understand what are the needs of these research works? And what are their results? The chapter first addresses the state of the art of organizational model and the definition of collaborative network model followed up with the example of collaborative network model. Secondly, the function model is presented and explained in detail through an example too. Last but not least, the collaborative process cartography and the collaborative process model are presented.
- Chapter IV focuses on presenting the collaborative ontology, the transformation rules, business services selection, and process sequence deduction. The collaborative ontology is presented as two main parts: the concept of collaboration (concepts of input models) and the concept of mediation (concepts of output models). The transformation rules are applied to transfer the concepts of collaboration to the concepts of mediator. To complete the transformation rules, the methodologies of business services selection and process sequence deduction are presented through examples.
- Chapter V aims at presenting the main functions and the global design of the software tool (Mediator modeling 2ool) and demonstrating how the software tool works by experimenting with a very simple collaborative situation. The experimentation uses the tool to perform every step in the global design, from knowledge gathering to the construction of the collaborative process cartography.

Finally, the thesis ends by giving the conclusions and perspectives of the research work. The summary of work and the outlook of MISE project are carried out.

Usually, the literary study is addressed as chapter II. But when the reader starts to read other chapters, the concepts, which are presented in literary study, may be forgotten. In this document, we chose a different manner by writing the literary study according to the reading of the document. Consequently, the literary study is broken into small parts and presented at the beginning of each chapter.

Chapter I: Problem Statement

I. MISE 1.0 Introduction and Assumptions 6
I.1. Objective of MISE 1.0 6
I.2. MISE 1.0 Introduction 7
I.3. MISE 1.0 Business Level Comparison 9
I.3.1. Comparison of Theoretical Principles 9
I.3.2. Comparison of Engineering Methods 13
I.4. Assumptions and Limits on MISE 1.0 15
I.4.1. Assumptions and Limits on CIM 15
I.4.2. Assumptions and Limits on PIM 16
I.4.3. Assumptions and Limits on MISE 1.0 Model-Driven Approach 16
II. MISE 2.0 Proposal 17
II.1. The Abstract Level of MISE 2.0 18
II.2. Related Works 20
II.2.1. ARIS Method Extension 21
II.2.2. Situation Calculus 23
II.2.3. Business Process Abstraction 25
III. MISE 2.0 Business Level Design 25
IV. Conclusion 28

I. MISE 1.0 Introduction and Assumptions

The aim of section I is to explain the objectives of the PhD subject: Business and Logic Characterization of Collaborative Situation and to position the PhD subject into the Mediation Information System Engineering (MISE) 2.0 project structure.

Firstly, we seek to introduce the MISE 1.0 project and explain the limitations of MISE 1.0 project. Then, we present the needs of business and logic characterization and position them in the MISE 1.0 project. Business and logic characterization embedded in the PhD subject solves some of the limitations in MISE 1.0 project.

I.1. Objective of MISE 1.0

Due to the current global and competitive market, the capacity of enterprises to collaborate with their partners is one of the critical factors for their development and their ability to survive. (Touzi, 2007) defined four levels of capacity: communicating (ability to exchange and share information), open (ability to share business services and functionalities with others), federated (ability to work with others by following collaborative processes in order to pursue a common objective, as well as objectives of the enterprise itself), and interoperable (ability to work with others without a special effort; the enterprises involved are seen as a seamless system). Thus, *"interoperability can be seen as an alternative to performing the integration of enterprises into a unique system"* (Vernadat, 2007).

The concept of interoperability first appeared in the domain of computer science in the early 1990s and has been developed continuously and extensively in many domains such as the military, medical, transportation, software ones, etc. Since then, many definitions related to this concept have been proposed. The most quoted one was given by (IEEE, 1990) which define interoperability as *"the ability of two or more systems or components to exchange information and to use the information that has been exchanged"*. The InterOp NoE defines the interoperability as *"the ability of a system or a product to work with other systems or products without special effort from the customer or user"* (Konstantas et al., 2005). According to Pr. Hervé Pingaud, *"interoperability is the capability of systems, natively independent from each other to interact in order to build harmonious and finalized collective behaviors without any deep modification of their own structure or behavior"* (Pingaud, 2009).

Furthermore (Chen et al., 2008) proposes an enterprise interoperability framework, which defines three dimensions, one of them concerns the interoperability barriers: conceptual, technological and organizational. Considering that the information systems of enterprises are the practical and operational part in the enterprises, it is a crucial requirement to break the technological barriers through interoperability among the information systems. The possibility to break organizational and conceptual barriers by breaking technological barriers is also considered. Yet, one strong hypothesis we base our work on, is that partners' information systems are supposed to follow the same conceptual logical model: Service Oriented Architecture (SOA) (Krafzig et al., 2005). (Benaben et al., 2006) proposes the three following main interoperability functions:

- Conversion and delivery of data;
- Management of applications (or services in a SOA context);
- Orchestration of collaborative process.

As shown in Figure I-1, partners want to work together in a collaborative situation. But there are always some barriers. In order to break the technological barrier among information systems, a Mediation Information System (MIS) seems to be a possible and suitable solution for technical interoperability of enterprises' information system. (Benaben et al., 2008) The concepts of mediation are first presented in (Dr et al., 1996). Besides, a MIS should handle: data conversion, processes orchestration and application management at least.

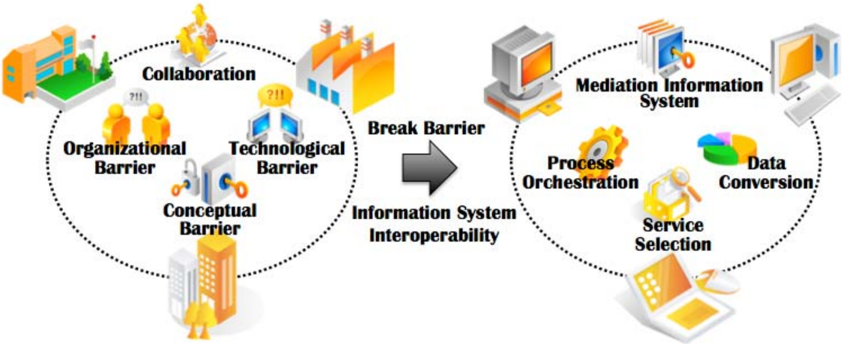


FIGURE I - 1 MEDIATION INFORMATION SYSTEM

I.2. MISE 1.0 Introduction

Since 2004, the Mediation Information System Engineering (MISE) project was launched in the industrial engineering department of Ecole des Mines d'Albi-Carmaux. From 2004 to 2010, MISE 1.0 is designed, tested and evaluated. MISE 1.0 is the huge research work of three doctors and several internship students. The engineering approach of MISE 1.0 is a model-driven engineering (MDE). Figure I-2 shows the global MDA of MISE 1.0. The approach is designed according to three branches: the business branch, the logic branch and the technological branch. Each branch corresponds to one step of MDE. The business branch develops Computer Independent Model (CIM). The logic branch takes the CIM as input and transfers it to the Platform Independent Model (PIM), which is confirmed by the logic metamodel. The technological branch takes the PIM as input and transfers it to the Platform Specific Model (PSM), which respects the physical architecture of the target platform.

The model-driven approach of MISE 1.0 starts with the business branch, which gathers the necessary and minimum business knowledge of collaboration and transfers it to the BPMN[1] based collaborative process model (CIM). The research work on the business level is completed by Dr. Vatcharaphun Rajsiri and explained in (Rajsiri et al., 2010). First, the software tool: Network Editor was developed to collect knowledge (collaborative network, partners' relationships and collaborative common goals) in the phase of gathering the collaborative knowledge. Second, the collaborative ontology and the collaborative process ontology with transformation rules are defined. The knowledge base: the business knowledge of the MIT Process Handbook (Malone et al., 2003) supports the two ontologies. Finally, the transformation rules and the knowledge base, the BPMN based collaborative process is transferred with the help of the collaborative ontology.

The logic branch takes the BPMN based collaborative process as an input and transfers the process to the UML[2] based logic model. The research work of on the logic branch was finished by Dr. Jihed Touzi and presented in (Touzi, 2007). In the logic branch, the metamodel of the collaborative process and the logic metamodel of the UML based logic model are defined. A set of transformation rules between the two metamodels is presented as equations. The BPMN based collaborative process model is transferred to UML based logic model by following the transformation equations. The UML based logic model only contains business knowledge. In order to keep moving on the approach, the technical knowledge (descriptions of web services in WSDL[3] file) has to be imported into the logic model. (Benaben et al., 2010) defines the logic metamodel based technical metamodel. The technical metamodel describes not only business knowledge and but also technical knowledge. The technical knowledge is exported from WSDL files and completed manually and then transferred to the technical model, which is the perfect input data for technical branch.

[1] Business Process Model and Notation
[2] Unified Modeling Language
[3] Web Service Description Language

The technical branch focuses on using business, logic and technical data of technical model and transferring the technical model to the PSM. In MISE 1.0, the PSM is designed for the specific platform: PetalsLink[1] ESB system. The technical branch has been studied by Dr. Sébastien Truptil and detail explained in detail in (Truptil, 2011). The PSM of MISE 1.0 is composed of four main parts: the Service Unit (description of web service), the Service Assembly (deploys web service on ESB), the Service Engine (orchestrates collaborative process) and the BPEL[2] file (describes the execution order of collaborative process). The most important part is the transformation of a BPEL file. In technical branch, a metamodel of the BPEL file is defined. Based on the metamodel of the BPEL file, several algorithms are developed to transfer the technical model to the BPEL file. The final results of the technical branch are: i) to deploy all the Service Unit and the Service Engine of partners' web services on ESB, ii) to provide the BPEL file, ii) to deploy Service Engine with the BPEL file on ESB, and iv) to execute the collaborative process on ESB, which is the Mediation Information System.

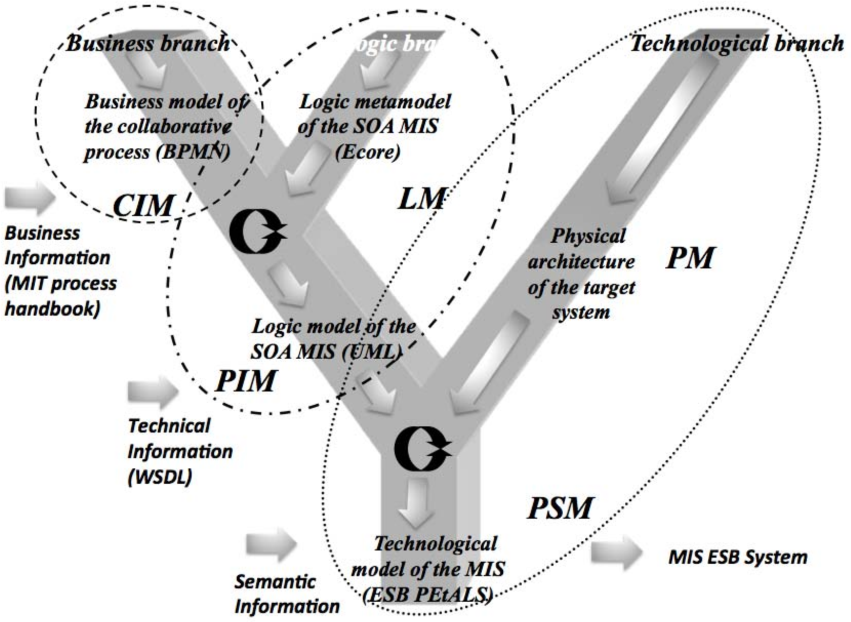

Business branch
Logic bra
Technological branch
Business model of the collaborative process (BPMN)
Logic metamodel of the SOA MIS (Ecore)
CIM
LM
Business Information (MIT process handbook)
Physical architecture of the target system
PM
Logic model of the SOA MIS (UML)
PIM
Technical Information (WSDL)
PSM
Technological model of the MIS (ESB PEtALS)
Semantic Information
MIS ESB System

FIGURE I - 2 THE GLOBAL STRUCTURE OF MISE 1.0

The MISE 1.0 also considers the agility of information system in both the runtime and the design time. Even through, the user uses MISE 1.0 tools to develop a MIS on ESB. There may be some occasional and sudden events, which are triggered at the runtime of MIS (for example, a partner quits, a web service is down or the goal of collaboration has changed). There should be an automatic mechanism to reset MIS and restart the MISE model-driven approach. With this objective, all the software tools and model transformation tools are developed as web services and orchestrated from CIM, PIM and PSM. If the user has to redesign the MIS, the user could just invoke the orchestration of the MISE 1.0 tools to re-launch the whole model-driven approach of the MISE 1.0. The MISE 1.0 is deployed on ESB as a MIS. The MIS is executed to develop another MIS. Figure I-3 shows the solution of agility in MISE 1.0 project. At the bottom, MISE 1.0 orchestration is deployed on ESB at the run-time. All the tools of CIM, PIM and PSM are deployed on ESB as web services of design-time. The tools are invoked by the MISE 1.0 orchestration by following specific orders. On the top of the ESB, MIS orchestration is also deployed on ESB as web services of run-time. Each web service of run-time are plugged and invoked by the MIS orchestration. If the MIS has to be redesigned, the user could invoke MISE 1.0 orchestration of design-time and launch all the tools to deploy a new one.

[1] PetalsLink is a software company who provides open source software tools: http://research.petalslink.org/display/research/Petals+Link+Research+Home

[2] Business Process Execution Language

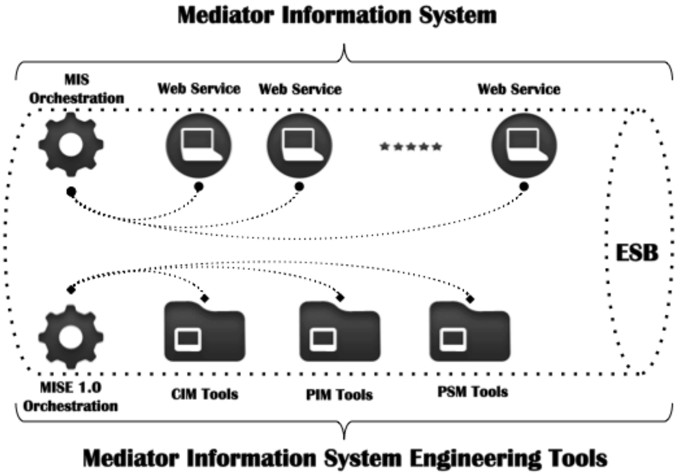


FIGURE I - 3 AGILITY OF MISE 1.0

I.3. MISE 1.0 Business Level Comparison

The engineering approach of MISE 1.0 project is a general approach. A general approach provides a map, a demo or a direction for a general collaboration, which is not in a specific domain. But if a specific situation (e.g. risk management, supply-chain management or product lifecycle management) comes out, it leads to some problems, in specific domain (e.g. education, hospital or police). There is professional knowledge in a specialized field. Due to the particularity of gathered knowledge, the gathered knowledge has to be re-defined. We have to think about if the engineering approach of MISE 1.0 is still adaptive? If it is not, how could the approach be changed? And which part of the approach should be re-defined? Dr. Sébastien Truptil worked on not only the technical branch of MISE 1.0 but also on crisis management domain. He re-used the engineering approach of MISE 1.0 on ISyCri[1] Project. According to chapter II in the PhD thesis (Truptil, 2011) of Dr. Sébastien Truptil, he re-defined specifically the business branch for ISyCri project in crisis management domain. Because the research work of this thesis is located on the business level of collaboration. The objective of this thesis is to define the business process cartography, which the similar with the objective of the research work of Dr. Sébastien Truptil and Dr. Vatcharaphun Rajsiri. It is necessary to make a comparison for these two research works. This section focuses on the discussions of differences on the business branches in general engineering approach (the research work of Dr. Vatcharaphun Rajsiri) and in crisis management (the research work of Dr. Sébastien Truptil). In this section the business branch work of Dr. Vatcharaphun Rajsiri is presented as BVR. The business branch work of Dr. Sébastien Truptil is abbreviated as BST. The discussions of differences are divided into two parts: the differences between based principals (Section I.3.1) and the differences between engineering approaches (Section I.3.2).

I.3.1. Comparison of Theoretical Principles

The BVR is based on Model Driven Architecture (MDA), which is defined by OMG[2] (Miller et al., 2003). MDA provides an approach for, and enables tools to be provided for: *" i) specifying a system independently of the platform that supports it, ii) specifying platforms, iii) choosing a particular platform for the system, and iv) transforming the system specification into one for a particular platform"* (Miller et al., 2003). The three primary goals of MDA are portability, interoperability and reusability through architectural separation of concerns.

MDA is divided into several models (Figure I-4). According to (Miller et al., 2003): the models could be summarized as followed:

[1] Interopérabilité des Systèmes en situation de Crise: Interoperability of Systems in Crisis situation. http://www.irit.fr/isycri/
[2] Object Management Group: http://www.omg.org/index.htm

- CIM (Computation Independent Model): presents the requirements for the system. A CIM is a model of a system that shows the system in the environment in which it will operate, and thus it helps in presenting exactly what the system is expected to do. Such a model is sometimes called a domain model or a business model. It may hide much or all information about the use of automated data processing systems. Typically such a model is independent of how the system is implemented. The CIM is useful, not only as an aid to understand a problem, but also a source of vocabulary that can be shared using another model. The CIM should be traceable to the PIM and PSM constructs that implement them and vice versa;
- PIM (Platform Independent Model): describes the system, but does not show details of how to its platform. It might consist of enterprise, information and computational ODP[1] viewpoints specifications. A PIM exhibits a specified degree of platform independence so as to be suitable for use with a number of different platforms of similar type.
- PSM (Platform Specific Model): is a view of a system from the platform specific viewpoint. A PSM combines the specifications in the PIM with the details that specify how the system uses a particular type of platform. The PSM is in the form of software and hardware manuals. Finally, the code of the implementation of the system is transformed or deduced from the PSM.

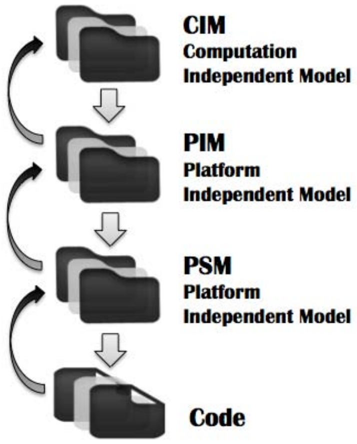


FIGURE I - 4 MODEL DRIVEN ARCHITECTURE

Among CIM, PIM and PSM, there are model transformations or metamodel mappings. The mappings provide the specifications for transformations from a CIM into a PIM and from a PIM into a PSM. The BVR defines the collaborative network model of collaborative situation and adds extra business knowledge from the collaborative ontology into the collaborative network model. The model is transferred into the CIM (for the BVR, it is a BPMN based collaborative process model for CIM) in MDA. Here, a question is proposed: what is the collaborative network model in BVR? Is it the CIM in MDA? If it is not, then what is it? Is it a business model? Is it another CIM? Does the MDA miss a model or a modeling level? Another problem is that: in the interoperability domain, is the MDA still suitable in collaborative situation, which involves several partners?

In 2007, Pr. Jean-Pierre Bourey defines Model Driven Interoperability (MDI) (Bourey et al., 2007). The MDI uses an MDA-like approach to solve interoperability problems. The BST is based on MDI. Figure I-5 shows the structure of MDI. Compared to MDA, the MDI has following different points:

- The MDI separates CIM level into top CIM level and bottom CIM level. Both of them are business levels. The top CIM level is business model, which concerns domain, process, organization, products, strategy and so on. It presents the initial situation in enterprise interoperability. The bottom CIM level is similar with the CIM of MDA. It is the system requirements, which is the result of arbitrations or expectations on what part of the business could be managed by a system.
- The MDI defines the interoperability models in collaboration. Between enterprises (or partners), the MDI provides interoperability models to manage the interoperability at each level (Top CIM - Bottom CIM - PIM - PSM - Code). The mappings from one to another are also defined in the architecture.

[1] Open Distributed Processing

- The MDI involves partners' ESAs[1] in collaboration. The MDI considers that each enterprise or partner owns ESA for each modeling level. Among the levels, the ESA deals with model transformation or metamodel mapping. The interoperability models among enterprises or partners are based on these ESAs, which means the ESAs provide some initial and valuable data for the interoperability models.

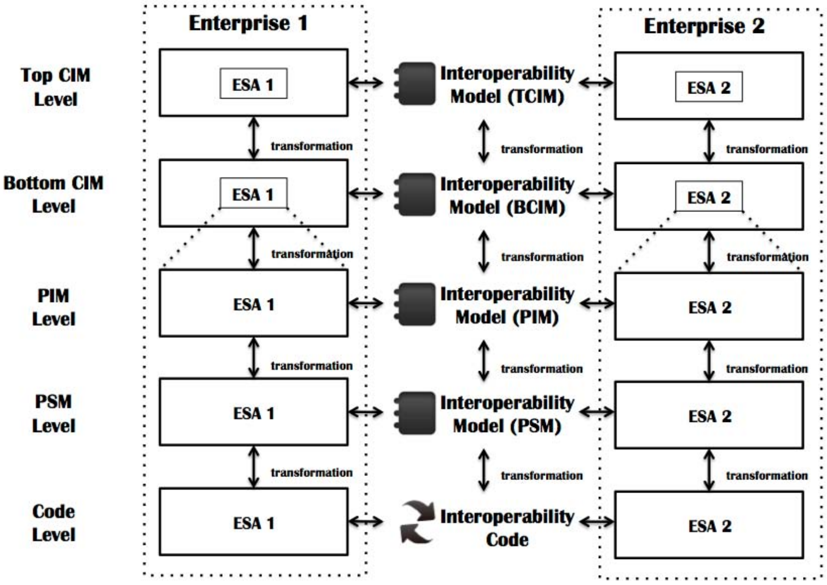


FIGURE I - 5 MODEL DRIVEN INTEROPERABILITY (BOUREY ET AL., 2007)

Although the MDI is based on MDA, the differences between them and the progress of MDI are clear and are not doubtable. They are two similar architectures, but the MDA is more general, the MDI is in interoperability domain. The BVR and the BST are based on different architectures. The BVR is based on MDA, but the engineering approach is very close to MDI. Meanwhile, the experience of the BVR shows that the MDA should be adopted for interoperability modeling. The implementation of the BST proves that: in interoperability domain, the MDI should be used to direct interoperability modeling.

The paragraphs above mentioned model transformation and metamodel mapping several times. Model transformation is a crucial part of MDA and MDI. From 2001 to 2010, the mechanism of model transformation has been improved. For the BVR and the BST, the theories of model transformation are different.

The theory of model transformation of the BVR comes from the 3+1 MDA organization (Bézivin, 2004) and the mechanism of model transformation of ATL[2] (Jouault and Kurtev, 2006). Pr. Jean Bézivin named precisely the organization of the classical four-level architecture of OMG as the 3+1 architecture. At the first level of the 3+1 architecture, the layer 0 is the real system. A model represents this system at layer 1. This model conforms to its metamodel defined at layer 2 and the metamodel itself conforms to the Metametamodel at level 3. The Metametamodel conforms to itself. Based on the 3+1 architecture, the mechanism of model transformation of ATL is defined (Figure I-6). The theory of model transformation covers three levels of the 3+1 architecture except the bottom level (the system). At the bottom of Figure I-6, there are source model and target model. The goal of model transformation is to transform the source model to the target model. Both of them confirms to source metamodel and target metamodel. The mapping rules are defined between the source metamodel and the target metamodel. The source model is transferred to the target model by respecting the mapping rules. Both the source metamodel and the target one confirms to a Metametamodel.

[1] Enterprise Software Application

[2] ATL (Atlas Transformation Language) is a model transformation language and toolkit. In the field of Model-Driven Engineering (MDE), ATL provides ways to produce a set of target models from a set of source models.

The BVR uses mechanism of model transformation of ATL to transfer an inter-model to the target collaborative process model. The research work is based on ATL transformation. The inter-model and the target model describe the same collaborative situation (the system) in different way (confirms to different metamodels). Or they define different structures for the same knowledge. It means that, somehow, the mechanism of ATL drops a hint that both sides of models should organize the same knowledge. If the source and the target models present different information, but have some information in common, how could the mechanism deals with this situation?

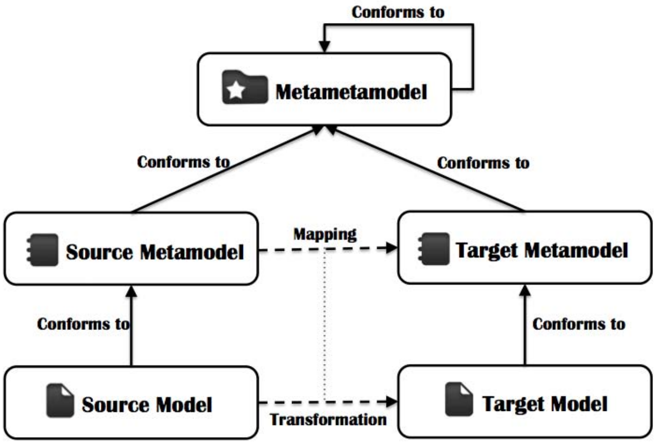


FIGURE I - 6 MODEL TRANSFORMATION OF THE BVR (JOUAULT AND KURTEV, 2006)

For the purpose of answering above question, Pr. Jean-Pierre Bourey re-defined the mechanism of model transformation. The research work of the BST is based on the new mechanism (for this paper, called BST model transformation). Figure I-7 presents the model transformation of the BST. The same part of ATL model transformation and BST model transformation is that, they both have source model (confirms to source metamodel) and target model (confirms to target metamodel). Transformation rules are defined to transfer source model to target model. The different part of ATL model transformation and BST model transformation is that, the BST model transformation considers source model and target model present different knowledge, which may have common part (shared part in Figure I-7). They may define different concepts for source metamodel and target metamodel, but they also may have common concepts. The BST model transformation only defines transformation rules between shared parts (confirm to shared concepts). The shared part of source is transferred into the shared part of target model. The special part of source model is maintained as remained knowledge with database or ontology. The special part of target model is enriched by additional knowledge, which comes form database, ontology, model, text and user etc. The strong points of BST model transformation are: i) reusing the mechanism of ATL model transformation, ii) clearly defining the special situation according to which the source metamodel and target metamodel may have common part and different ones, iii) pointing out that the transformation focuses on shared parts and iv) presenting that the special part of source model should be maintained and the special part of target model should be filled.

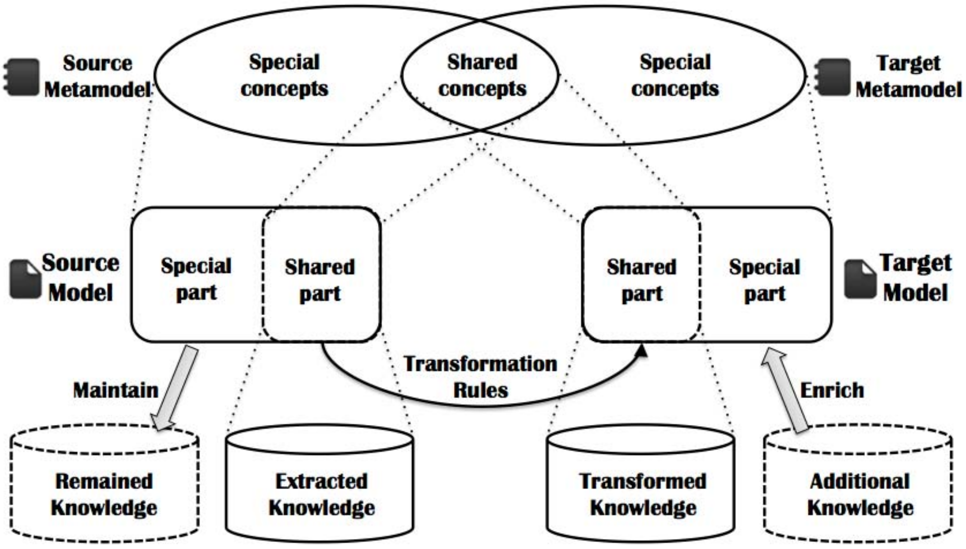


FIGURE I - 7 MODEL TRANSFORMATION OF THE BST (TRUPTIL ET AL., 2010)

Time flies, so the research work does. Dr. Vatcharaphun Rajsiri developed MISE 1.0 business level during 2006 – 2009. Her research work based on MDA (2001) and ATL model transformation (2006). Dr. Sébastien Truptil worked on MISE 1.0 business level of crisis domain during 2007 – 2010. His research work used MDI (2007) and model transformation theory (2010). This is a work just only one year after Dr. Vatcharaphun Rajsiri, but the model-driven architectures and model transformation mechanisms has been changed to adapte the real world.

I.3.2. Comparison of Engineering Methods

The first line of Figure I-8 presents the engineering method of the BVR. The method has three main steps:

- Knowledge gathering: the collaborative network model is defined. The model gathers initial collaborative information: for example, partners, partners' relationship, topology of collaborative network and the common goals of collaboration.
- Process deducing: the collaborative ontology and transformation rules are defined. The collaborative ontology aims to manage all the collaborative concepts and provides transformation rules among these concepts. The collaborative ontology manages concepts such as partner, business service, abstract service, collaborative network etc. All the knowledge of MIT process handbook is extracted form the handbook and filled into the collaborative ontology as instances. With the link between the common goal (in the collaborative network model) and the abstract service (in the collaborative ontology), the instances of business service, which could achieve the abstract service, are selected for the common goal. The transformation rules help to transfer all the knowledge of the collaborative network model and the ontology to a collaborative process model.
- Collaborative process: the metamodel of the collaborative process model, which is defined by Dr. Jihed Touzi (Touzi, 2007), is reused. With the help of transformation rules and the metamodel of collaborative process model, as a result, a BPMN based collaborative process model is deduced. This model has got one mediator pool and several partners' pools. In the mediator pool, there is the collaborative process. In the partners' pools, the business services, which are provided and shared by partner, are presented and invoked by the collaborative process in the mediator pool.

The engineering method of the BST is shown at the bottom lane of Figure I-8. The engineering method also has the same three main steps:

- Knowledge gathering: the service model and the crisis model are defined. The service model gathers information about the services provided by the partners with input and output messages. The crisis model gathers initial crisis situation, for example, accident or crisis, involved population (firemen, police and injured people), place of the accident or crisis. The aim of the crisis model is the same with the collaborative network model of the BVR. Both of them gather the initial information of the collaboration. Before the deduction of the process, the BST asks the user to choose services from the service model for each crisis defined by the crisis model.

- Process deducing: the crisis metamodel is defined. The metamodel owns three parts: studied system, extracted system and collaborative process (the metamodel of the collaborative process of Dr. Jihed Touzi). The part of the studied system aims to manage the knowledge, which is gathered by the service model and crisis model, for example, risk, crisis, people etc. The part of extracted system aims to manage the knowledge, which is automatically deduced from the studied system. The part of collaborative process is the same as the metamodel of collaborative process, which is defined by Dr. Jihed Touzi and reused by BVR. Model transformation rules form studied system and extracted system to collaborative process.
- Collaborative process: the collaborative process model confirms to the collaborative process part of crisis metamodel. The format of the deduced collaborative process has the same as format with the one of BVR.

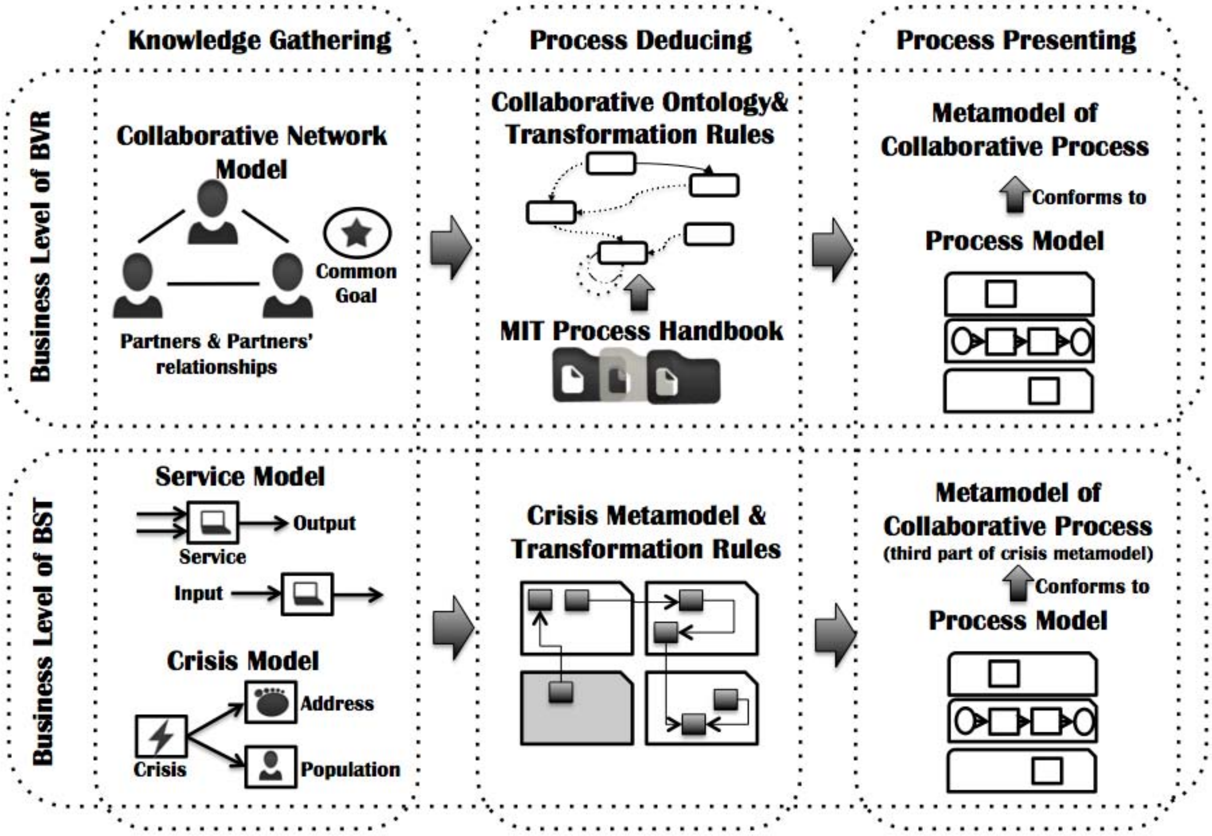

Knowledge Gathering
Process Deducing
Process Presenting
Business Level of BVR
Collaborative Network Model
Common Goal
Partners & Partners' relationships
Collaborative Ontology& Transformation Rules
MIT Process Handbook
Metamodel of Collaborative Process
Conforms to
Process Model
Business Level of BST
Service Model
Output
Service
Input
Crisis Model
Address
Crisis
Population
Crisis Metamodel & Transformation Rules
Metamodel of Collaborative Process
(third part of crisis metamodel)
Conforms to
Process Model

FIGURE I - 8 COMPARISON OF BUSINESS LEVEL ENGINEERING OF BVT AND BST

BVR and BST have the same engineering steps: knowledge gathering, process deducing and process presenting. The differences of each step are listed as followes.

First, differences of knowledge gathering:

BVR uses the collaborative network model to gather information. BST uses the service model and the crisis model to gather knowledge. After the explanations of the content of the collaborative network model and the crisis model above, it is obvious that both of them gather initial collaborative information. But the crisis model is changed to fit crisis domain. The two models serve the same function, which is gathering initial collaborative knowledge. BST uses also the service model to gather shared business services from partners. Here, a question appears, why does the BVR not collect information about services? If the BVR does collect information of services, then which step? And how? In the second point, we could find the answer.

Second, differences of process deducing:

BVR defines the collaborative ontology with transformation rules and inserts MIT process handbook into the ontology. BST defines the crisis metamodel and transformation rules. The main difference is that: the collaborative ontology defines concepts, relationship among concepts and has instances of concepts and relationships, but the crisis metamodel only defines concepts and relationships without instances. This difference could answer the question proposed above. BVR does not define a service model, but the instances in collaborative ontology provide business services such as the service model of BST. Another difference is that the collaborative ontology defines concepts of general collaboration, but the crisis metamodel defines not only concepts of general collaboration but also crisis concepts.

Third, differences of collaborative process:

BVR and BST use the same metamodel of collaborative process, which is defined by Dr. Jihed Touzi. The difference is that, BVR re-uses the metamodel as individual metamodel, but BST re-uses the metamodel as the third part of the crisis metamodel. The transformation rules of BVR are from the collaborative ontology to the metamodel of collaborative process model. But the transformation rules of BST are inside the crisis metamodel (from studied system and extracted system to collaborative process).

I.4. Assumptions and Limits on MISE 1.0

Assumptions and limits were identified after the research and development of the MISE 1.0 project. Assumptions and limits in MISE 1.0 cover CIM, PIM and general approach. In section I.4.1, assumptions and limits on abstract level are presented. Assumptions and limits on concrete level are introduced in section I.4.2. Section I.4.3 presents assumptions and limits on general approach.

I.4.1. Assumptions and Limits on CIM

In this section, assumptions and limits in business level (CIM) are presented. The assumptions and limits are separated into two parts.

First, limitations of collaborative knowledge are discussed; CIM collaborative process model is an operational process model.

Collaborative process is introduced in section 3.2 of chapter 4 of Dr. Touzi PhD report (Touzi, 2007). According to, the model of a collaborative process is BPMN-oriented and based on the SOA. He considered the activities provided by partners as their internal process. But, he also stated, based on the point of view of the enterprise, that a collaborative activity can be seen as an internal process and presents an interface dedicated to the collaboration.

Collaborative process model from MISE 1.0 business layer defines a single collaborative process, which covers strategy, operation and support. However, referring to the ISO 9000:2000 recommendations, the decision level and support level should be considered. In order to make collaborative process model presents the information more clearly, the collaborative process model should be presented by different views (for example, decision view, support view and operational view).

Second, the MIT Process Handbook is used as a process knowledge repository to catch definitions of services at different levels of granularity.

Usage of MIT Process Handbook is described in the section 2.3 of chapter 2 in (Rajsiri, 2010). The MIT Process Handbook is a repository of business processes. It contains about 5000 business processes including use cases, alternative business models, and so on. These instances of knowledge are about business processes that are used to constitute the knowledge base in MISE 1.0 business level. We used the knowledge provided in the MIT Process Handbook and we also adopted its modeling mechanisms to complete our collaborative process design. Based on these studies, we summarized the principal elements for defining a collaborative process as being: partner's service, resource, flow of resources between services (dependency), and MIS service (coordination service of the MIT Process Handbook).

Once the user has built a collaborative network model, business services are selected from a special version of the MIT repositories available as ontology. CIM process could choose useful services to build the collaborative process model. This approach, based on reusability, leads to two problems. Firstly, collaborative process model is general. It is not focused on specific domain or specialized field. Secondly, limits of MIT Process Handbook knowledge became limits of MISE 1.0 on process modeling capabilities.

I.4.2. Assumptions and Limits on PIM

During the CIM to PSM transformations, (Touzi, 2007) makes the following main assumption: linking business activities (from the BPMN model) and available technical services is always possible and computerizable. Unfortunately, this assumption is viable only if each activity has an existing technical service (and they have the same name). The potential semantic problems are avoided. Information brought from PSM is supposed to be correct and enough to match web services. But for the research work of Dr. Sébastien Truptil in IsyCri project, he did define the business activities based on the technical service.

Furthermore, (Touzi, 2007) regards business activities and technical services as two resources with the same granularity and only makes one to one connections. In concrete cases, the one to one connections are not enough to complete BPMN to SOA transformation (one business activity may be implemented by several technical services; several business activities may be completed by one technical services). The many to many relations must be considered.

In order to improve agility and maturity of the system, (Truptil et al., 2008) adds a technical ontology containing functional (address, messages…) and non-functional information (requirements…) about existing services.

Unfortunately, this service ontology is not based on a standard despite that a lot of specification are available and are closed to requirement. It does not promote interoperability, which is one of the main goals of this project. Besides, the semantic matchmaking is manually performed and only existing technical services are usable.

Transformation from business activities to services is one to one in PIM. Then, during PSM construction, technical information from technical ontology is added to model in order to perform the BPEL building.

In the technical level, (Touzi, 2007) treat web services without taking heterogeneity of exchanged messages into account. For two consecutive services, he considers the output message of the first service is similar to the input message expected by the second service. When (Truptil et al., 2008) improves the MIS to use it from top to bottom, Dr. Truptil adds exchanged messages in the mediation services. For now, exchanged messages are added into correspondence services manually. No semantic information is available on data models, which make data transformation impossible to automate. If any new sequence of service is necessary, mediator services information has to be added manually.

I.4.3. Assumptions and Limits on MISE 1.0 Model-Driven Approach

In this section, assumptions and limits on MISE 1.0 model-driven approach are presented. The assumptions and limits are separated into two parts.

Firstly, knowledge is gathered in each step.

In Figure I-2, knowledge has been gathered in each step. First, in business branch, in order to create collaborative process model, business knowledge concerning the network has been gathered and some information from the MIT Process Handbook has been added. This step has been described in (Touzi, 2007). Second, in logic branch, technical information about service address, service descriptions, exchanged message and attributes has been gathered. Third, web services are mapped one to one. But web services should be mapped “many” to “many” as mentioned in section I.4.2. So we consider semantic information should be gathered in technical branch. In Figure I-2, collaboration knowledge is gathered in each step (from business to logic, from logic to technologic, from technologic to target system). Before the development of MISE 1.0, we did not consider the situation according to which the collaborative knowledge would be gathered in every step. So a collaborative situation framework, which manages the model-driven approach and defines the collaborative knowledge in each step, is required.

Secondly, MISE 1.0 model driven approach is a linear engineering process.

In (Benaben et al., 2010) the whole MISE 1.0 model driven approach has been introduced. In Wenxin Mu's (this is me ^0^) master report, the MISE 1.0 model driven approach has been introduced through an example. From these two papers, we could summarize MISE 1.0 model driven approach as a linear engineering process. As the linear engineering process, there are two related problems. First, if one part of one model in the model transformation chain is changed, the whole remaining part of the chain must be driven again. This leads that, if one modeling tool does not work, the whole chain does not work. To solve the problem, we decided to overcome the limitations induced by the CIM and PIM segregations.

II. MISE 2.0 Proposal

In order to solve or avoid all the assumptions and limits mentioned in above section. MISE 2.0 (for Mediation Information System Engineering, second iteration) project has been launched in 2009. The MISE 2.0 engineering approach is based on BPM and MDA. Running a BPM approach on a specific organization consists in gathering structural, informational, and functional resource knowledge in order to design cartography of processes covering the behavior of the modeled organization. A BPM approach is classically dedicated to three types of goals: using the processes cartography to (i) certify the modeled organization, (ii) optimize the processes of the modeled organization, and (iii) define the requirements for the IS design of the modeled organization. In this thesis, we will focus on the third objective. This BPM approach is not dedicated to any single organization, neither to a set of organizations, but to the target collaborative situation between a set of organizations. This is the MISE 2.0 project, which can be considered according to the following layers:

- Knowledge gathering (situation layer): collect information concerning the collaborative situation.
- Processes cartography design (solution layer): design the processes according to the knowledge gathered.
- MIS deployment (implementation layer): implement an IT structure able to run the processes cartography.

The transitions between these layers (from *situation* to *solution* layers and from *solution* to *implementation* layers) are the hard-points of this approach. Indeed, driving such a BPM approach requires designing relevant processes cartography from the dedicated situation (first gap). However, this is usually a manual activity, which requires a good knowledge of the modeled situation as well as a large amount of work to draw the processes and their links. Besides, once that cartography designed, the IS design requires to bridge the semantic gap (second gap) between the *business* activities (modeled in the processes cartography) and the *technical* services (concerned by the IS deployment). The first gap is managed at the *abstract level* of MISE 2.0 while the second one is managed at the *concrete level* of MISE 2.0. Figure I-9 illustrates the overall BPM approach considered in the context of MISE 2.0:

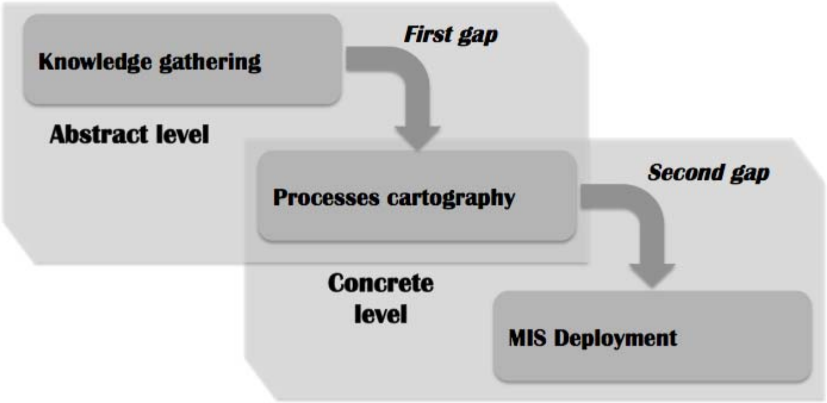


FIGURE I - 9 OVERVIEW OF MISE 2.0 PROJECT

Assumptions and limits are introduced in section I.4. In order to solve all these problems, MISE 2.0 project has been started in September 2009; two PhD subjects have been proposed to Wenxin Mu and Nicolas Boissel-Dallier. The first PhD subject mainly aims to solve assumptions and limits on *abstract level* and some on general approach. The second PhD subject is mainly to solve assumptions and limits on *concrete level* and some on general approach. In section II.1, the objective of abstract level of MISE 2.0 is addressed. In order to obtain the objective, the problems, which have to be solved

II.1. The Abstract Level of MISE 2.0

Regarding the abstract level (or business level), the BVR and the BST have been introduced and compared in section I.3. Both of them followed three steps: knowledge gathering, process deducing and process presenting. Their main objective is to deduce the collaborative process, which operates the business collaboration of partners. For MISE 2.0 abstract level design, the main objective is to build the collaborative process cartography. But what is the collaborative process cartography? And why? As mentioned in section I.4, the collaborative process of the BVR and the BST is a “mixed” process, which covers the information of strategy, operation and support. This kind of collaborative process is very difficult for user to understand and execute. The collaborative process runs among different levels of users (managers, workers from operating unit or warehouse and so on). The users own different knowledge, which makes them to understand part of the collaborative process. Because the users have different functional distribution in the enterprise, they concern only part of the collaborative process. It is better to build several small collaborative processes, which present different part of the “mixed” collaborative process. The several small collaborative processes should be managed and presented by a main process, which is the collaborative process cartography. The collaborative process cartography is to break the “mixed” process into small processes and classify these small processes as strategy, operation or support. The collaborative process cartography presents the process as one main process (with the information of classification) and several sub processes.

In order to build the collaborative process cartography, the collaborative knowledge of the process cartography should be gathered and transferred. Our principles are to i) gather the essential and minimum initial collaborative knowledge (e.g. partners, collaborative objective and shared functions) in the mode of model, ii) deduce the missing knowledge with the help of ontology/metamodel and transformation rules and iii) complete the collaborative process cartography with the deduced knowledge and necessary algorithms (in the case of fixing the small gap of model transformation). As shown in Figure I-10, based on the above principles, in a collaborative situation, the partners come with their shared functions and their private objectives to achieve the collaborative objectives of the collaboration. The shared functions and the collaborative objectives could be seen as the initial collaborative knowledge. The goal of MISE 2.0 abstract level (the goal of the research work in this manuscript) is to select the shared functions and build the collaborative process cartography, which is made up by a main process and several sub processes of strategy, operation and support. The collaborative process i) provides the order of shared businesses functions to follow, ii) presents divided strategy, operation and support processes for different levels of users, iii) obtains the collaborative objectives and iv) potentially achieve the individual objectives of partners.

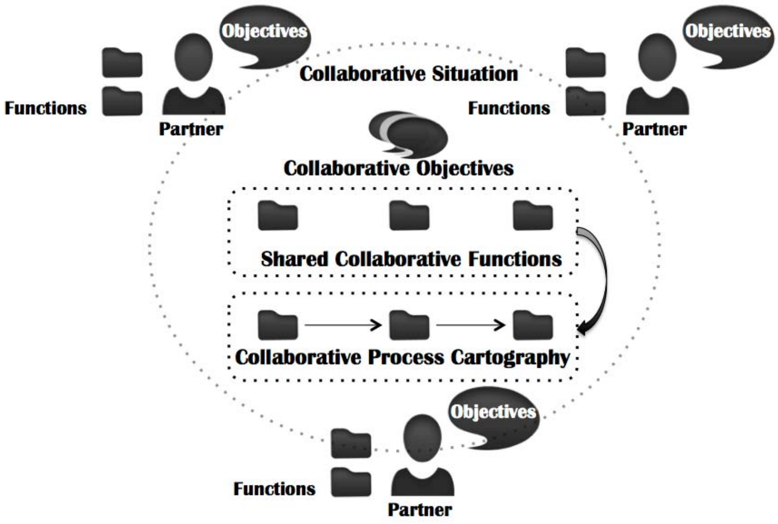


FIGURE I - 10 THE TARGET OF MISE 2.0 ABSTRACT LEVEL

As discussed above, to build the collaborative process cartography, we have to gather collaborative knowledge, transfer missing knowledge (deduce the missing knowledge of process) and present the collaborative process cartography. For each step, their some problems, which has to be solved. These problems are listed as followed and also presented in Figure I-11.

- Question 1: in the phase of knowledge gathering, what kinds of collaborative knowledge should be gathered? And how?
- Question 2: If the functions of partners are gathered, how to select the correct functions to achieve the collaborative objective?
- Question 3: in the phase of process deducing, which model transformation mechanism should be chosen? Is it the transformation mechanism of BVR? Or is it the transformation mechanism of BST?
- Question 4: if we use the transformation mechanism of BVR, how could we solve the limits of MIT process handbook? If we use the transformation mechanism of BST, which kind of model should be added?
- Question 5: in the phase of process cartography, how could we change BPMN based collaborative process model to provide the full process cartography? How could we classify collaborative process as strategy process, operation process and support process?
- Question 6: In order to build the processes, how could the orders or the sequences among the business functions be deduced?

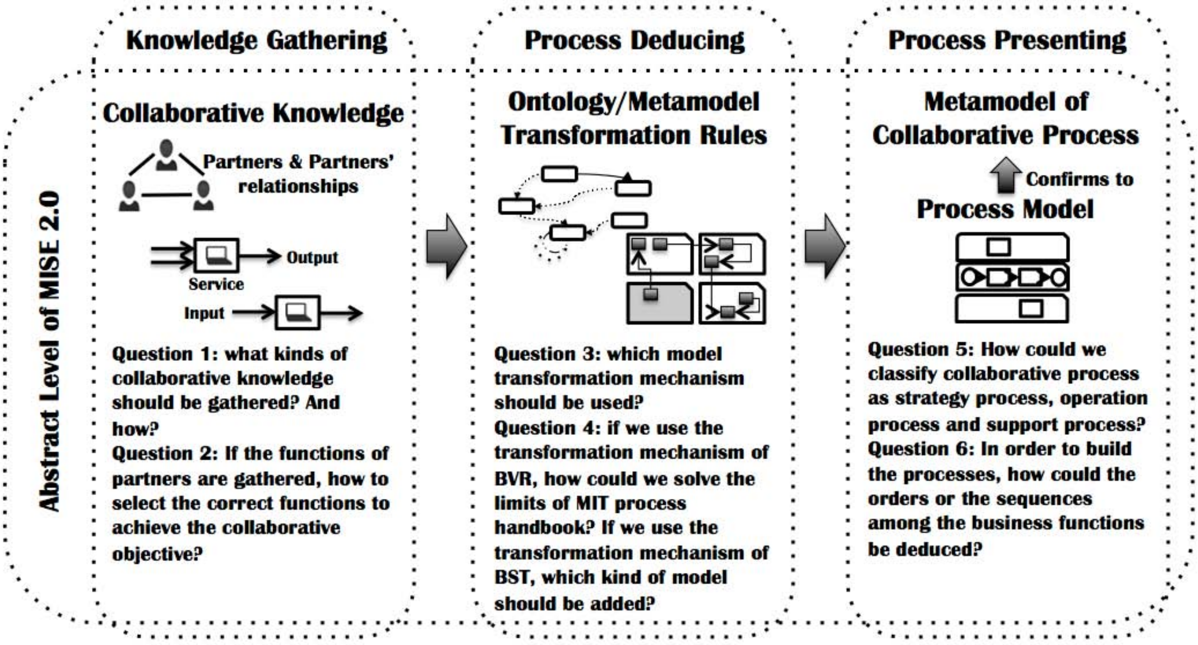

Abstract Level of MISE 2.0
Knowledge Gathering
Collaborative Knowledge
Partners & Partners' relationships
Service
Output
Input
Question 1: what kinds of collaborative knowledge should be gathered? And how?
Question 2: If the functions of partners are gathered, how to select the correct functions to achieve the collaborative objective?
Process Deducing
Ontology/Metamodel Transformation Rules
Question 3: which model transformation mechanism should be used?
Question 4: if we use the transformation mechanism of BVR, how could we solve the limits of MIT process handbook? If we use the transformation mechanism of BST, which kind of model should be added?
Process Presenting
Metamodel of Collaborative Process
Confirms to
Process Model
Question 5: How could we classify collaborative process as strategy process, operation process and support process?
Question 6: In order to build the processes, how could the orders or the sequences among the business functions be deduced?

FIGURE I - 11 PROBLEMS TO SOLVE IN MISE 2.0 ABSTRACT LEVEL

In order to clearly understand these problems, the next section provides a small state of art for Business Process Management (BPM). Three BPM research works are explained. These works could provide some clues for solutions of problems or help reader to understand the listed problems.

II.2. Related Works

Some existing research fields can be related to MISE 2.0. Many research approaches in System Engineering, Decision Science, etc., can be applied to implementation of dynamic management of business process in the ubiquitous environment. In this section we only review the research fields that are closely related to this thesis.

The approach in this thesis is greatly motivated by some advancement in Business Process Management (BPM). A business process comprises a *"series or network of value-added activities, performed by their relevant roles or collaborators, to purposefully achieve the common business goal"* (Ko, 2009). BPM life cycle can be grouped into six categories: design, modeling, execution, monitoring, optimization and re-engineering. But Van Der Aalst considers that business process management covers the whole life cycle of business processes, including process design, simulation, enactment, monitoring and control, diagnosis, etc. (Van Der Aalst and Ter Hofstede, 2000; Ter Hofstede et al., 2003) They describe BPM life cycle by different words, but which present the same meaning. Application of formal methods in business process management systems is critical to ensure correctness properties of business process definition and furthermore enables the potential analysis. (Van der Aalst and Van Hee, 2004)

In the world of BPM, many different process modeling notations and tools have been proposed (e.g. IDEF Suite, BPMN, ARIS, UML, Structured Analysis and Design Technique, Petri Nets, Object Oriented Modeling, CIMOSA, IEM approach). Their functionalities and characteristics vary and can lead to misunderstanding and failure. Furthermore, executable languages used to implement the models (e.g. BPEL or classical programming languages) are also diverse. These identified issues are similar to those identified in the Model-Driven Software Development (MDSD) concept (Stahl and Völter, 2006), which is a specialization of MDA. (Patig et al., 2010) well summarized the software and tools used to describe business process in sample companies (Table I-1). Patig, Casanova-Brito and Vögeli have conducted a worldwide survey of major public companies to elicit the requirements, which are grounded in the nature of processes and the usage of software. The analysis of 127 responses indicates that human-oriented process modeling languages and BPM tools as well as BPM tools with software integration capabilities are most urgently required. Obviously, many companies combine text (55.9%)

and some modeling languages (55.9%), but also tables are widespread (31.5%). Among the languages, BPMN dominates, followed by the Unified Modeling Language (UML) and Event-driven Process Chains (EPC). The worldwide MDA still has a long way to go.

TABLE I - 1 CURRENT DOCUMENTATION OF PROCESSES (N=127, NA=3) (PATIG ET AL., 2010)

Answers		Count C_i	Percentage (C_i/ ΣC_i)	Percentage responses (C_i/N)
As text		71	36.2%	55.9%
As tables		40	20.4%	31.5%
As flow charts		2	1.0%	1.6%
With languages	BPMN	27	13.8%	21.3%
	UML	19	9.7%	15.0%
	EPC	16	8.2%	12.6%
	BPEL	5	2.6%	3.9%
	IDEF	4	2.0%	3.1%
Other		12	6.1%	9.4%
Total (ΣC_i)		196	100.0%	154.3%

There are numerous valuable research works in the business process management field, beside these well known modeling languages and modeling architecture, for example situation calculus (Li and Iijima, 2007a), business process abstraction (Smirnov et al., 2012), business process correctness (Lohmann and Wolf, 2010), self-adjusting process (Dorn et al., 2010), re-structure of process model (Polyvyanyy et al., 2010), etc. Their works focus on the detail research problem of business process management. All these works could give some clues to solve problems encountered in the MISE 2.0 business level design.

In this section, we choose several special and interesting process management works as related work introduction. First, in section II.2.1, ARIS method extension for Business-Driven SOA is explained. The method of ARIS extension defines SOA metamodel, which covers CIM, PIM and PSM levels of MDA. The concepts of metamodel are similar with the concepts of our MISE 1.0 and MISE 2.0 engineering approach. In section II.2.2, situation calculus is presented. This method mainly presents the selection of business services for collaborative goal, which is a main problem to solve in this thesis. Section II.2.3 explains business process abstract. The research work solves two problems. First, what is a main business process? Second, how to extract a high-level business process (main business process) from low-level business process (detailed business process)? It relates to business process cartography in this thesis.

II.2.1. ARIS Method Extension

According to the section 2 of (Stein et al., 2008), there is no complete SOA modeling method which is integrated with enterprise architecture frameworks and business process management methods. It is the authors' research goal to fill this gap in case of the ARIS modeling method. ARIS already covers all parts of a modeling method, but it misses specific modeling language concepts for SOA. This section describes how Stein extended the ARIS modeling language to allow service-oriented enterprise modeling. They refer to this extension as the "ARIS extension". The "ARIS extension" provides valuable ideas to answer Question 1 in previous section. Designing the ARIS extension was limited by the following constraints:

- The ARIS extension must enable reuse of as many existing ARIS models as possible to preserve users' investments.
- The proposed ARIS extension must be fully integrated with the existing ARIS modeling language.
- Relying on pilot users is not possible, because users expect from the authors to propose a solution rather than standardizing the users' competitive advantage.
- The skill sets of the typical ARIS user must be taken into account. Internal user analysis shows that most ARIS users have neither natural science nor mathematical background but instead studied business administration and related science subjects.
- There are many modeling languages available like UML, which must be integrated if possible.

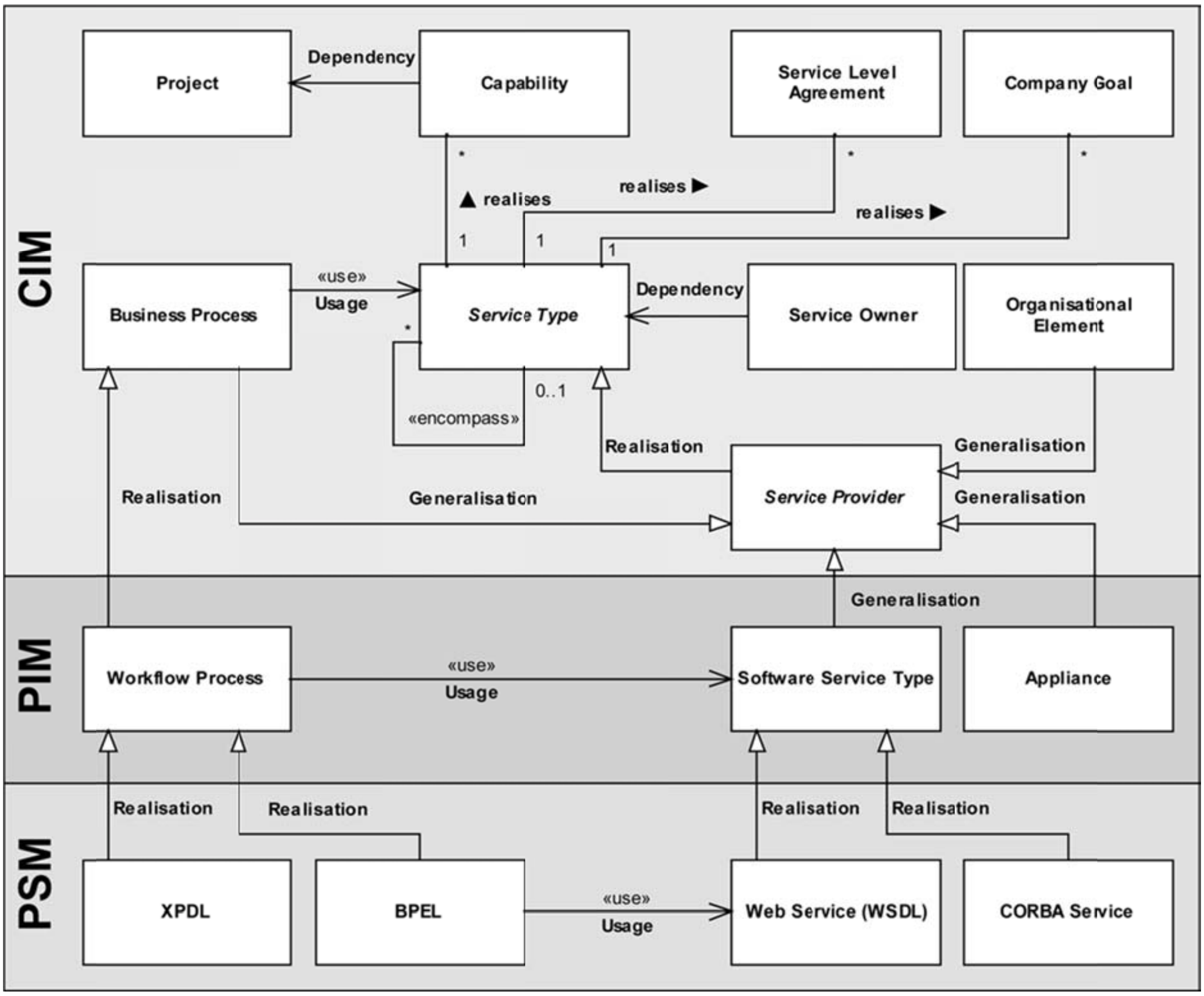


FIGURE I - 12 SOA METAMODEL (STEIN ET AL., 2008)

Figure I-12 shows the core part of the SOA metamodel as a UML class diagram. The SOA metamodel is structured into 3 levels, namely: CIM, PIM and PSM. Those levels are taken form MDA. Figure I-12 shows that some kinds of services are the core element on each level. On the CIM level, there is a service type, which can be realized by different service providers. Software service types are a possible way to provide a service type. They are located on the PIM level. Each software service type can be implemented with different technologies, which reside on the PSM level at the bottom of Figure I-13. According to (Stein et al., 2008), there are three main elements of the SOA metamodel. They are summarized as follows:

- Capability: entities like organizations, IT systems, and business process have a set of capabilities. Their capabilities enable them *"to solve or support a solution for the problems they face in the course of their business"* (MacKenzie and others, 2006). A capability is a descriptive element documenting a system. In the specific case of the SOA metamodel, capabilities are used to describe service types and software service type. *"Capabilities can be used to describe functional and non-functional properties."* (Stein et al., 2008) A capability is always global and can be reused to describe different services. Therefore, a capability must clearly define the context it can be used in. Reusing capabilities among services allows identifying services with similar capabilities.

- Service Type: the service concept presented in (MacKenzie et al., 2006) is so generic that is does not just cover software or web services, but instead any *"mechanism enabling access to one or more capabilities"*. The service type is a mechanism enabling access to a set of capabilities. Figure I-12 shows that different service providers can realize the service type: software service type, organization, appliance and business process. Each service type has a service owner. The service owner is responsible for maintaining the service type's description and advertising the service offering. A service type can be decomposed into other service types.
- Software Service Type: are platform independent but software-based services. They can be implemented using different technologies. A software service type is used in a workflow or integration process. Each software service type can be implemented with different technologies, which reside on the PSM level at the bottom of Figure I-12. The software service type is one of the types of Service Type (also maybe organization, appliance and business process).

The ARIS extension has been included into ARIS 7.1 (AG, 2009). For ARIS 7.02 (Brabänder and Davis, 2007), there are no modeling concepts available in ARIS to represent a service on a computation independent level, so every service is expected to be implemented by software. Figure I-13 shows the current and newly introduced objects of ARIS 7.1 mapped to the three MDA levels. It can be seen that the PIM and PSM levels are covered in ARIS, but that a computation independent service description is missing. The information system function object is at the border between CIM and PIM. The application system type object is independent of any specific technology, but it already mandates an IT implementation. It is there fore on the platform independent model level. The WSDL[1] specific models are technology dependent and are therefore located on the platform specific model level.

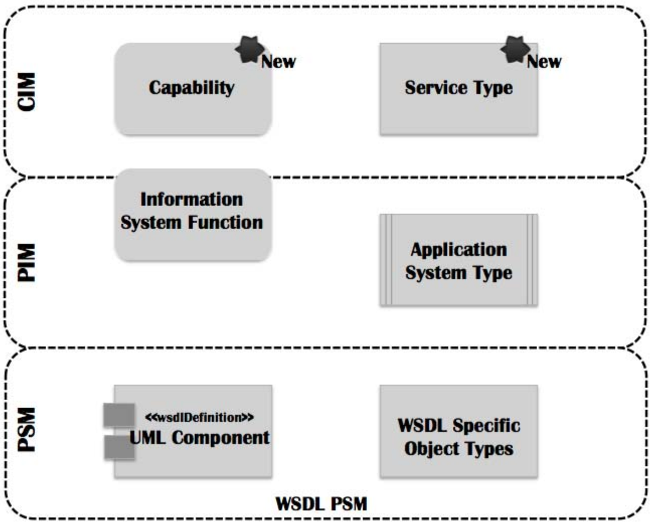

CIM
Capability
New
Service Type
New
PIM
Information System Function
Application System Type
PSM
<<wsdlDefinition>> UML Component
WSDL Specific Object Types
WSDL PSM

FIGURE I - 13 ARIS OBJECTS GROUPED ACCORDING TO MDA LEVELS (STEIN ET AL., 2008)

II.2.2. Situation Calculus

The situation calculus can be applied to formally specify and analyze business processes by considering the intuitive mapping from an activity in a process to an action in the situation calculus domain. Li and Iijima did the research in verifying business processes by employing the situation calculus (Li and Iijima, 2007b, 2007c). The situation calculus can be briefly summarized as two tasks i) automatically selecting business activities (functions or services) to achieve the goal (business goals, e.g. selling product or producing product) and ii) verifying that if the selected functions successfully and correctly complete the goal. The first task of situation calculus may be a solution for Question 2 in previous section. It provides us some information for selecting

[1] Web Service Description Language

shared business functions to achieve the collaborative objective. In the research work of Li and Iijima, they call the first task as Service Automatic Composition and the second on as Service Model Verification.

Service Automatic Composition is to compose services automatically from the existing isolated services inside or outside an enterprise. In order to enable this automatic composition, formal descriptions of services are prerequisite. A formal service description refers to specifying services by employing formal methods, usually mathematical logic. With such a formal specification, services are described precisely and unambiguously. Logical reasoning can be performed, which enables the automatic composition.

Service Model Verification ensures that a service model should satisfy the service specification. That is, given the initial situation of a service and the formal service specification, see if the goal situation can be satisfied. In other words, model verification is to check if the successor states resulting from executing a service is just what is desired or expected for the service. The model verification can be applied at two levels: one is the lower level inside a service, i.e., to check if the constituent activities can work to reach the goal of a single service; the other is the upper level across services, i.e., to chick if the constituent services of a composite service can collaborate to reach the overall goal.

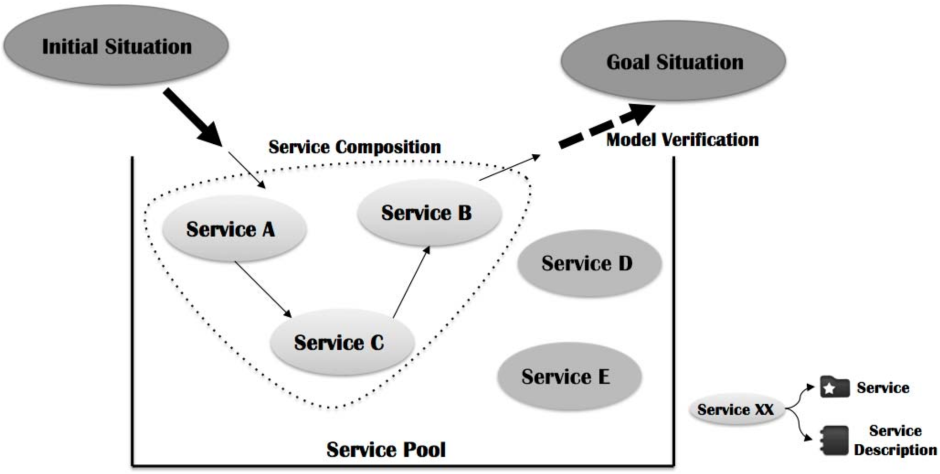


FIGURE I - 14 AUTOMATIC COMPOSITION AND MODEL VERIFICATION (LI AND IIJIMA, 2007B)

Figure I-14 illustrate the reasoning work. Initially, there is a service pool that collects the existing services and meanwhile each service in the pool has a formal service specification. For the case of automatic composition, the initial situation and goal situation of a desired service are given. The initial situation refers to the initial condition or states when the desired service is to be enacted; the goal situation refers to the successor state when the desired service is completed. From these three types of information (service specification in the service pool, the initial situation and the goal situation), the service system will provide the service composition. In Figure I-14, the automatically composed service model includes Service A, B and C. The execution sequence is, firstly A, then B and lastly C. Similarly, Figure I-14 illustrates the reasoning mechanism in model verification. The initial situation and the integrative service model (a business process) will be given, based on which the successor state will be checked if it is reachable and if it satisfies the given goal situation.

In order to automatically build service composition (process), all the business services of enterprises must be gathering into service pool and described in Service Description (contain information of capability of service, previous service, next service and so on) with the format of XML Process Definition Language (XPDL) (Van Der Aalst, 2003). With the Service Descriptions, based on the initial situation, the services are composited as a process. In order to check whether the selected services could reach the target situation, the verification of the composition is performed. For the MISE 2.0 abstract level, in order to select the correct shared business functions, each function has to define some properties or profiles to support the selection.

II.2.3. Business Process Abstraction

As, traditionally, for each modeling goal a specific process model was designed, companies maintain large process model repositories consisting of hundreds or even thousands of models. The stored models have complex interrelations: they may overlap, describe processes that subsume each other, or describe one process from different perspectives. Models that formalize the same business process typically vary in the level of abstraction, so that along with detailed models also more coarse-grained models are maintained. As such models are stored independently, it is hard to keep them in sync. Each change of the process needs to be applied to all its models, which incurs a significant overhead. To solve this problem, business process model abstraction (BPMA) has been proposed (Bobrik et al., 2007; Polyvyanyy et al., 2008). The general idea is to develop a detailed process model and to provide view on it using abstraction mechanisms. This research work solves the problem, which is similar with Question 6 in section II.1.

Against the background, business process model abstraction has emerged as a technique that allows one to inspect a business process model at different abstraction levels. Informally, business process model abstraction can be seen as an operation on a business process model that preserves process properties that are essential for a particular purpose, while it leaves out insignificant details. To further pin down the notion of business process model abstraction, (Smirnov et al., 2012) provides two perspectives on the relations between models capturing one business process with different precision. First, they postulate a finite non-empty set of process models and an infinite non-empty set of process instances. A mapping sets up a correspondence between a process model and the set of instances it describes. Second, they allocate the artifacts relevant for BPMA to different levels of the Meta Object Facility (MOF). They refer to MOF as a standard for model-driven engineering, which organizes (meta-) modeling artifacts into 4 levels. Figure I-15 relates the BPMA artifacts constellation according to MOF. A set of process instances *inst(m)* related to process model *m* is allocated to level *M0*. The business process model m is put on level *M1*, as it describes/models a set of instances *inst(m)*. Process model m conforms to the modeling notation in which it is described – metamodel *n*. The process model $m_a \in abstr(m)$ is an abstraction of m and also belongs to level *M1*. Model m_a describes the set of instances *inst(m)*. Notice that it requires models *m* and m_a to conform to one metamodel.

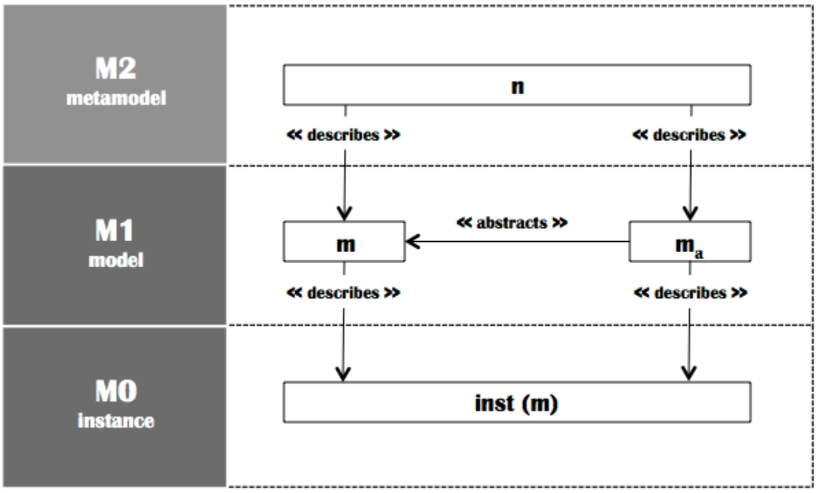


FIGURE I - 15 ALLOCATION OF BUSINESS PROCESS MODEL ABSTRACTION ON MOF LEVELS (SMIRNOV ET AL., 2012)

III. MISE 2.0 Business Level Design

In a collaborative situation, all the partners come with collaborative objectives to achieve and business services to share. They expect to combine their own business services with suitable ones from other partners to work towards their common objectives. In addition, the collaborative business process is a combination of business functions, which is inter-linked and filled with sequences and orders. With such needs, objective-oriented business service selection and collaborative business process creation are absolute essentials in a collaboration world. Considering self-updating and re-building of a collaborative business process, we should design an automatic way to deal with service selection and process creation in a design level.

As introduced in Section I.2, Dr. Vatcharaphun Rajsiri has created a knowledge-based system for a collaborative process specification. This system automatically deduces a BPMN based collaborative process model with the help of collaborative objective model and MIT process handbook. But this system has weaknesses. First, the system only collects the main goal of the whole collaborative network leading to lack of partners' objectives and sub-network information. Designing a model, which models all the information above is necessary. Second, the deduced BPMN collaborative process is a "mixed" collaborative process, which contains the knowledge of strategy, operation and support process. If in a complex collaborative situation, people may come from different departments and units. An operational collaborative business process could not satisfy the partners. According to (ISO 9000, 2005; ISO 9000 X50-130, 2005), the business process covers strategy, operation and support levels. We come to the conclusion that target collaborative process should contain strategy, operation and support levels.

A model-driven and ontology based methodology, which takes collaborative objectives and business services as an input and deduces collaborative business process as an output as automatically as possible, seems to be a good solution in this situation. The global structure of the methodology is shown in Figure I-16 (which is called the UJ picture). In the global structure, there are two parts: U (in dash line) and J (in dash-dot line).

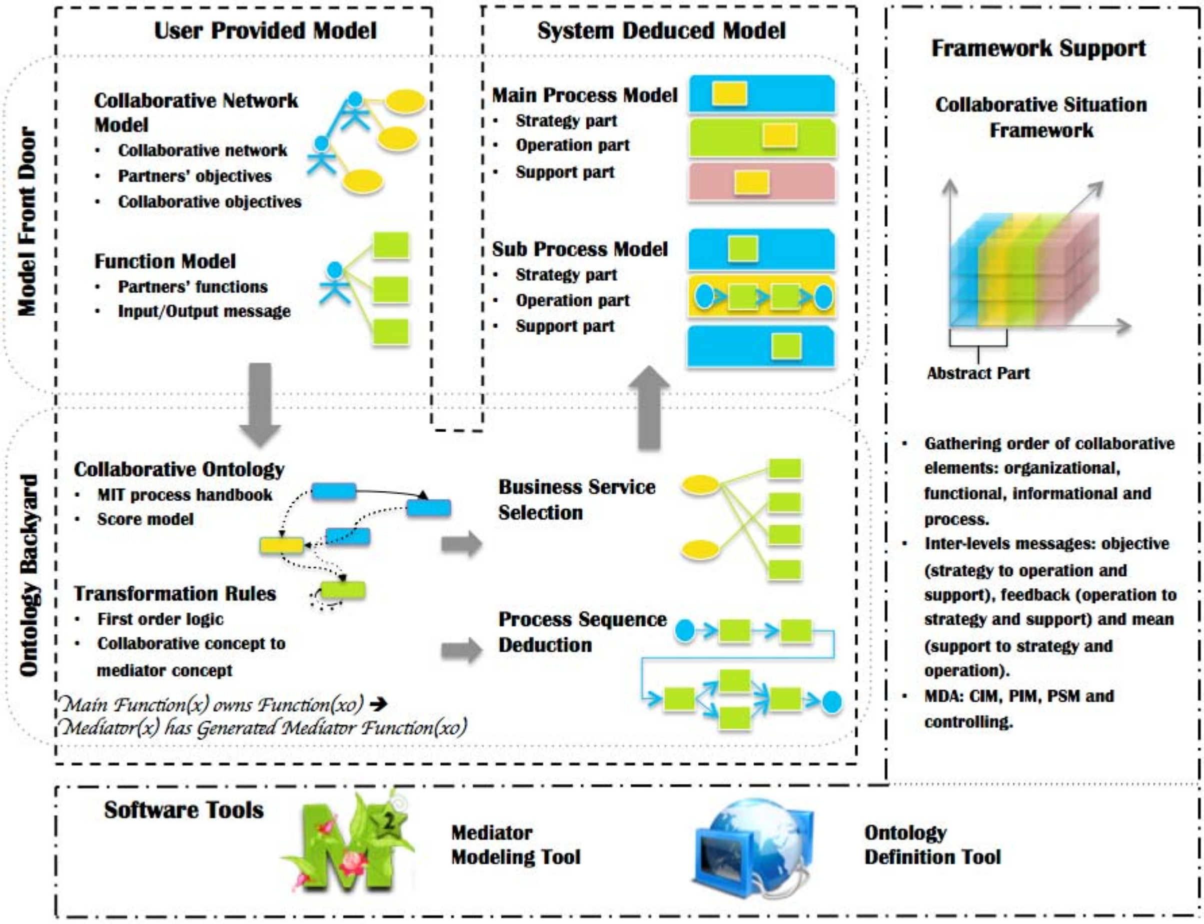

User Provided Model
System Deduced Model
Framework Support
Model Front Door
Collaborative Network Model
• Collaborative network
• Partners' objectives
• Collaborative objectives
Function Model
• Partners' functions
• Input/Output message
Main Process Model
• Strategy part
• Operation part
• Support part
Sub Process Model
• Strategy part
• Operation part
• Support part
Collaborative Situation Framework
Abstract Part
Ontology Backyard
Collaborative Ontology
• MIT process handbook
• Score model
Transformation Rules
• First order logic
• Collaborative concept to mediator concept
Main Function(x) owns Function(xo) →
Mediator(x) has Generated Mediator Function(xo)
Business Service Selection
Process Sequence Deduction
• Gathering order of collaborative elements: organizational, functional, informational and process.
• Inter-levels messages: objective (strategy to operation and support), feedback (operation to strategy and support) and mean (support to strategy and operation).
• MDA: CIM, PIM, PSM and controlling.
Software Tools
Mediator Modeling Tool
Ontology Definition Tool

FIGURE I - 16 GLOBAL PICTURE OF THE DESIGN OF MISE 2.0 BUSINESS LEVEL (UJ PICTURE)

This global structure answers all the questions proposed in Section II.1 Figure I-11. The global structure is explained answering all the questions.

U presents the modules of the engineering approach of MISE 2.0 business level.

Question 1, in the phase of knowledge gathering, what kinds of collaborative knowledge should be gathered? And how?

- User provided model, for the input, we define the collaborative network model (chapter III, section II) and the function model (chapter III, section III) to collect basic collaborative knowledge from partners. The collaborative network model collects collaborative network/sub-network, collaborative objectives, partners' objectives and partners' relationships. The function model presents partners' business services and input/output messages exchanged between them.

Question 3, which model transformation mechanism should be used? Is it the transformation mechanism of the BVR? Or is it the transformation mechanism of the BST?

- Collaborative ontology (chapter IV, section III): the collaborative ontology and transformation rules are defined. The transformation rules could deduce the most important part of target collaborative process. As for the remaining part, we defined several algorithms for a business service selection and process sequence deduction.

Question 2, if the functions of partners are gathered, how to select the correct functions to achieve the collaborative objective?

- Business service selection (chapter IV, section IV.1): with the help of collaborative ontology, three algorithms are defined to automatically select business functions for collaborative objectives.

Question 6, in order to build the processes, how could be the orders or the sequences among the business functions deduced?

- Process sequence deduction (chapter IV, section IV.2): with the help of collaborative ontology, the method of the deduction of sequences among business functions is defined.

Question 5, how could we change BPMN based collaborative process model to adapt process cartography? How could we classify collaborative process as strategy process, operation process and support process?

- System deduced model (chapter III, section IV): for the output, the BPMN based collaborative process cartography is deduced. The collaborative process cartography contains main processes and sub processes. Each sub process is classified to in strategy process, operation process and support process.

J presents the supporting framework and software tools in the background.

- Collaborative Framework (chapter II, section III): due to the complexity of the MISE 2.0 engineering approach, a framework is very essential to manage the engineering approach and to present the position of the MISE 2.0 engineering approach. The collaborative framework with three dimensions is defined.
- Software Tools (chapter V): for the implementation, because the software tool deals with a collaborative situation, all the partners may use the software at the same time or individually. Secondly, in order to interact with other software tools (which have been developed in our lab), the software should be able to deploy on an ESB (Enterprise Service Bus). This consequently means that the software tool involved in the methodology should be a web service. SaaS (Software as a Service) seems to be a good solution. Both software tools (Mediator Modeling 2ool and Ontology Definition Tool) are developed as SaaS.
- The first part of the software tool is Mediator Model 2ool, which deals with the definitions of the collaborative network model and function model and the deduction of the collaborative process model.

Question 4, if we use the transformation mechanism of the BVR, how could we solve question of the limits of the MIT process handbook?

- Ontology Definition Tool (second part of software tools) is developed to collect collaborative knowledge from different domain and then insert or transform them in the collaborative ontology as instances. The tool aims to enrich the instances of the collaborative ontology to avoid the limits of MIT process handbook. Tiexin Wang works on this part for his master internship.

IV. Conclusion

The MISE 1.0 project is a global solution to promote interoperability between industrial actors who intend to collaborate following MDA principles. MISE promotes an interoperability design tool that aims to deal with data exchange functionalities, services sharing and dynamic process orchestrations. The engineering is based on a four-step process:

- Build collaborative network and collaborative processes;
- Transfer collaborative processes models into PIM;
- Gather technical information and transfer PIM to PSM;
- Transfer PSM to target software system.

In order to provide a clear viewpoint on the abstract level, the BVR and BST are explained and discussed. They followed the same approach. The main difference is that the BST defined a special crisis model and service model to fit the crisis domain. Besides, the assumptions made at the abstract level, concrete level and general approach during a first experience are depicted and have been discussed.

We also introduced introduction of business process management. The ARIS extension tells that capability modeling (service model in BST) and service type have to be collected in the CIM. The situation calculus gives us a clue about how could the business functions be selected for business goals. The process abstraction shows that one detailed process only cannot fit the needs of collaborative process modeling. A main process or main process of several levels has to be provided. Furthermore, the modeling level of a detailed process and abstract process is presented.

As far as the abstract level is concerned, the main problems are that i) collaborative process only covers operation level and ii) the MIT process handbook limits the knowledge used in collaborative ontology. MISE 2.0 project was started in order to solve the problem. The research works presented in this manuscript aims to improve business level work and build collaborative process model with different levels (strategy, operation and support). After the introductions and discussions of the BVR and BST, we know that the abstract level has to go through three steps. The abstract level (business level) design of MISE 2.0 also follows three steps:

- Knowledge gathering: the objective model and function model are defined to gather collaborative knowledge as simply as possible and as little as possible;
- Process deducing: collaborative ontology and transformation rules are defined. In order to solve the problems of the business function selection and process sequence deduction, we also defined several algorithms;
- Process presenting: the BPMN based collaborative process model is enlarged to the main process and sub process (strategy process, operation process and support process).

In the next chapter (chapter II), the collaborative framework is presented. The framework aims to define and organize all the collaborative knowledge, which is needed at each step of the model-driven approach of MISE 2.0. The chapter first provides a small state of the art of the enterprise architecture framework, enterprise integration framework and enterprise interoperability framework. Then the reason of building the collaborative framework is addressed. Finally, the collaborative framework (a three-dimension cube) is presented in section III. The relationships between the elements of each dimension are also explained.

Chapter II: Collaborative Situation Framework

I. Introduction 30
II. State of Art 31
II.1. Enterprise Architecture Framework 32
II.1.1. CIMOSA 32
II.1.2. GERA 34
II.1.3. ARIS 36
II.2. Enterprise Interoperability Framework 37
II.2.1. IDEAS 37
II.2.2. AIF 38
II.2.3. EIF 39
II.3. Conclusion 41
III. Collaborative Situation Framework 44
III.1. Framework Introduction 44
III.2. Relations of Collaborative Elements 45
III.3. Relations of Collaborative Lifecycle 46
III.4. Relations of Collaborative Levels 47
IV. Conclusion 48

I. Introduction

This chapter aims to present the collaborative knowledge framework of MISE project. The position of this chapter in the UJ picture is presented in Figure II-1 (the module in the bold black dash line). The engineering approaches of both MISE 1.0 and MISE 2.0 are complex. The engineering approach of MISE 1.0 goes through the model transformations from CIM to PIM and from PIM to PSM, and also contains the management of Agility. Each step of model transformation gathers or transfers different types of collaborative knowledge. The engineering approach of MISE 2.0 is also based on MDA and even covers more collaborative knowledge than MISE 1.0. In order to well organize the structure of MISE project and to well present collaborative knowledge, the collaborative situation framework is defined.

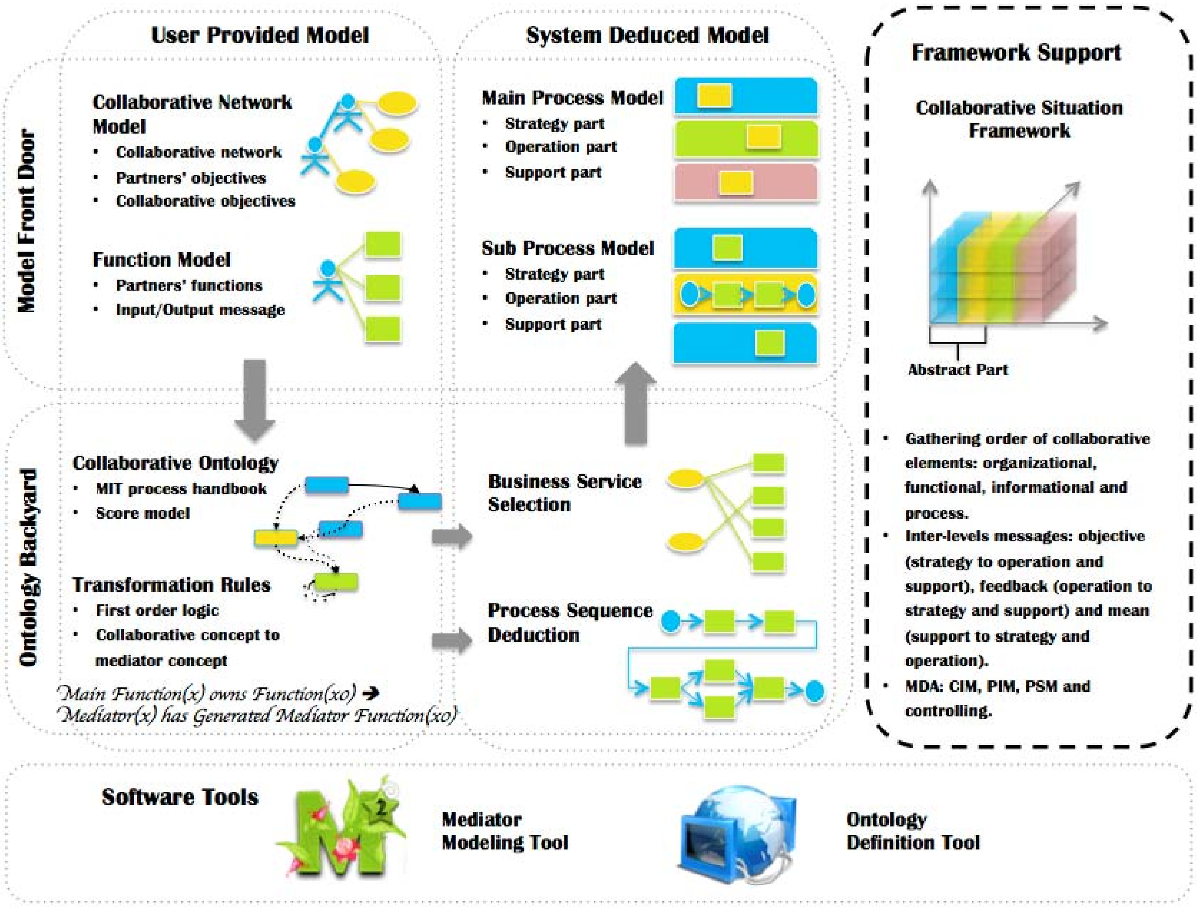

User Provided Model
System Deduced Model
Framework Support
Model Front Door
Collaborative Network Model
Collaborative network
Partners' objectives
Collaborative objectives
Function Model
Partners' functions
Input/Output message
Main Process Model
Strategy part
Operation part
Support part
Sub Process Model
Strategy part
Operation part
Support part
Collaborative Situation Framework
Abstract Part
Ontology Backyard
Collaborative Ontology
MIT process handbook
Score model
Transformation Rules
First order logic
Collaborative concept to mediator concept
Main Function(x) owns Function(xo) →
Mediator(x) has Generated Mediator Function(xo)
Business Service Selection
Process Sequence Deduction
Gathering order of collaborative elements: organizational, functional, informational and process.
Inter-levels messages: objective (strategy to operation and support), feedback (operation to strategy and support) and mean (support to strategy and operation).
MDA: CIM, PIM, PSM and controlling.
Software Tools
Mediator Modeling Tool
Ontology Definition Tool

FIGURE II - 1 POSITION OF CHAPTER II IN UJ PICTURE

What is a framework? A framework can be understood in this way. If a thing is complex, we cut the thing into small modules, and then we try to understand each module. If the small module is still complex, we cut it again. The method or the structure of how we cut the thing could be seen as a framework. The framework helps to simplify the complexity of an object. The framework provides a single map or an abstract to understand the object. An enterprise is an object, which contains numerous information from different domains, for example, people, apartment, marketing, production, and so on. In order to present and optimize the knowledge, the knowledge is defined by model, which is the world of enterprise modeling. An enterprise could be modeled from different vision and different objective. There exist many enterprise's modeling tools and modeling languages. An enterprise architecture framework is very necessary to classify enterprise models and to define the prioritization of enterprise modeling. In chapter II, section II.1, the state of art of enterprise architecture framework is presented.

As mentioned, the cooperation and collaborative between enterprises are critical issues. An enterprise may buy products from other enterprises. It may outsource orders with others. It may share resources with others. This leads to another domain: enterprise interoperability. Enterprise interoperability means the ability of en enterprise to cooperate with another in human level, operation level and information system level. Enterprise interoperability tells if an enterprise could fit into any kinds of collaborations with any kinds of partners. Enterprise architecture framework manages the internal information of enterprise. Enterprise interoperability framework manages the external interface of the enterprise with others. So enterprise interoperability framework may cover the part of the knowledge of enterprise architecture framework. Section II.2 of chapter II provides a state of art for enterprise interoperability framework.

Enterprise interoperability is an ability of an enterprise. Then the enterprises' collaboration is how the enterprises use and promote their own interoperability to get more economic interest. The collaborative situation framework of MISE 2.0 project is based on enterprises' collaboration. It defines the types of the collaborative knowledge and the approach to deduce the MIS. It differs from enterprise interoperability framework. The collaborative situation framework of MISE 2.0 describes not only the collaborative situation but also the methodology to deduce the MIS. Section III of chapter II addresses the definition of the collaborative knowledge framework and the hidden relationships and priorities in the framework.

II. State of Art

In general, framework is a real or conceptual structure intended to serve as a support or guide for the building of something that expands the structure into something useful. In English Dictionary, framework is a hypothetical description of a complex entity or process. Framework has followed features:

- *Imperfection*, it does leave some fill-in-the-blanks for its user. More powerful it is, less complex are full-in-the-blanks and more efforts in learning to use it;
- *Solubility*, it drives solution. It dictates how you do fill-in-the-blanks; it dictates overall architecture of complete specific solution;
- *Reusability*, it reduces repetitive task and often re-usable regarding high-level design consideration. (Prasanna et al., 2009)

According to Camarinha-Matos, *"in the modeling area, a framework can be seen as an "envelope" that might include a number of (partial) models, collections of templates, procedures and methods, rules, and even tools (e.g. modeling languages)"* (Camarinha-Matos and Afsarmanesh, 2008). We consider that modeling framework provides a set of viewpoints of subject, which is correlated organized and interacted. The subject is a system, an object or a situation, which needs to be modeled. The viewpoint is main element of the subject (or system). One subject could have several features. For each feature, some measures could exist. For example, human could be a subject. To model human, we could start from three viewpoints: height, weight and sex. Each element may have different measures (height: short, normal and high; weight: light, normal and heavy; sex: female and male). With viewpoints and their measures, the human could be classified. These classifications may be correlated and interactional. For example, if a person is higher, he should be heavier. If a person is female, she should be lighter than the male in the same height.

Framework could be used in several domains. A software framework, in computer programming, is an abstraction in which common code providing generic functionality can be selectively overridden or specialized by user code providing specific functionality. An Enterprise Architecture Framework defines how to organize the structure and views associated with Enterprise Architecture(Marley, 2008). The three components of the enterprise architecture framework are:

- Views: provide the mechanisms for communicating information about the relationships that are important in the architecture;
- Methods: provide the discipline to gather and organize the data and construct the views in a way that helps insure integrity, accuracy and completeness;
- Training/Experience: support the application of method and use of tools.

An enterprise architecture framework should provide views. These views are used to characterize useful and important enterprise information. For example, in Zachman Framework (1987), enterprise information is separated as: data (what), function (how), network (where), people or organization (who), time or schedule (when) and motivation or strategy (why). In order to model these information views, methods have to be provided. In Zachman framework, it provides data model (entity relationship model), organization model, process model etc. To use the methods or build models, modeling tools and modeling rules have to be learned and taught. So training and experience are necessary.

In order to understand what is the enterprise architecture framework, in section II.1, CIMOSA, GERA and ARIS are selected to explain as Enterprise Architecture Framework. CIMOSA well presents enterprise engineering and enterprise integration, which is similar with the objective of the collaborative situation framework of the MISE 2.0 project. GERA precisely defines the life cycle of enterprise engineering and manufacturing. ARIS provides Event Process Chain (EPC) and well developed software tool. Section II.2 presents IDEAS, AIF and EIF are presented as Enterprise Interoperability Framework. IDEAS manages the enterprise integration from business, knowledge and application level by solving semantic problems. AIF improves IDEAS. It defines enterprise interoperability with four levels: business, service, process and data. The solutions of each level can be ontology, semantics and model-driven interoperability. EIF differs from IDEAS and AIF. IDEAS and AIF are two dimension frameworks. EIF is three-dimension framework. It defines enterprise interoperability from three angles. Finally, in the conclusion, more enterprise architectures are briefly introduced and summarized.

II.1. Enterprise Architecture Framework

II.1.1. CIMOSA

Enterprise Integration certainly is a huge challenge to the manufacturing industry. It doesn't happen by itself, it takes strong efforts from many sides and a consensus between all parties involved reaching better integrated solutions. Enterprise integration has to be an ongoing process rather than a onetime effort. The enterprise will evolve over time according to both internal needs and external challenges and opportunities. Only if the corresponding changes are taken into account continuously will the operation gain and preserve the operational flexibility needed in today and tomorrow global markets. To solve the many problems of industry, integration has to recognize and to proceed in more than one operational aspect. Figure II-2 shows the evolution from system integration in the 1950s to the current attempts of enterprise integration.

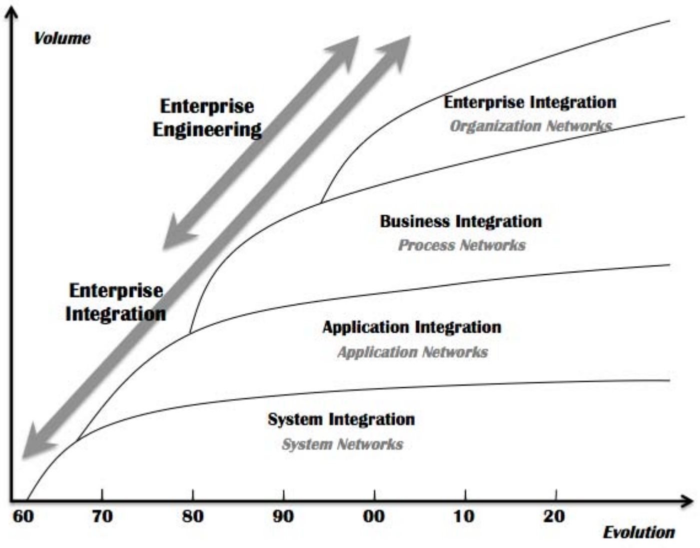


FIGURE II - 2 FROM INTEGRATION TO ENGINEERING (KOSANKE ET AL., 1999)

System integration is a wide concept. The system integrator brings together discrete systems utilizing a variety of techniques such as computer networking, enterprise application integration, business process management or manual programming. It contains Application Integration, which is concerned with the control and integration of applications in the data processing sense, which means interoperability between applications and users (humans as well as machines) and supply and removal of information through inter and intra system communications. Business Integration is concerned with integrating those functions, which manage, control and monitor business processes. Functions, which provide supervisory control of the operational processes and in turn co-ordinate the day-to-day execution of activities at the application level. Modeling of business processes and their interrelations and use for decision and operational support is key to business integration. Enterprise integration has to encompass all these levels of integration. However, the emphasis has to be on business integration. Only with a focus on the business needs rather than on application or system, needs all the aspects of enterprise operation, which can be identified and considered in the course of modifying and optimizing the operation itself.

CIMOSA: Computer Integrated Manufacturing Open System Architecture (Kosanke et al., 1999), derived from the ESPRIT-funded consortium AMICE, aimed to develop an all-embracing conceptual framework in implementing Computer Integrated Manufacturing (CIM). CIMOSA is a reference model and a complete description of a manufacturing enterprise using various representations such as organization, resource, information and function. It describes, using these representations, each function and its activities of the enterprise in generic form. The areas within the scope of CIMOSA are product information, manufacturing planning and control information, shop floor information and basic operation information. As open system reference architecture for CIM, CIMOSA supports the definition, development and continuous maintenance of a consistent architecture and its related operational system for a particular enterprise. This particular architecture will provide the explicit structure of the enterprise operation and thereby allow the modeling, simulation and control in real time of all internal and external information needs of the whole enterprise, including its relationships to suppliers, customers, government agencies, financial service, etc.

CIMOSA has four modeling views Function, Information, Resource and Organization. This set of views may be extended if needed. The CIMOSA Reference Architecture supports three modeling levels of the complete lifecycle of enterprise operations. They are Requirements Definition, Design Specification and Implementation Description. The enterprise modeling sequence is optional. The modeling process can be started in each phase of the lifecycle. It also allows different people modeling different parts of the cube and then combines them together.

The modeling framework, which is shown in Figure II-3, structures the CIMOSA Reference Architecture into a generic and a partial modeling each level supporting different views on the particular enterprise model. The concept of views allows working with a subset of the model rather than with the complete model providing especially the business user with a reduced complexity for his particular area of interest.

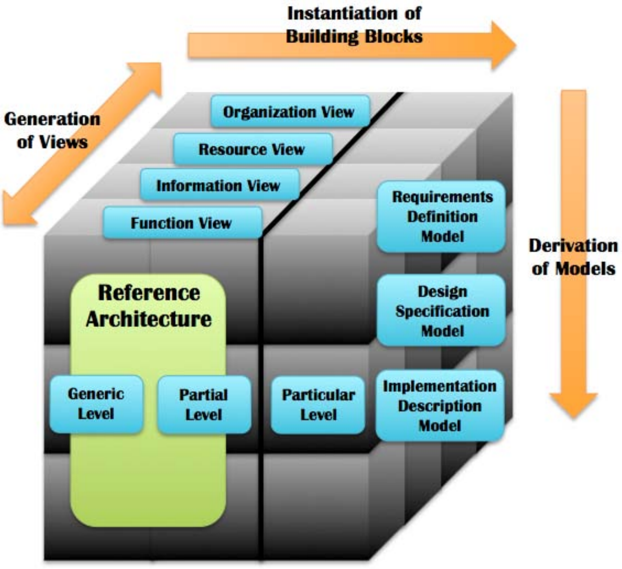


FIGURE II - 3 CIMOSA MODELING FRAMEWORK (ZELM ET AL., 1995)

CIMOSA modeling process has been introduced in (Zelm et al., 1995). We can say that CIMOSA is a decomposition process. It has five steps.

i. Model enterprise domains: define domains' processes by domains' functions. Events and results among these domain processes have to be shown.
ii. Model business processes in each domain process. Focus on each domain processes, represent detailed business processes and enterprise activities.
iii. Represent domain process by enterprise activities' network. In this phase, a process diagram, which is made up by events and results of, the domain process has to be defined. The process diagram describes a execute process.
iv. Define inputs, outputs, control I/O and resources for each enterprise activities.
v. Decompose enterprise activities into functional operations. Such Functional Operations are defined in relations to their executing resource types: the Functional Entities. One Functional Entity will execute functional Operation, but a Functional Entity may be executed more than one type of Functional Operation.

II.1.2. GERA

GERAM: Generic Enterprise Reference Architecture and Methodology defines a tool-kit of concepts for designing and maintaining enterprises for their entire life history. GERA: Generic Enterprise Reference Architecture (Force, 1999) is crucial component of GERAM. It defines the enterprise related generic concepts recommended for use in enterprise engineering and integration projects. These concepts contain human oriented concepts, process oriented concepts and technology oriented concepts.

Human oriented concepts, describe the role of humans as an integral part of the organization and operation of an enterprise. The key benefit of human oriented concepts is reusability. The role of individuals and individual groups, organizational structure and individual's capability could be reused. Its reuses can enable an enterprise to respond rapidly in new business environment, reengineer business and manufacturing processes, improve its management and utilization of resources and improve resilience to the loss of core competencies.

Process oriented concepts describe enterprise business processes. The process-oriented concepts defined in GERA are: enterprise entity life-cycle and life-cycle phases, life history, enterprise entity types, and enterprise modeling with integrated model representation and model views. Unlike CIMOSA, GERA process oriented concepts provide a concept: life history (in Figure II-4). *"The life history of a business entity is the representation in time of tasks carried out on the particular entity during its entire life span".* (Force, 1999) This demonstrates the iterative nature of the life-cycle concept compared with the time sequence of life history. These iterations identify different change processes required on the operational processes and, or the product or customer services.

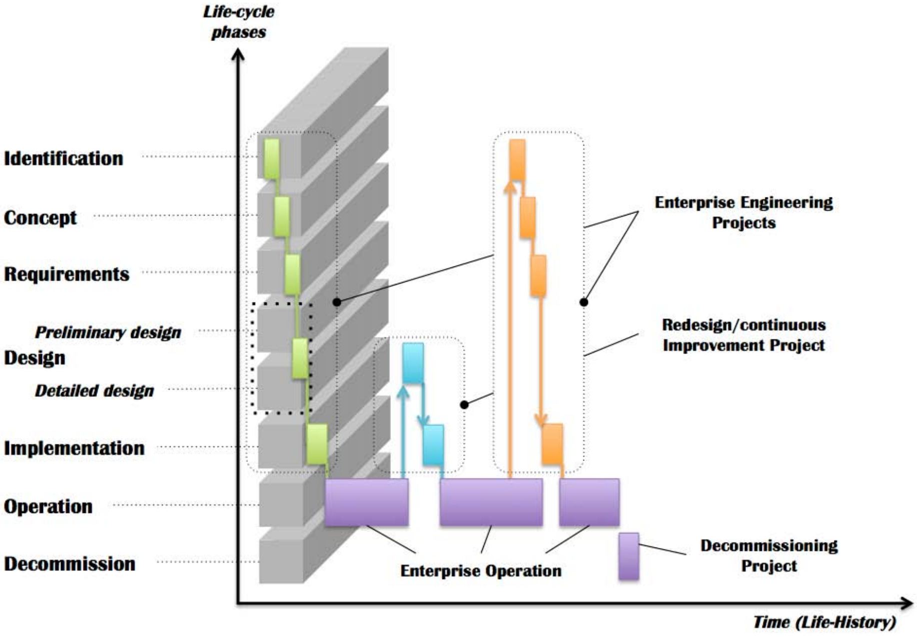


FIGURE II - 4 RELATIONSHIPS BETWEEN LIFECYCLE AND LIFE HISTORY (KOSANKE ET AL., 1999)

Technology oriented concepts, describe business process supporting technology involved in both enterprise operation and enterprise engineering efforts. Technology oriented concepts have to provide descriptions of the technology involved in both the enterprise operation and the enterprise engineering efforts.

As introduced in (Force, 1999), GERA provides an analysis and modeling framework (Figure II-5) that is based on the life-cycle concept and identifies three dimensions for defining the scope and content of enterprise modeling:

- Life-Cycle Dimension: providing for the controlled modeling process of enterprise entities according to the life-cycle activities;
- Genericity Dimension: providing for the controlled particularization (instantiation) process from generic and partial to particular;
- View Dimension: providing for the controlled visualization of specific views of the enterprise entity.

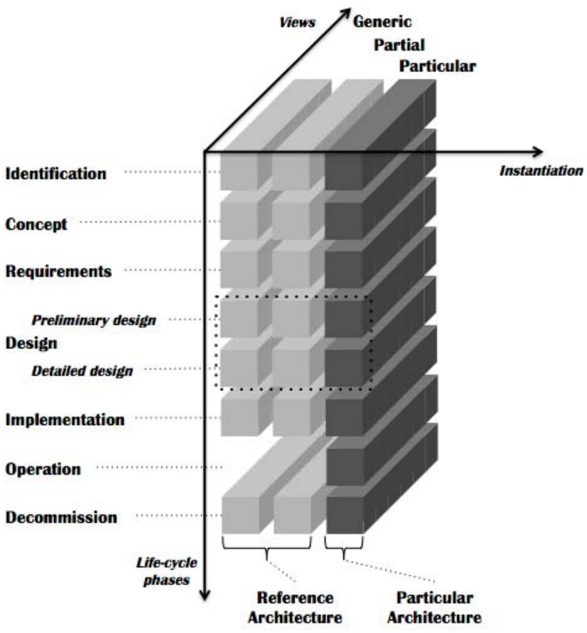


FIGURE II - 5 GERA MODELING FRAMEWORK

As shown in Figure II-5, the reference part of the modeling framework covers the generic and the partial levels only. The particular level represents the results of the modeling process - which is the model or description of the enterprise entity at the state of the modeling process corresponding to the particular set of life-cycle activities. However, it is intended that the modeling languages should support the derivation of models from an upper to a lower state or the abstraction of lower models to an upper state.

II.1.3. ARIS

ARIS: Architecture of Integrated Information Systems (Lankhorst, 2009) is an approach to enterprise modeling. ARIS started as the academic research of Prof August-Wilhelm Scheer in the 1990s. It offers methods for analyzing processes and taking a holistic view of process design, management, workflow, and application processing. The ARIS-approach not only provides a generic and well documented methodological framework but also a powerful business process-modeling tool.

The conceptual design of the Architecture of Integrated Information Systems (ARIS) is based on an integration concept, which is derived from a holistic analysis of business processes. The first step in creating the architecture calls for the development of a model for business processes, which contains all basic features for describing business processes. The result is a highly complex model, which is divided into individual views in order to reduce its complexity. According to (Scheer and Schneider, 2006), the views of ARIS are explained as followed:

- Function view: the processes transforming input into output are grouped in the function view; The designations “function”, “process” and “activity” are used synonymously. Objectives are also allocated to the function view – because of the close linkage.
- Organization view presents the hierarchical organization structure. It is created in order to group responsible entities “human output”, responsible devise, “financial resources” and “computer hardware” are allocated to the organization view;
- Data view comprises the data processing environment as well as the messages triggering functions or being triggered by functions. Preliminary details on the function of information systems as data media can be allocated to data names. Information services objects are also implicitly captured in the data view;
- Output view contains all physical and non-physical input and output, including funds flows;

- Control view/Process view displays the respective classes with their view-internal relationships. Relationships among the views as well as the entire business process are documented in the control or process view, creating a framework for the systematic inspection of all bilateral relationships of the views and the complete process description.

ARIS varies three main perspectives of techniques. ARIS uses a modeling language known as Event Process Chains (EPC), which is an important aspect of the ARIS-model. EPC is the center of the House of ARIS and connects all other views, as well as describes the dynamics of the business process. It differs from swim lane because it is process oriented and swim lane is function oriented. On the other hand, based on the conceptual description, ARIS can model and structured Business Process Models. Furthermore, ARIS House[1] has been developed to implement business models in information system. Figure II-6 shows the three layers of ARIS: business concept layer, IT concept layer and implementation layer. The data view, the control view and the functional view use different types of models or technical files to present and use the same kind of knowledge. The business concept layer is the externally visible functionality, which is meaningful to the environment and is realized by business behavior. The business concept layer is similar with the CIM of MDA. The IT concept layer is an externally visible unit of functionality, provided by one or more components (the PIM of MDA). The implementation layer is a software system designed to support interoperable machine-to-machine interaction (the PSM of MDA). The ARIS also concludes MDA as one crucial character of enterprise integration.

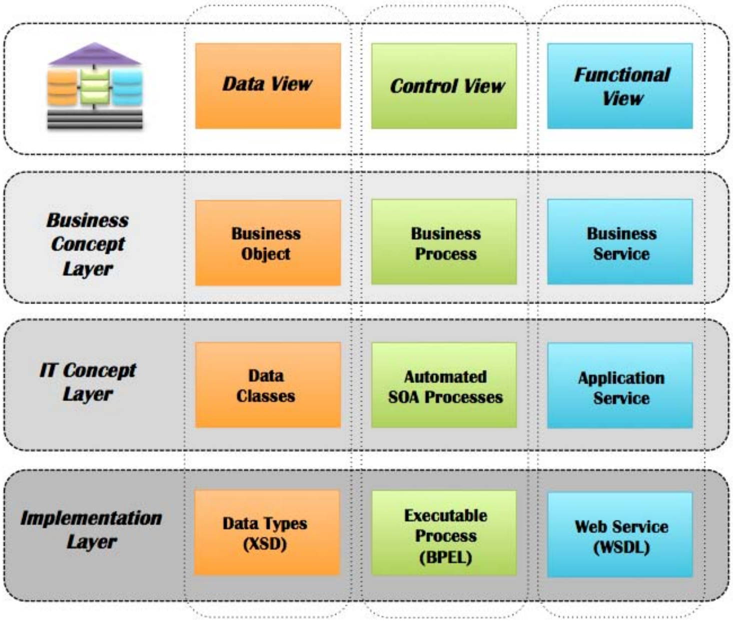


FIGURE II - 6 ARIS MODELING FRAMEWORK (IDEE, 2009)

II.2. Enterprise Interoperability Framework

II.2.1. IDEAS

IDEAS stands for Interoperability Development for Enterprise Application and Software. (IDEAS, 2003) It is a thematic network project intending to deliver roadmaps in the domain of interoperability of enterprise applications and software. It defines interoperability as an interaction capability between enterprise software applications.

[1] The ARIS house is the ARIS framework, which looks like a house. The ARIS framework defines organization view, function view, data view, output view and control view. Each view includes concept, data processing concept and application.

According to **(IDEAS, 2003)**, the IDEAS interoperability framework has three levels: business, knowledge, and ICT (Information and Communication Technology) systems. A common semantic relates these three levels to each other. They should be considered all together in order to obtain substantial and effective results, as well as pragmatic applications in today's business world. The IDEAS interoperability framework describes how interoperability can be achieved if the interactions can at least take place at three levels between two cooperating enterprises. Figure II-7 shows the IDEAS framework:

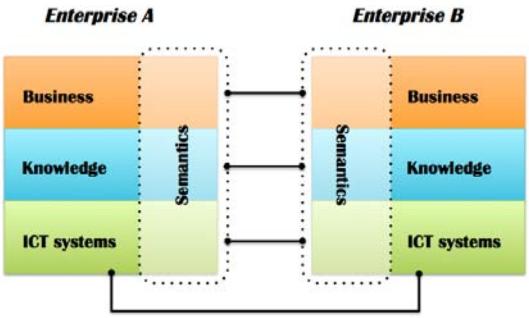


FIGURE II - 7 IDEAS INTEROPERABILITY FRAMEWORK (IDEAS, 2003)

- The Business level concerning the problems related to the organization, and business processes. It is divided into three sub-levels: decisional model, business process, and business model;
- The Knowledge level concerns acquiring, structuring, and representing knowledge of enterprises;
- The ICT system level (application and data) level concerns the technical solutions for transferring resources from one enterprise to the others.

The semantic barriers concern mutual understanding on all layers, for example, business terms for the business level, dictionaries and ontologies for the knowledge level, and ontology tools and services for the application level. This barrier concerns every level of the enterprise when it establishes interoperability with others.

II.2.2. AIF

AIF: ATHENA interoperability framework adopts a holistic perspective on interoperability by integrating the different results and solutions developed in the ATHENA project. It builds upon the thematic network of IDEAS and merges three research areas: 1) architecture and platform to provide implementation frameworks, 2) enterprise modeling to define interoperability requirements and to support solution implementation, and 3) ontology to identify interoperability semantics in the enterprise. The AIF aims to provide approaches to the solution, while the IDEAS framework focuses on structuring the interoperability issues (into business, knowledge, semantic, and technologic issues). According to (Berre et al., 2007), the AIF is structured into three parts as follows:

- Conceptual integration focuses on concepts, metamodels, languages, and model relationships. It provides us with a modeling foundation for systemizing various aspects of interoperability;
- Application integration focuses on methodologies, standards and domain models. It provides us with guidelines, principles and patterns that can be used to solve interoperability issues.
- Technical integration focuses on the technical development and ICT environments. It provides us with ICT (it can be described as a synthetic discipline that studies phenomena created by humans rather that those given by nature, and there is a huge room for creativity and few direct physical constraints) tools and platforms for developing and running enterprise application and software systems.

Figure II-8 is a simplified view of the reference model that indicates the required and provided artifacts of two collaborating enterprises:

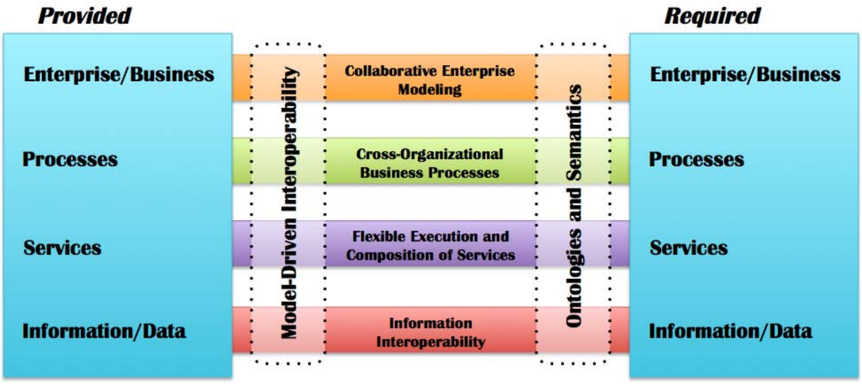

Provided
Required
Enterprise/Business
Processes
Services
Information/Data
Model-Driven Interoperability
Collaborative Enterprise Modeling
Cross-Organizational Business Processes
Flexible Execution and Composition of Services
Information Interoperability
Ontologies and Semantics
Enterprise/Business
Processes
Services
Information/Data

FIGURE II - 8 ATHENA REFERENCE ARCHITECTURE (BERRE ET AL., 2007)

Interoperations can take place at four levels:

- The Enterprise/business level concerning the organizational and operational ability of an enterprise to cooperate with others. This level requires a collaborative enterprise modeling;
- The Process level focusing on making various processes work together. A cross-organizational business process is needed at this level;
- The Service level concerned with identifying and executing applications. This requires a flexible execution and composition of services which can be supported by PIM4SOA (PIM for SOA);
- The Information level concerning the management and exchange of messages (information interoperability).

For each level, the model-driven interoperability approach is used to formalize and exchange the provided and required artifacts that must be negotiated and agreed upon. The semantic annotation gives meaning to any kind of resources (e.g. business processes) in terms of the shared reference ontology and reconciliation rules.

- Applicative integration focuses on methodologies, standards, and domain models. This part ensures the establishment of interoperability by providing guidelines, principles, and patterns that can be used to solve interoperability problems;
- Technical integration concerns the technical development, and ICT environments. It provides the ICT tools, and platforms for developing and executing enterprise application, and software systems.

This platform provides a compound framework and associated reference architecture for capturing the research elements and solutions to solve the interoperability problems. It addresses the problem in a holistic way by capturing and inter-relating information from many perspectives covering business, knowledge, technical (ICT), and semantic issues relevant to interoperability.

II.2.3. EIF

The EIF stands for Enterprise Interoperability Framework, developed in the InterOp NoE project (Interoperability Research for Networked Enterprise Applications and Software, FP6 508011) (Chen et al., 2008).This framework aims at defining the research domain of interoperability of enterprise applications. Generally, research on interoperability is an applied and problem-driven type of research. The framework has three basic dimensions: interoperability levels, interoperability barriers, and interoperability approaches.

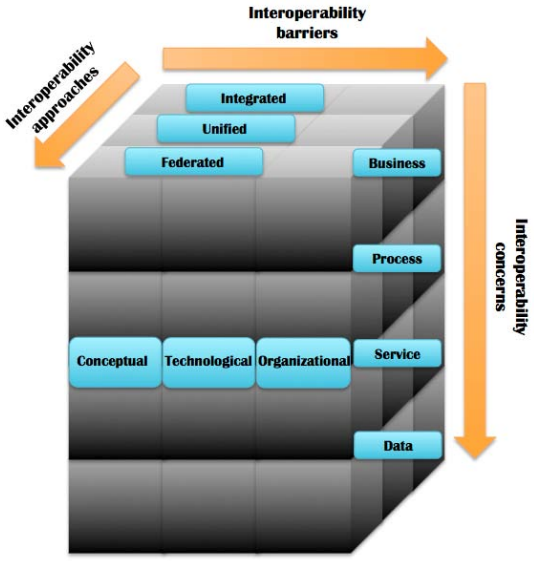


FIGURE II - 9 ENTERPRISE INTEROPERABILITY FRAEMWORK OF INTEROP (CHEN ET AL., 2008)

The interoperability levels concern the interactions that can take place from the various viewpoints of the enterprise. Four levels of interoperability have been defined:

- Interoperability of data aims to make different data models related to particular applications work together. This interoperability allows data coming from heterogeneous bases to be found and shared by different machines, operating systems, and database management systems;
- Interoperability of services concerns identification, combination, and making various applications work together by dealing with syntactic and semantic differences;
- Interoperability of processes focuses on making various processes work together;
- Interoperability of business refers to working in a seamless way at the organizational level in spite of the different modes of decision-making, culture, etc., so that business can be developed and shared between companies.

The interoperability barriers describe an incompatibility, which obstructs the sharing of information and exchanging of services. Three categories of barriers have been defined as follows:

- Conceptual barriers relating to the syntactic and semantic differences in information to be exchanged, as well as in the expressivity of the information;
- Technological barriers relating to the incompatibility of information technologies (e.g. architecture, infrastructure, etc.). This kind of barrier prevents collaboration between two or more systems;
- Organizational barriers relating to the definition of responsibility, authority, and organization structure. This kind of barrier particularly concerns human and organization behavior, which can be incompatible with interoperability.

These approaches to interoperability allow knowledge and solutions relating to enterprise interoperability to be categorized according to the ways of removing various interoperability barriers. They have been defined under three categories:

- Integrated approach, referring to the existence of a common format for all models. If the need for interoperability comes from a merger between enterprises, this approach seems the best adapted. In this case, there is only one common format agreed by all partners to elaborate models and build systems;

- Unified approach, referring to the existing of a common format, but only at a metalevel. If the need for interoperability concerns a long-term based collaboration, this approach is a possible solution. A common metamodel provides a means for establishing semantic mapping between models;
- Federated approach, having no common format. Partners do not impose their models, languages, or methods of work. For the need for interoperability originating from the short-term collaboration project, this approach is the most relevant. To interoperate, partners must adapt themselves dynamically by sharing an ontology rather than having a predetermined metamodel.

II.3. Conclusion

Enterprise Architecture started with the Zachman Framework (Zachman, 1987) in 1987. Another early implementation of an Enterprise Architecture framework was the "Technical Architecture Framework for Information Management" (TAFIM) (DTIC, 1996). The first draft of TAFIM was completed in 1991 with the TAFIM Technical Reference Model (TAFIM TRM). This technical reference model wanted to use open systems and new technologies available in the commercial market, to develop a DoD-wide application. During 80s, researches were carried out in Europe and USA to develop enterprise architecture. Among them the most known: the Purdue Enterprise Reference model: PERA (1991), Architecture of Integrated Information Systems: ARIS (1991), the Computer Integrated Manufacturing Open System Architecture: CIMOSA (1993). The Open Group Architecture Framework: TOGAF (1995) the GIM architecture (1996), Enterprise-Reference Architecture and Methodology: GERAM (1997) and Federal Enterprise Architecture Framework: FEAF (1999). In 2003, DODAF (Department of Defense Architecture Framework) was developed by the US Department of Defense. DODAF (Umheh et al., 2007) is an evolutionary upgrade of the C4ISR (Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance) architecture framework (C4ISR, 1997) both of which prescribe three views of the architecture: Operational, Technical and System. In 2005, The British Ministry of Defense Architectural Framework (MODAF) v1.0 (Biggs, 2005) was developed by MOD from DODAF version 1.0, but has been extended and modified to meet MOD requirements.

The Zachman Framework (Noran, 2003) is used for enterprise engineering and manufacturing. It provides the views from different members (e.g. planner, owner, designer, etc) and knowledge (data, function, network, etc). Zachman framework almost covers all the knowledge of enterprise. But the main problem is that there is no related methodology followed to use the framework and potential connections among views are ignored.

The GIM (GRAI Integrated Methodology) architecture (Chen et al., 1997) is a modeling methodology intended for general description, focused on details in manufacturing control system. This framework has four main parts: functional model, informational model, decision-making model and physical model. The strong point of this framework is the decision-making model: GRAI (Doumeingts et al., 2006), which allows user to model all the decision units and activities by time and organization. The weak point is that the framework considers information system as an important part in enterprise without defining detailed connections with business part.

PERA (Williams and Li, 1999) considers that enterprise has three main components: facility, organization and information system. It manages these three components by different phases (life-cycle).

Architecture of Integrated Information Systems: ARIS (Scheer, 2000) is architecture for information system integration, which is the most contiguous with the objective of MISE 2.0. ARIS manages integration by views: data, process, function, organization and control. The connections among views are considered also. It also provides modeling tools and software platform to support model building and process transformation from business to technical. But user must build the business process. The regretful point is that MDA model transformation phase is not shown in the framework.

The Computer Integrated Manufacturing Open System Architecture: CIMOSA (Kosanke, 1995) is a three-dimension cube (generation of views, instantiation of building blocks and derivation of models). The clear structure makes the framework easy to understand. Each dimension provides an angle for starting enterprise modeling. CIMOSA provides a process model without software supporting. In CIMOSA, it combines function and process in the function view. With the objective of computer integration, it limits the further development of web services.

The Open Group Architecture Framework: TOGAF (Umheh et al., 2007) is an industry standard architecture framework that may be used freely by any organization wishing to develop information systems architecture for use within that organization. It has been developed and continuously evolved. In TOGAF v9[1] (2009), TOGAF Architecture Development Method (ADM) is provided. It is used to manage the using process of all the sub-architectures (ADM guidelines & technique, TOGAF Architecture Content Framework, Enterprise Continuum, TOGAF Reference Models and TOGAF Capability Framework) in TOGAF. According to TOGAF v9 survey results (Vamus and Panaich, 2009), more than 50% organizations are using TOGAF 8 and 9 to manage their enterprises[2]. The scope of the four architecture domains[3] of TOGAF aligns with the first four rows (what, how, where and who but without when and why) of the Zachman Framework.

Enterprise-Reference Architecture and Methodology: GERAM (Force, 1999) provides a generalized framework for describing the components needed in all types of enterprise engineering/enterprise integration processes. The shining point of this framework is lifecycle. GERA (generalized enterprise reference architecture) classified generic concepts by human oriented, process oriented and technology oriented. The shining point is process oriented which detailed defines enterprise lifecycle into enterprise engineering, re-engineering and re-design. And then, the framework even details views (model content, purpose, implementation and manifestation) and objects (customer service, software, hardware, information, function, machine, human etc.) on the lifecycle. The GERAM framework defines the minimal set of elements, which should be accompanied with, to build enterprise architectures. But these elements are abstract, for example, enterprise-engineering methodologies, modeling languages, modeling methodology and so on. Users have to develop their own specific methodology or choose a developed methodology.

Federal Enterprise Architecture Framework: FEAF (Council, 1999) provides a common methodology for information technology acquisition, use and disposal in the Federal government. It provides performance reference model, business reference model, service component reference model, data reference model and technical reference model.

In recent years, it has become apparent that a key benefit to be gained from Enterprise architecture is the ability to support decision-making in changing businesses. Because Enterprise Architecture brings together business models (e.g. process models, organizational charts, etc.) and technical models (e.g. systems architectures, data models, state diagrams, etc.) it is possible to trace the impact of organizational change on the systems, and also the business impact of changes to the systems. (Marley, 2008) As this benefit has emerged, many frameworks such as DoDAF and MODAF have adopted a standard metamodel (UPDM 1.0 based on DoDAF v1.5 and MODAF v1.2, 2009), which defines the critical architectural elements and the dependencies between them. Application based on these models can then query the underlying architectural information, providing a simple and strong mechanism for tracing strategies to organizational and technological impacts.

The architectures mentioned above have been summarized in Table II-1.

[1] http://www.opengroup.org/togaf/

[2] TOGAF 8: 30%, TOGAF 9: 21%, Zachman: 24%, FEAF: 7%, DODAF: 7% and MODAF 2%

[3] Data architecture (what?), business architecture (how?), technical architecture (where?) and applications architecture (who?)

TABLE II - 1 SUMMARY OF ENTERPRISE ARCHITECTURE

Framework	Type	Update	Completeness	Practicability	Tool Support	Method Support	Summary
Zachman (1987)	EA	2011 V3	Middle	Low	No	No	A general framework without specification
GIM (1988)	EA-PM	No	Middle	Middle	No	Yes	Develop a useful decision making method-GRAI
PERA (1991)	EA	No	Middle	Middle	No	No	Consider enterprise as facility, organization and information system
ARIS (1991)	EIA-IS	2011 V7.2	Middle	High	Yes	Yes	MDA and SOA based, consider enterprise integration views as organization, data, function process and control
CIMOSA (1993)	EIA	No	Middle	Middle	No	Yes	Clear structured three dimensions' framework which are views, models and levels
TOGAF (1995)	EA-IS	2009 V9	High	High	No	Yes	Architecture of architectures, each part is well detailed
GERA (1997)	EA	No	High	Low	No	Yes	Enterprise engineering lifecycle is well detailed
FEA (1999)	EA-IS	2012 V3	High	High	No	Yes	Organize enterprise by levels: business, design, application and technology; contain as-is and to-be system modeling
DoDAF (2003)	EA-IS	2009 V2.0	High	Middle	No	Yes	Separates enterprise by viewpoints: all, data and information, standards, capability, operational, services, systems, project
IDEAS (2003)	EF	No	Low	Low	No	No	Defines interoperability levels: business, knowledge and application. The solution is semantics
AIF (2007)	EF	No	Middle	Middle	No	No	Defines interoperability levels: business, service, process and data. The solutions are ontology, semantics and model-driven interoperability
EIF (2008)	EF	No	High	Middle	No	No	Defines interoperability from: interoperability concerns, interoperability barriers and interoperability approaches

Enterprise Architecture: EA; Enterprise Integration Architecture: EIA; Enterprise Interoperability Framework: EF
Information System: IS; Product Manufacturing: PM

III. Collaborative Situation Framework

In this section, the collaborative situation framework is presented. The framework is represented as three dimensions: collaborative situation elements, collaborative situation life cycle steps and collaborative situation levels. In section III.1, the dimensions of the collaborative framework are explained. Section III.2 presents the collaborative situation elements in detail (gathering order of collaborative situation elements). Section III.3 addresses the collaborative situation life cycles (back to PIM and back to CIM). The collaborative situation levels are explained in section III.4.

III.1. Framework Introduction

One of the tasks mentioned in section III of chapter I is to define a collaborative situation framework, which is clearly structured and easily understood. The collaborative situation framework should also cover all the collaborative knowledge and direct collaborative situation modeling and helps mediation information system generation. In MISE 2.0, we consider that a collaborative framework should define viewpoints by organization, function, information, process and inter-connections among them. Furthermore, the engineering approach of MISE 2.0 goes through all the steps of MDA. So in our framework, two dimensions with viewpoints and MDA are confirmed.

However, almost all the frameworks mentioned in chapter II, section II.3 have a module or a unit for strategy management or decision-making, which is not shown in the main framework. Furthermore according to ISO 9000 (ISO 9000, 2005; ISO 9000 X50-130, 2005), a business process should contain strategy process, operation process and support process. With our experience on MISE 1.0 deployment, one collaborative process is not good enough to manage collaborative situation. It is very hard to understand for different levels' managers and workers. So we break the two dimensions framework into 3 levels. As shown in Figure 2, it is MISE 2.0 collaborative knowledge framework.

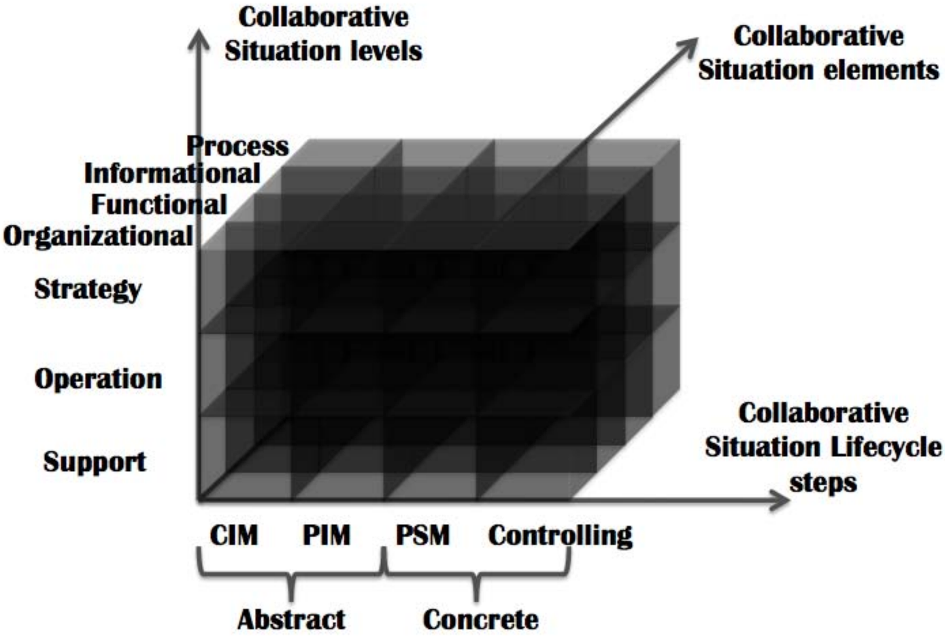


FIGURE II - 10 FRAMEWORK OF COLLABORATIVE SITUATION

MISE 2.0 collaborative situation framework has three dimensions:

- Collaborative situation lifecycle steps, separate collaboration situation knowledge by mediation information system building steps. The collaboration situation lifecycle covers CIM, PIM, PSM and Controlling. The CIM and PIM respect to the abstract level. In the abstract level, business information and collaboration requirement have to be gathered. With this information, the business collaborative process may be deduced. Then, at the concrete level, the problem of semantic web service may be additional at PSM and Controlling stages. In this part, collaboration process and semantic information are used to build target mediation information system.
- Collaborative situation levels, separate collaboration situation knowledge by different collaboration management levels. The dimension provides not only the operation level but also the strategy and support level. The strategy level helps decision-making, collaboration direction choosing and management level communicating. The operation level provides detailed collaboration solutions and execution results. The support level complements needs and functions for operation level and strategy level.
- Collaborative situation elements, separate the collaboration situation knowledge by different knowledge viewpoints. It covers the organizational view, the informational view, the process view and the functional view. The organizational view concerns collaboration network, partners and collaborative objective. The informational view provides basic business data. Process view provides collaboration process. The functional view provides the capabilities of each partner.

The goal of collaborative situation framework is to transfer organizational, functional and informational elements of CIM level to process element (which presents the process as strategy, operation and support process) in PIM level.

III.2. Relations of Collaborative Elements

Relations of Collaborative Elements (RCE) are based on Collaborative Situation Framework Abstract level. The RCE aims at: i) defining modeling methods or modeling languages to gather or organize the collaborative knowledge at abstract level, which is defined in Collaborative Situation Framework and ii) providing the gathering orders of collaborative elements at the abstract level. As shown in Figure II-11, the RCE has two parts: i) organizational, functional, informational and process and ii) models and metamodel (and ontology). In MISE 2.0, we consider that there exists an order, which should be followed when the collaborative elements are gathered.

In our point of view, when the collaboration starts, the first thing to know is: what are the objectives? And who are the partners? The **organizational elements** should be gathered first. For these elements, collaborative network, partners, partners' relationships and objectives of network and partners are gathered. All the knowledge of organizational element is the initial knowledge for a collaborative situation. In order to gather the organizational elements, an organizational model is necessary to gather and present the organizational knowledge (the MISE 2.0 uses the collaborative network model to gather organizational elements, the collaborative network model is explained in detail in chapter III, section II).

In organizational elements, the objectives and the partners of the collaboration are provided. Then the next thing to know is: if the partners are willing to involve in the collaboration, what are the functions of partners? Which functions could be used to achieve the identified objectives? So, the **functional elements** should be gathered second. For this element, partners' functions have been gathered. In order to fix this requirement, a functional model is required to gather partners' functions. The abstract level of MISE 2.0 reuses IDEF0 to gather functional information. The functional model of MISE 2.0 is explained in chapter III, section III.

Even though normally, a functional model does not just gather functional information. It also covers input/output messages, which are exchanged among functions. In some case, the input/output messages of functional model do not contain enough informational knowledge. An informational model may be necessary to gather additional informational knowledge to complete the collaborative knowledge. The additional knowledge of informational knowledge may provide the attributes of messages, the relations among messages and semantic annotation. The third elements to gather are the **informational elements**.

Finally, all the information, which has been gathered by above three types of elements are re-used, re-organized and re-presented to deduce a collaborative process model. This collaborative process model (chapter III, section IV) is based on BPMN. This BPMN based collaborative process model is specialized to one mediation pool (containing three collaborative lanes: strategy process, operation process and support process) and several partners' pools. In order to transfer organizational, functional and informational elements as process element, the definitions of models cannot accomplish the transformation. The modeling elements of organizational, functional, informational and process model should be managed and confirmed by metamodel or ontology. Based on ontology and metamodel, transformation rules could be defined to transfer organizational, functional and informational models to process model. In MISE 2.0, the collaborative ontology and the model transformation rules are defined to complete this mission. This part of work is presented in chapter IV.

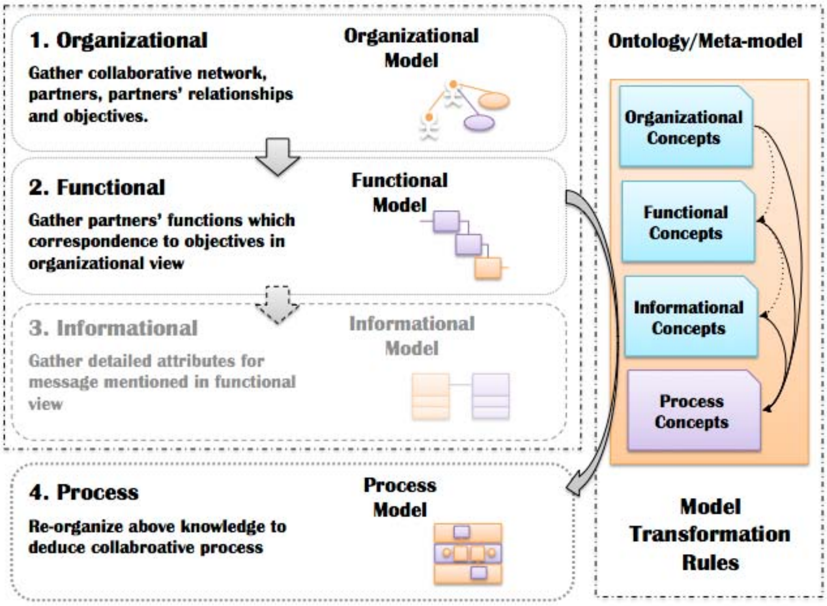

1. Organizational
Gather collaborative network, partners, partners' relationships and objectives.
Organizational Model
2. Functional
Gather partners' functions which correspondence to objectives in organizational view
Functional Model
3. Informational
Gather detailed attributes for message mentioned in functional view
Informational Model
4. Process
Re-organize above knowledge to deduce collabroative process
Process Model
Ontology/Meta-model
Organizational Concepts
Functional Concepts
Informational Concepts
Process Concepts
Model Transformation Rules

FIGURE II - 11 RELATIONS AMONG COLLABORATIVE SITUATION ELEMENTS

Relations among collaborative situation elements mainly provide an order to gather collaborative knowledge. The order is: i) gathering organizational elements, which contains collaborative objectives, collaborative network and partners, ii) gathering functional elements, which requires shared functions of partners and part of input/output messages, iii) gathering informational elements, which demands attributes of messages, relations of messages and semantic annotations and iv) transferring above organizational, functional and informational elements to process element. In order to obtain the main goal of process elements, ontology/metamodel and model transformation rules are defined.

III.3. Relations of Collaborative Lifecycle

In the collaborative situation framework, the collaborative situation lifecycle contains CIM, PIM, PSM and controlling. In the collaborative situation framework, reader could understand them as: the collaborative situation lifecycle starts with the CIM, moves from the CIM to the PIM, from the PIM to the PSM. The controlling helps to go back to the CIM and to start over a new cycle. But the dimension of collaborative situation lifecycle is not that simple. The dimension could be opened and presented in a much more complex way. In order to present the dimensions correctly, the relations of collaborative lifecycle (RCC) are defined (Figure II-12).

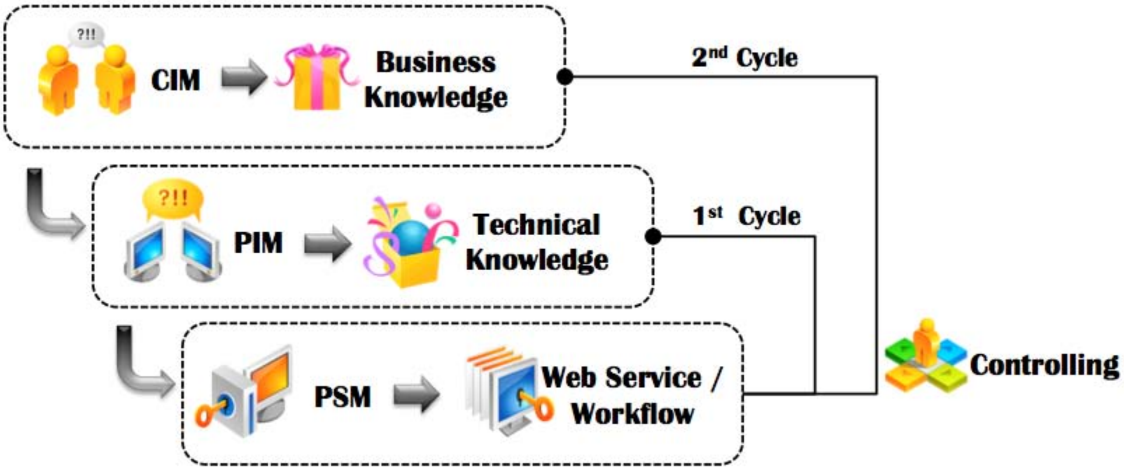


FIGURE II - 12 RELATIONS IN COLLABORATIVE SITUATION LIFECYCLE

As presented in the collaborative situation framework, the dimension of lifecycle is separated as four layers: CIM, PIM, PSM and controlling. The RCC in Figure II-12 also contains these four layers. The CIM present or define the gathered collaborative knowledge. The knowledge of CIM is business knowledge. But the knowledge of PIM is technical knowledge. In order to move from the CIM to the PIM, there is a gap to fix. The gap is to add the technical knowledge and transfer form the CIM to the PIM. After gathering technical knowledge, the lifecycle moves from the CIM to the PIM.

The knowledge of the PIM contains technical functions of each partner. But technical functions are not web services. The technical functions have to be implemented or executed by web services. Semantic web service is the next gap to fix. Then the PIM is transferred to the PSM. The lifecycle moves from the PIM to the PSM. The PSM is deployed as mediation information system (MIS) at run-time (it is an ESB system to orchestrate BPEL file). Though the MIS is launched to invoke the whole collaborative process. There may be several kinds of failures and errors at run-time. This leads to the last layer of lifecycle: the controlling. The controlling is a layer to decide that which layers of design-time lifecycle should be redone to point against the specific failures or changes at run-time. The RCC defines two kinds of lifecycle:

- The first lifecycle goes back to the PIM layer. It is designed to solve the failures of technical knowledge. For example, if the web service of one technical function is down, the semantic web service has to be redone to select new web services, which could implement the technical function.
- The second lifecycle goes back to the CIM layer. It is designed to correct the mistakes of business knowledge. For example, if a new partner entered the collaborative situation or a partner is no longer available for the collaborative situation, the lifecycle has to restart all over from the beginning to collect the correct business information.

III.4. Relations of Collaborative Levels

As we have mentioned in previous section, all the models, which have been defined in the RCE, cover strategy, operation and support level. As the results of process deduction architecture, strategy, operation and support collaborative processes are generated. But we do not know what are the communications among these processes? How could strategy process trigger an operation process? How could a support process complete an operation process? In order to answer these questions, the relations of collaborative levels (RCL) are defined to manage communications among different collaborative processes.

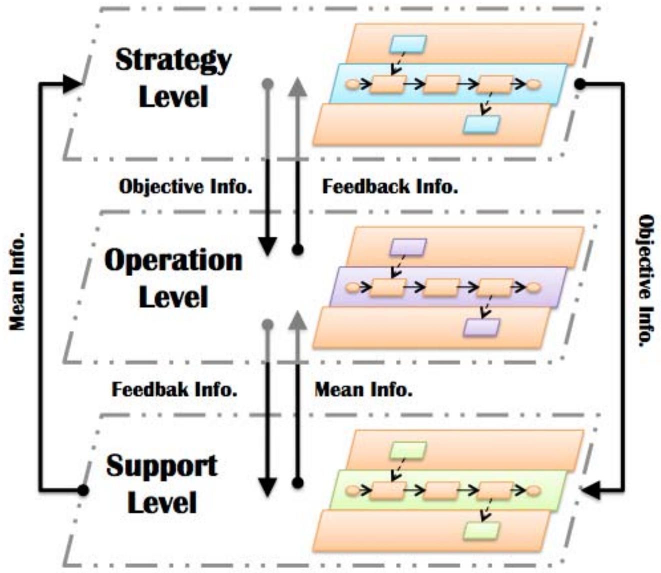


FIGURE II - 13 RELATIONS AMONG COLLABORATIVE SITUATION LEVELS

The communications among strategy level, operation level and support level have been shown in Figure II-13 Among these three levels, three kinds of messages have been involved: objective information, feedback information and mean.

- Objective Information: objective is the goal, which is intended to attain. Objective information is a message, which contains the decision result of strategy level. The objective information could be sent to operation and support level. The operation level and the support level invoke homologous process and useful information to attain the goal in objective information.
- Feedback Information: Feedback information is a message, which contains the operation level result. The feedback information is sent from operation level to strategy level. It is used to report the operational exception, error, result and so on. Feedback information could also be sent from operation level to support level. This kind of feedback information is used to trigger or direct support process.
- Mean: in the collaboration situation, mean is a message, which could contain any kind of information. It could be an exception, an error, a feedback or a signal.

IV. Conclusion

In the collaborative situation framework, there are three dimensions: collaborative situation elements, collaborative situation life cycle steps and collaborative situation levels. In the dimension of collaborative situation element, the elements are gathered by order: organizational, functional, informational and then process. For each element, a model is defined to present the knowledge of the element. In the dimension of collaborative situation lifecycle, the collaborative situation goes through CIM, PIM, PSM and controlling. On the step of controlling, the first lifecycle goes back to the PIM layer. It is designed to solve the failures of technical knowledge. The second lifecycle goes back to the CIM layer. It is designed to correct the mistakes of business knowledge.

In the dimension of collaborative situation level, the collaborative process is classified into three types: strategy process, operation process and support process. The strategy process sends objective information to trigger the operation process and support process. The operation process gives back feedback information to the strategy process. The feedback information of the operation process also triggers support process. The support process sends back mean information to the strategy process and the operation process as feedback message.

MISE 1.0 and MISE 2.0 go through the same collaborative lifecycle (CIM, PIM, PSM and controlling). Both of them contains organizational, functional, informational and process element. They have the same objective, which is to gather collaborative knowledge, to deduce collaborative process, and to develop MIS. But comparing the abstract level of MISE 1.0 to MISE 2.0, what is the main difference? As shown in Figure II-14, the black cubes present the position of MISE 1.0 abstract level in the collaborative situation framework. The abstract level of MISE 1.0 covers all the four elements (organizational, functional, informational and process) and operation level. Figure II-15 presents the position of abstract level of MISE 2.0 in the collaborative situation framework. The abstract level of MISE 2.0 also covers all the four elements. But it also covers strategy level, operation level and support level. The main difference of MISE 1.0 and MISE 2.0 is that: the first one only deduces collaborative operation process, but the second one not only deduces collaborative operation process, but also collaborative strategy process and collaborative support process.

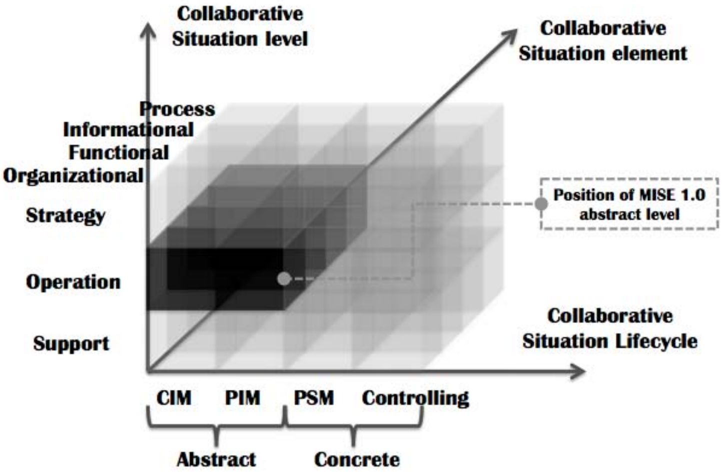

Collaborative Situation level
Collaborative Situation element
Process
Informational
Functional
Organizational
Strategy
Operation
Support
Position of MISE 1.0 abstract level
Collaborative Situation Lifecycle
CIM
PIM
PSM
Controlling
Abstract
Concrete

FIGURE II - 14 POSITION OF MISE 1.0 ABSTRACT LEVEL IN COLLABORATIVE SITUATION FRAMEWORK

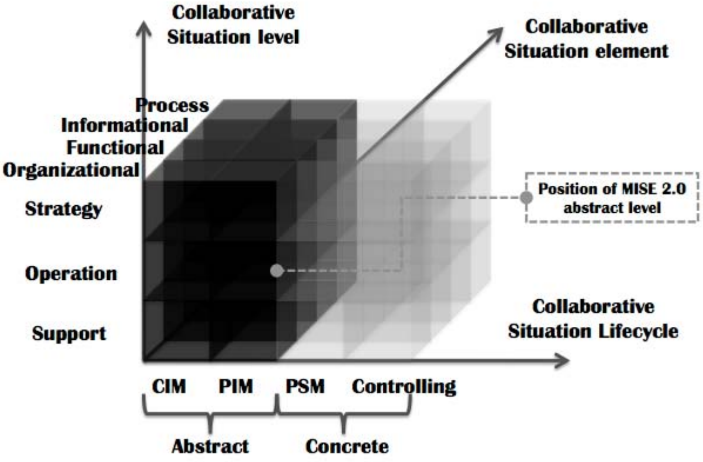

Collaborative Situation level
Collaborative Situation element
Process
Informational
Functional
Organizational
Strategy
Operation
Support
Position of MISE 2.0 abstract level
Collaborative Situation Lifecycle
CIM
PIM
PSM
Controlling
Abstract
Concrete

FIGURE II - 15 POSITION OF MISE 2.0 ABSTRACT LEVEL IN COLLABORATIVE SITUATION FRAMEWORK

Chapter III: Models

I. Introduction 52
II. Collaborative Network Model 54
II.1. Definition of Collaborative Network Model 54
II.2. Example of Collaborative Network Model 59
III. Function Model 60
III.1. Definition of Function Model 60
III.2. Example of Function Model in MISE 2.0 65
IV. Collaborative Process Model 67
IV.1. Definition of Collaborative Process Model 67
IV.2. Example of Collaborative Process Model 70
V. Conclusion 75

I. Introduction

In the most general sense, a model is anything used in any way to represent anything else. Some models are physical objects, for instance, a toy model, which may be assembled, and may even be made to work like the object it represents. A model is a simple description of a system, used for explaining how something works or calculating what might happen. Model could be used in almost all the systems. A conceptual model may only be drawn on paper, described in words, or imagined in the mind. They are used to help us know and understand the subject matter they represent. A business model describes the rational of how an organization creates, delivers, and captures value: economic, social, or other forms of value. The process of business model design is part of business strategy. In theory and practice the term business model is used for a broad range of informal and formal descriptions to represent core aspects of a business, including purpose, offerings, strategies, infrastructure, organizational structures, and operational processes and policies.

The term "enterprise model" is used in industry to represent differing enterprise representations, with no real standardized definition. *"An Enterprise Model is a computational representation of the structure, activities, processes, information, resources, people, behavior, goals and constraints of a business, government, or other enterprise."* (Fox and Gruninger, 1998) From a design perspective, an enterprise model should provide the language used to explicitly define *an enterprise*. From an operations perspective, the enterprise model must be able to supply the information and knowledge necessary to support the operations of *the enterprise*. However, nowadays enterprise modeling has been linked to enterprise integration. Pr. Vernadat believes that *"Enterprise modeling is clearly a pre-requisite to Enterprise Integration while Enterprise Integration is first of all a matter of business process coordination and cooperative decision-making"* (Vernadat, 2002). According to (Pépiot et al., 2007), the goal of enterprise model has been changed:

- Describe the elements of a business entity, a part of a single enterprise, *the whole enterprise or a network of enterprises*. The enterprise model usually takes into account functions, behaviors, information, resources and economic aspects.
- Represent or formalize the structure and behavior of *enterprises*, components and operations in order to understand, to engineer, evaluate, optimize and even control the business organization and operations.

In the knowledge-gathering phase of MISE 2.0 abstract level, we have to define or choose models to collect and present knowledge. The requirements of models are focused on collaborative networks, business functions and semantic annotation. As shown in Figure III-1, the collaborative network model describes the initial collaborative knowledge, which includes the collaborative context and the collaborative objectives. For the functions of partners, the function model is necessary to describe the capabilities of partners. Further more, each function must contain a semantic annotation for further transformation from business process to executable workflow. Then the collaborative process model is defined to represent the behavior of the collaboration. This section answers: what are the models chosen for knowledge gathering and process deducing? Why do we choose these models? And how may we use them?

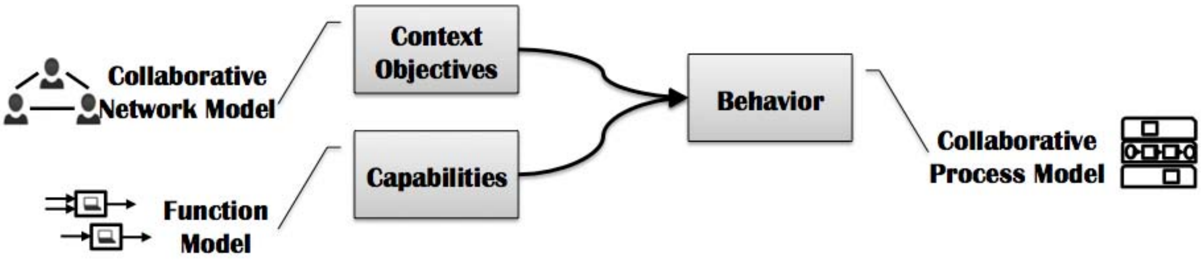


FIGURE III - 1 ORGANIZATION MODEL, FUNCTION MODEL AND COLLABROATIVE PROCESS MODEL

Please pay attention that this chapter just presents how to define the collaborative network model, the function model and the collaborative process model, but the transformation mechanism from the collaborative network model and the function model to the collaborative process model is not explained. The main goal of this chapter is to explain the input models and the output models of the transformation. Figure III-2 shows the position of the chapter III in the UJ picture.

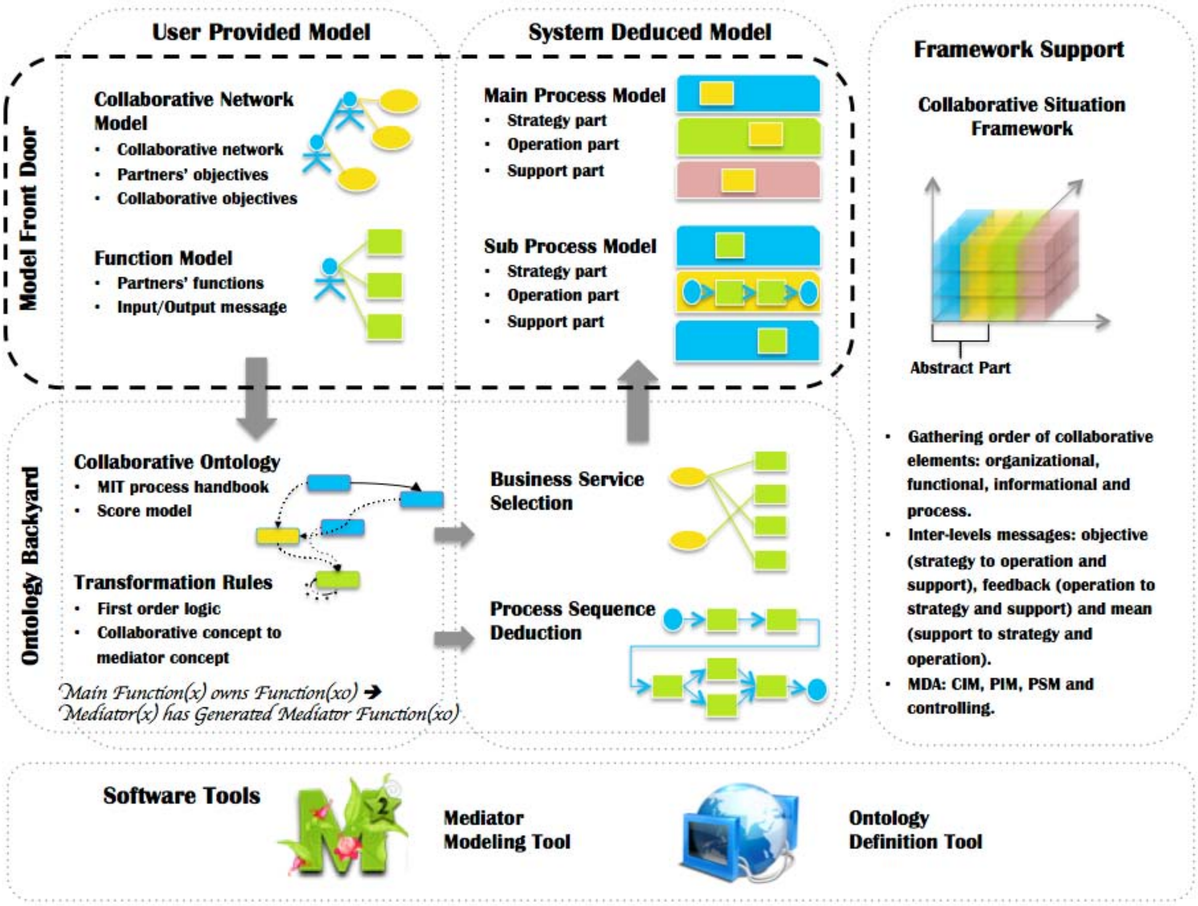

User Provided Model
System Deduced Model
Framework Support
Model Front Door
Collaborative Network Model
Collaborative network
Partners' objectives
Collaborative objectives
Function Model
Partners' functions
Input/Output message
Main Process Model
Strategy part
Operation part
Support part
Sub Process Model
Strategy part
Operation part
Support part
Collaborative Situation Framework
Abstract Part
Ontology Backyard
Collaborative Ontology
MIT process handbook
Score model
Transformation Rules
First order logic
Collaborative concept to mediator concept
Main Function(x) owns Function(xo) →
Mediator(x) has Generated Mediator Function(xo)
Business Service Selection
Process Sequence Deduction
Gathering order of collaborative elements: organizational, functional, informational and process.
Inter-levels messages: objective (strategy to operation and support), feedback (operation to strategy and support) and mean (support to strategy and operation).
MDA: CIM, PIM, PSM and controlling.
Software Tools
Mediator Modeling Tool
Ontology Definition Tool

FIGURE III - 2 POSITION OF CHAPTER III IN UJ

In this chapter, an example, which covers the objective model, the function model and the process model, is used. This example helps to understand the modeling method of the objective model, the function model and the process model.

As shown in Figure III-3, this example has a collaborative network, which has four partners: client, assembler, supplier1 and a group of supplier (supplier 2 and supplier 3). The client buys product from an assembler. The assembler cooperates with suppliers 1, 2 and 3. Among them, supplier 1 is a long term and stable supplier.

Supplier 2 and 3 provide same components. The assembler has to choose one suitable supplier from suppliers 2 and 3.

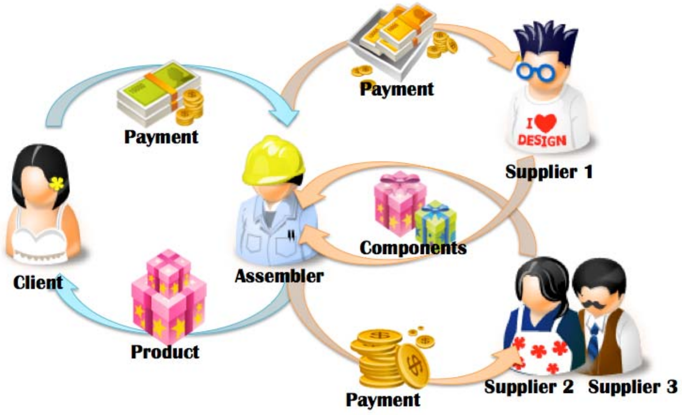


FIGURE III - 3 EXAMPLE OF CHAPTER II

In section II, the collaborative network model is introduced. It is structured according to: i) the state of art of organization models and the definition of the collaborative network model and ii) the defined collaborative network model of the example. In section III, the functional model-IDEF0 based functional main model and sub model are presented. It also follows the same structure as section II: i) the state of the art, of function models and the definition of the function model and ii) the function model based on the example. Finally and similarly, in section IV, the BPMN based collaborative process model is explained. It presents: i) the state of the art of collaborative process models and the definition of our collaborative process model and ii) the collaborative process model of the example.

II.Collaborative Network Model

In this section, the collaborative network model of MISE 2.0 abstract level is presented. The collaborative network model mainly collects the information, which concerns collaborative network, partners, partner relationships, and collaborative objectives. In section II.1, firstly, the definition of organizational model and the definition of organizational model in collaborative situation are addressed. Secondly, the modeling elements and the modeling method of the collaborative network model are introduced. Section II.2 proposed an example of the collaborative network model to better explain the collaborative network model.

II.1. Definition of Collaborative Network Model

The collaborative network model of MISE 2.0 mainly collects and defines the collaborative knowledge, which covers the collaborative network, the partners and the collaborative objectives of partners. There are two key elements of the objective model. First, it is organization, which presents the collaborative network and the partners. Second it is objective, which is the part of the collaborative objectives of partners. Because the collaborative network model defines organizational knowledge, to understand the collaborative network model of MISE 2.0, it obliges to understand what is organization model? And why the objective should be considered in an organization model?

For (Jong and Dietz, 2010), "*an organization can be understood as a social system, i.e. a system whose elements are social individuals or actors. The actors operate in an environment of customers, suppliers, partners and others which share a part of the organization's world.*" According to (Rupietta, 1994), "*organization determines when and how tasks are processed and by whom and which business procedures are automated and in which way.*

Employees of an enterprise, organizational positions, functions, or units, tasks, resources, authorities, procedures, activities, rules … are objects of organization." They define organization from different angles. Jong thought an organization is just the actor or the role involved in a situation. Rupietta believed organization is not only the concept of itself (the actor or the role) but also the components included in an organization (for example, tasks, processes, resources and so on).

In a collaborative situation, (Jiang et al., 2011) considers that: "*Collaborative organizations are involved in the value chain to accomplish not only their own goals but also the cooperative goals.*" In this thesis, an organization is an enterprise, an actor or a role, which is involved in a collaborative situation. The organization may include its own functions and objectives. The organizations try to operate with others and to accomplish their own goals and collaborative objectives in the collaborative situation. (Jiang et al., 2011) defined an inter-organizational collaboration model (IOCM). The model defines organizational model by two fundamental concepts: Role and Organization. **It considers the role as a set of objectives.** The role (objectives) indicates its individual responsibility, i.e., if a role is enacted, its individual responsibility is undertaken. An organization is a set of inter-connected roles. Therefore, Jiang also considered the role (objectives) is an important element in collaborative situation. The objectives are represented and classified through roles and assigned to organization as properties. In an organization model, the objectives are necessary to gather. Figure III-4 provides an example of the modeling.

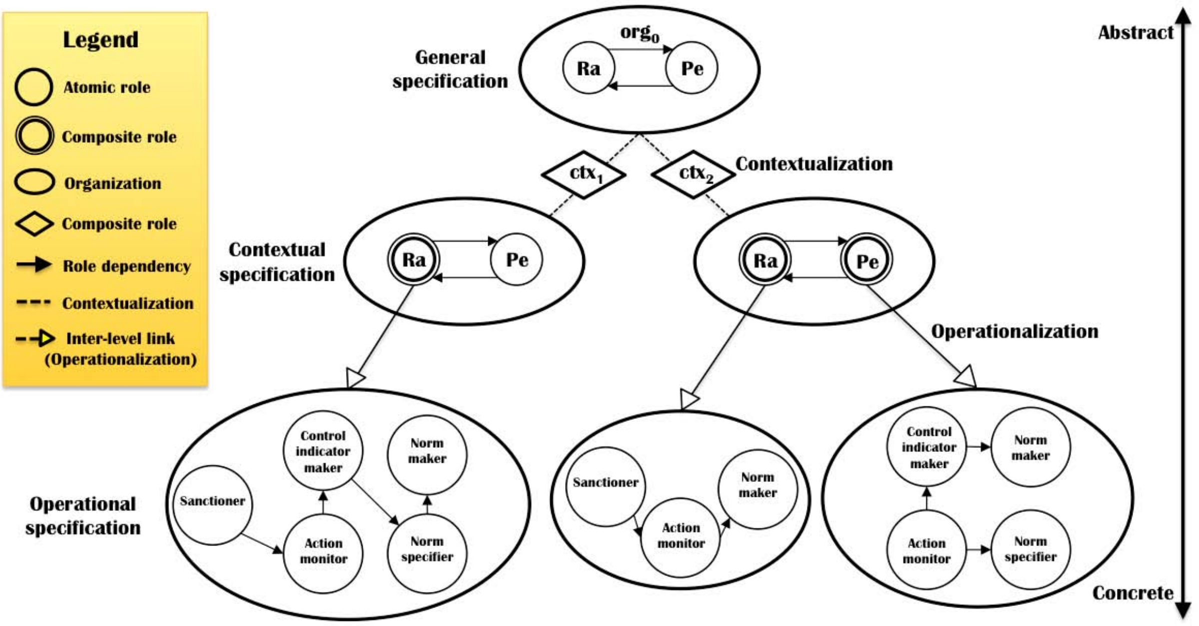


FIGURE III - 4 MODELING PROCESS OF AN EXAMPLE (JIANG ET AL., 2011)

(Bouslimi et al., 2009) defines an organization model in three levels of abstraction (OMOA). The First level, a Role Model (RM) is defined by a set of **Typical Roles[1] and the Interactions Types**, which exist between them. This level of description is the most abstract; it is independent of the task. They have defined the typical roles: the mediator, coordinators, information exchange manager, the matchmaker, user and translators. Figure III-5 represents the global view of typical roles, the interventions and the order of these interventions. At the second level they found Organizational Structures (OS). An OS is a specialization of the Role Model, which defines the structure of an organization specific to a task, which is linked to the defined Role Model. It is defined by a set of specialized roles and the interactions between these roles. At the third level: concrete level,

[1] A typical role is a class of roles defined by Duties and Rights. The Duties are defined by a set of abstract Actions, a set of Interventions, their coordination rules, and the Invariants. The abstract Actions are the internal actions that the typical roles can perform without interacting with other roles. To preserve the autonomy of the agents, which play a role, the local actions are represented in an abstract way by giving their name but without referring to their implementation.

they found Concrete Organizations (or more simply an Organization). An Organization is an instance of an Organizational Structure where agents play roles and interact for solving in cooperation a task instance.

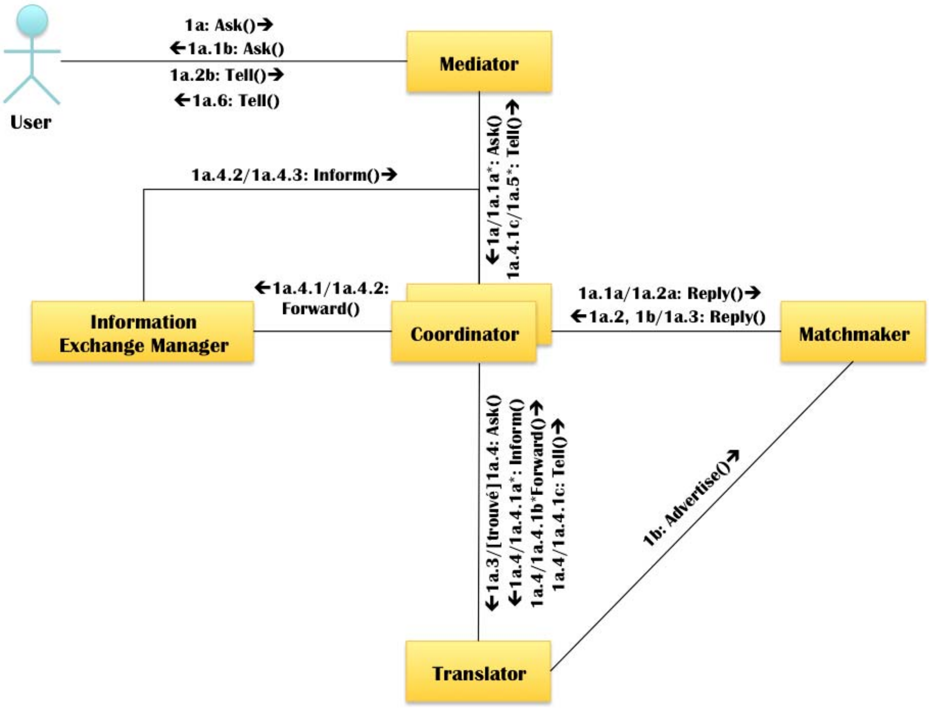


Figure III - 5 Communication diagram between typical roles (Bouslimi et al., 2009)

(Rajsiri et al., 2010) defines an organizational model to collect the collaborative knowledge (OMCK). It aims at facilitating users to collect and formalize knowledge about collaboration. The organization model of Dr. Rajsiri defines elements: network, participants, abstract service, topology, role and common goals. A network is composed of several participants, topologies, and common goals. Each participant is composed of several services, and roles. Each role is related to a set of services to show the capabilities of the role. Topology contains the relationship, which links two participants together at the role level.

The IOCM and the OMCK are the organization models in collaborative situation. The main idea of IOCM is to model both the collaborative objective and individual goals of organization. The IOCM focus on the modeling of roles (a set of objectives). It describes the organization as inter-connected roles, which is network. The two main concepts of OMCK are the network and the common goals. It considers the collaborative situation as network of collaboration. It defines the common goals of the whole network. For OMOA, the strong point is that it considers the organization as three levels of abstraction. It first defines the role, which focus on task (here the task could be considered as objectives of roles of IOCM or the abstract service or common goals of OMCK). And then it defines the structured organizations (the abstraction of the network of OMCK), which could complete the task. Finally, it defines the instances of organizations, which realize the structured organizations. Table III-1 provides a summary of IOCM, OMOA and OMCK. To summarize, for an organization model in collaborative situation, there are three fundamental concepts: network, objective and decomposition of network (sub networks).

TABLE III - 1 SUMMARY OF IOCM, OMOA AND OMCK

	IOCM (Jiang et al., 2011)	OMOA (Bouslimi et al., 2009)	OMCK(Rajsiri et al., 2010)
Network	Yes, the network is defined by roles through three level of abstraction	Yes, the network is defined with 6 typical roles by using Petri-net	Yes, the network is defined by partners and roles of partners and completed by concepts of topology
Collaborative objective	Yes, the role is considered as a set of objectives	No	Yes, the objectives are defined through common goals of collaborative situations
Relations of Partners	Yes, the relations are defined by using role dependency	Yes, the relations are defined by interactions among roles	Yes, there are three kinds of relations: competition, group of interest and customer-supplier
Phases of Modeling process	Yes, there are three levels (from abstract to concrete): general specification, contextual specification and operational specification	Yes, there are three levels: role model, organizational structures and concrete organizations	No

Organization modeling is not a new subject in enterprise modeling. But most of the organizational models only define the tree structure of enterprises, responsibilities, departments and workers. In a collaborative situation, the structure is graph (discrete mathematics) rather than a tree. Based on the concept of topology (Kelley, 1975), we decide to define our own organization model – collaborative network model.

The collaborative network model is an objective-oriented organizational model. This model is defined to gather: (i) collaborative network partners and partners' relationships and (ii) objectives of main network, sub-network and partners. "An objective model is required to facilitate: (i) identification, communication and structuring of business objectives, and (ii) measurement of the level of success in achieving objectives. But individual modeling methodologies focus primarily on selected aspects of objectives representation and measurement. (Neiger et al., 2009) " Rajsiri has proposed a definition of collaborative network model, which models collaborative network and collaborative main goal (Rajsiri et al., 2010).

But for our individual needs, the collaborative network model here should collect the collaborative main goals. For each collaborative goal, partners are regrouped in a sub-collaborative network. Partners also have their own objectives. We come up with a result: a real collaborative situation is like a multi-level pyramid: each level could be decomposed from the whole collaborative network into several sub-networks until the end-nodes: partners.

As shown in the left part of Figure III-6, there are four main elements in the collaborative network model: partner, collaborative network, objective and relationship. In the right part, the table defines the possible connections among different elements. The explanation of each element is listed as follows:

- A collaborative network could either represent the whole collaborative network, (which is made by all the partners) or represent a sub-network (a part of the whole network, which contains several partners). The whole network contains the main objectives. A sub-network could implement the objective.
- Partner means organizations, persons, enterprises, etc. which are involved in a collaborative situation.
- Objective means a goal, which is a desired result of a partner, a collaborative network or a part of the network. Objective is a plan and commitments to achieve - a personal or organizational desired end-point in some sort of assumed development in the collaborative situation. Objective is classified into three types: Strategy Objective, Operation Objective and Support Objective ((ISO 9000 X50-130, 2005) and (ISO 9000, 2005) explain that business processes contain strategy, operation and support

processes. Our goal is to deduce business processes, so we separate objectives into strategy, operation and support objectives).

- Relationship contains two parts: Objective Relationship and Partner Relationship (Strategy Relationship, Operation Relationship and Support Relationship). If Partner or Collaborative Network has Objective, then Objective Relationship is created between them. If one Partner co-works with another Partner under the same Strategy Objective, then the two Partners owns a Strategy Relationship (same as Operation Relationship and Support Relationship).

The right part of Figure III-6 shows the relationships, which are used among modeling elements. Partner, Collaborative Network, Objective, Operation Objective, Strategy Objective and Support Objective fill the first line and first column. Objective Relationship, Operation Relationship, Strategy Relationship and Support Relationship fill other cells of the matrix, if the modeling element of correspondence first column could be linked to first line modeling element. For example, Operation Relationship, Strategy Relationship or Support Relationship may relate Partner in the first column to Partner in the first line.

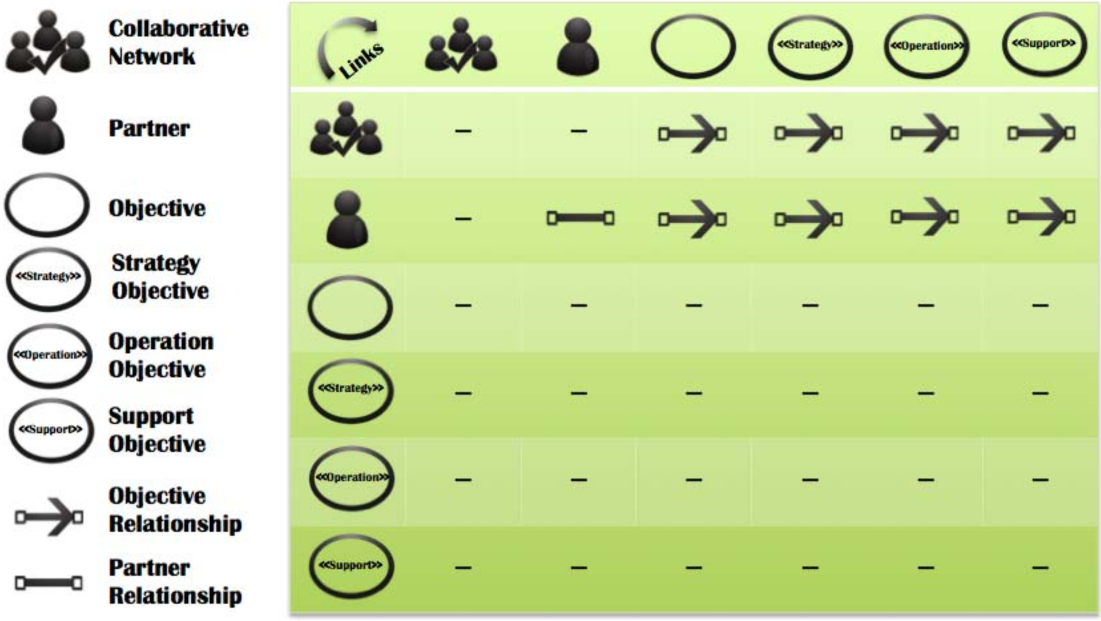


FIGURE III - 6 THE DEFINITION OF THE ORGANIZATION MODEL

Organizational model definition rules are summarized as follows (using the collaborative network model as an example):

- The first step in building a collaborative network model is to define objectives of the whole collaborative network. A collaborative network could have general objectives of three types: strategy, operation and support.
- A sub-network implements a general objective. The general objective also may contain small goals. So a sub-network could have several small goals/objectives.
- For a strategy objective, an operation objective or a support objective, the objective could be implemented by a set of partners. All the objectives of partners must be of the same type as the objective (strategy, operation or support). Because they have to keep with the same type of the objective in higher level.
- A partner could have a relationship (strategy, operation and support) with other partners. A partner could only have strategy objectives, operation objectives or support objectives (not general objectives).

To explain these model definition rules, a collaborative network model example is provided in section II.2.

II.2. Example of Collaborative Network Model

To illustrate all the concepts involved by, let's define one for example. As shown in Figure III-7 and Figure III-8, the collaborative network model has four levels. The first level is to define the main objectives of the whole network. The main objectives could be strategy objectives, operation objectives, support objectives or objectives without specific type. Considering the network presented in Figure III-3, the first level of the collaborative network model in Figure III-7, *0 network*, defines strategy objective: *choose partner*, objective: *sell product* and operation objective: *sell component*. Each objective has a sub-network to achieve the objective.

In level 2, sub networks complete all the objectives, which are defined in level 1. For the strategy objective: *choose partner*, because it is defined as strategy objective, the sub network of this objective is created to achieve it. It leads to that the collaborative objectives of this sub network are also strategy objective. For the operation objective: *sell component*, because it is defined as operation objective, the sub network of this objective is made to reach this objective. The collaborative objectives of this sub network are operation objectives. For the objective: *sell product*, the objective has not been sorted, the sub-network could have different kinds of objectives.

As shown in Figure III-7, in the second level, *choose partner* and *sell component* have been enlarged into a sub-network which contains partners already. But for *sell product*, the objective of the sub-network has been sorted into three different kinds of objectives: *place order*, *deliver product* and *pay product*. With itemized objectives, these objectives could be enlarged directly by a sub-network, which is composed of partners. For these sub-networks, objective types must be the same as for objectives in the previous level. In level 3, objectives: *place order*, *deliver product* and *pay product* have been decomposed by sub network, which contains partners: *assembler* and *client*. The decomposition of collaborative network has reached the end-node: partner. Level 3 would be the last level of collaboration in the pyramid of collaborative network.

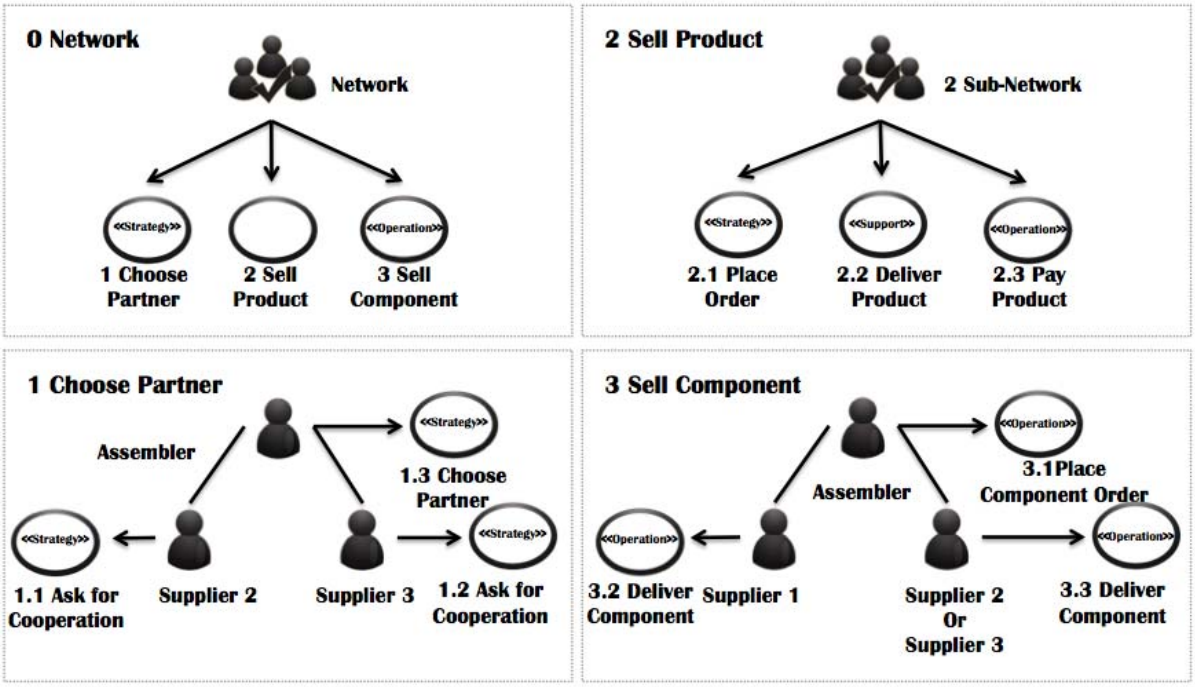


FIGURE III - 7 LEVEL 1 AND LEVEL 2 OF THE COLLABORATIVE NETWORK MODEL

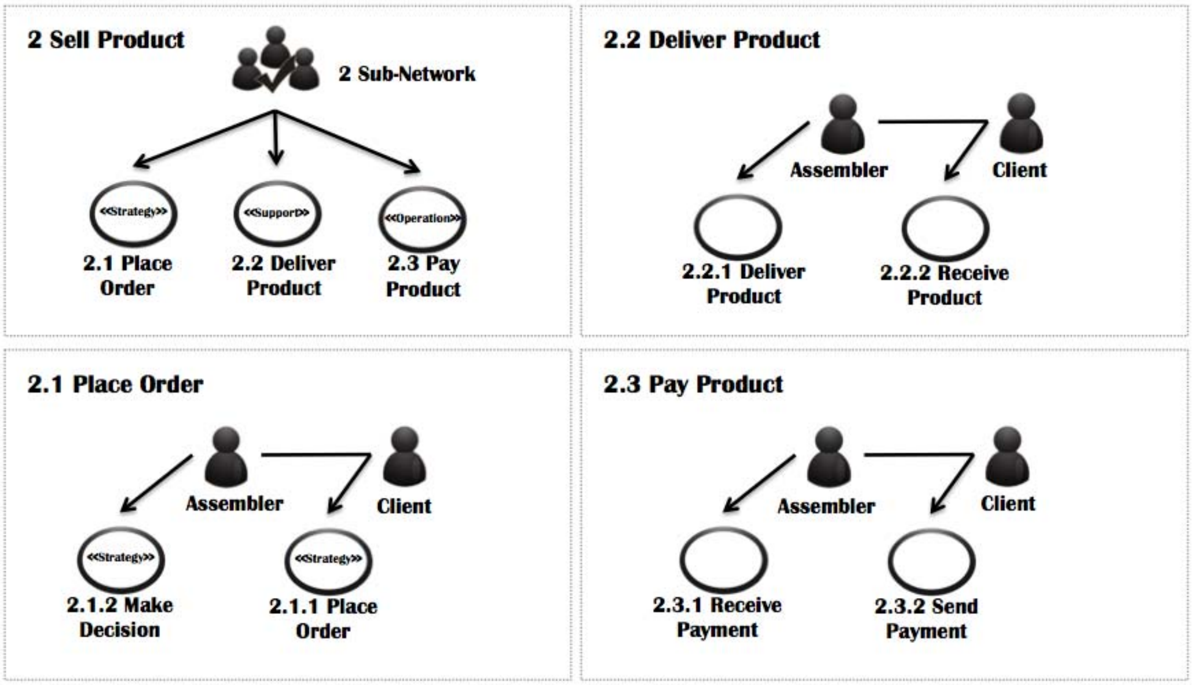


FIGURE III - 8 LEVEL 2 AND LEVEL 3 OF THE COLLABORATIVE NETWORK MODEL

III. Function Model

In this section, the IDEF0 (Integration Definition for Function Modeling) based function model of MISE 2.0 abstract level is presented. The main goal of function model is to gather the information of shared functions of partners in a way, which is simple and easy. In section III.1, firstly, the definition of function model and the IDEF0 modeling method are addressed. Secondly, the modeling elements and the modeling method of the function of MISE 2.0 are introduced. Section III.2 proposed an example of the function model.

III.1. Definition of Function Model

A function model or functional model in systems engineering and software engineering is a structured representation of the functions (activates, actions, processes, operation) within the modeled system or subject area. For (Komoto et al., 2008), "*function modeling is the name given to the activity of developing models of devices, products, objects, and processes based on their functionalities and the functionalities of their subcomponents.*" In (PUBS, 1993), "*a function model, also called an activity model or process model, is a graphical representation of an enterprise's function within a defined scope. The purposes of the function model are to describe the functions and processes, assist with discovery of information needs, help identify opportunities, and establish a basis for determining product and service costs.*" In this thesis, a function model in a collaborative situation is a graphical representation of shared functions of partners. The shared function must include the information about requirement (input) and result (output). The sufficient condition of the function model is to correctly present the function and in/out information of the function. Based on this need, a set of function models (activity models or process models) is studied (e.g. IDEF0, UML activity diagram, flow chart and EPC).

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of object-oriented software engineering. UML activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In UML, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. The activity diagram is similar with flow chart. Both of them provide perfect flows among functions, but without input and output messages, which does not fit the need. UML activity diagram and flow chart are removed from the list. Event Process Chain (EPC) is a type of flowchart used for business process modeling. The EPC method was developed within the framework of ARIS. The elements of EPC include: event, function, process owner, organization unit, input, output, supporting system and result event. But the EPC is an "event-oriented" model. In this thesis a "flow-oriented" model is needed. Furthermore, the main objective of knowledge gathering phase is to collect information in a way as simple as possible. The EPC is removed from the candidate list.

IDEF0 (Integration Definition for Function Modeling) is a function modeling methodology for describing manufacturing functions, which offers a functional modeling language for the analysis, development, reengineering, and integration of information systems; business processes; or software engineering analysis. (Lightsey, 2001) According to (Li et al., 2009), IDEF0 is a modeling method including combined graphics and text to obtain understanding, support analysis, provide logic for system adjustment, specify requirements, or support systems level design and integration. IDEF 0 is part of IDEF family of modeling languages. As introduced in (IDEF0, 1993), IDEF0 is based on SADT (Structured Analysis and Design Technique TM), developed by Douglas T. Ross and SofTech, Inc. IDEF0 model includes a hierarchical series of diagrams, text and glossary cross-referenced to each other. Two primary modeling components of this method are functions and data that inter-relate those functions. Due to IDEF0 defines function with data in/out, it fits the need of function model of this thesis. The modeling unit of IDEF 0 is not difficult to understand and use. To summarize, the IDEF0 is chosen as function model in this thesis. To adapt the modeling needs of IDEF0, it still needs to be changed and simplified (section III.2).

The IDEF0 definition of a function is "*a set of activities that takes certain inputs and, by means of some mechanism, and subject to certain controls, transforms the inputs into outputs*." (PUBS, 1993) According to (PUBS, 1993; Abt et al., 2009), the IDEF0 language semantic is then based on six major concepts:

- Functions are the functionalities of the system.
- Inputs are elements to be processed by the activity (e.g., files, documents, raw materials, products).
- Controls are elements like laws, policies, standards, and unchangeable facts of the environment. They control, direct, or force the execution of the activity but are not modified by it.
- Outputs are elements produced or modified by the activity (e.g., data, materials, products).
- Mechanisms are means to execute the activity. They are resources (human or material) that are used in bringing about the intended goals of the activity.
- Calls arrow the sharing of detail between models (linking them together) or between portions of the same model. The "called" function provides detail for the "caller" box.

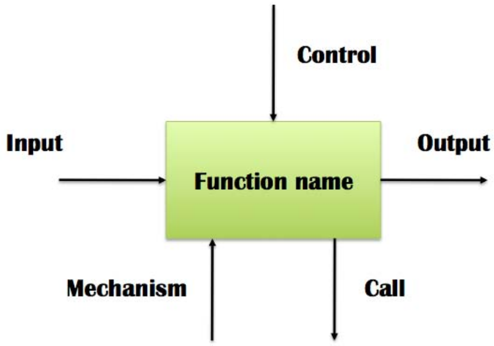


FIGURE III - 9 MODELING UNIT OF IDEF0 (PUBS, 1993)

The IDEF0 language syntax is based on boxes and arrow segments. Boxes represent functions. Each box has a label: an active verb or verb phrase that describes the function. Arrows represent the inputs, controls, outputs, mechanisms and calls. The arrows may be separated into branches. The branches may represent either the same message or portions of the same message. Box and arrow segments are combined in various ways to form diagrams. The boxes in a diagram are connected by sequences of arrow segments. IDEF0 models are hierarchically arranged IDEF0 diagrams (Figure III-10). Unlike every other diagram in the model, the top-level diagram (context diagram, numbered A-0) contains only one box. This box represents, at the coarsest granularity, the single high-level activity that is being represented and decomposed in the IDEF0 diagrams. The parent–child relation holding between two diagrams signifies that the child node is the decomposition of a box in a parent node. A decomposition of a box is a diagram that represents a finer-grained view of the function. Diagrams are numbered. This hierarchical decomposition results in both wide-scope and detailed representations of system activities.

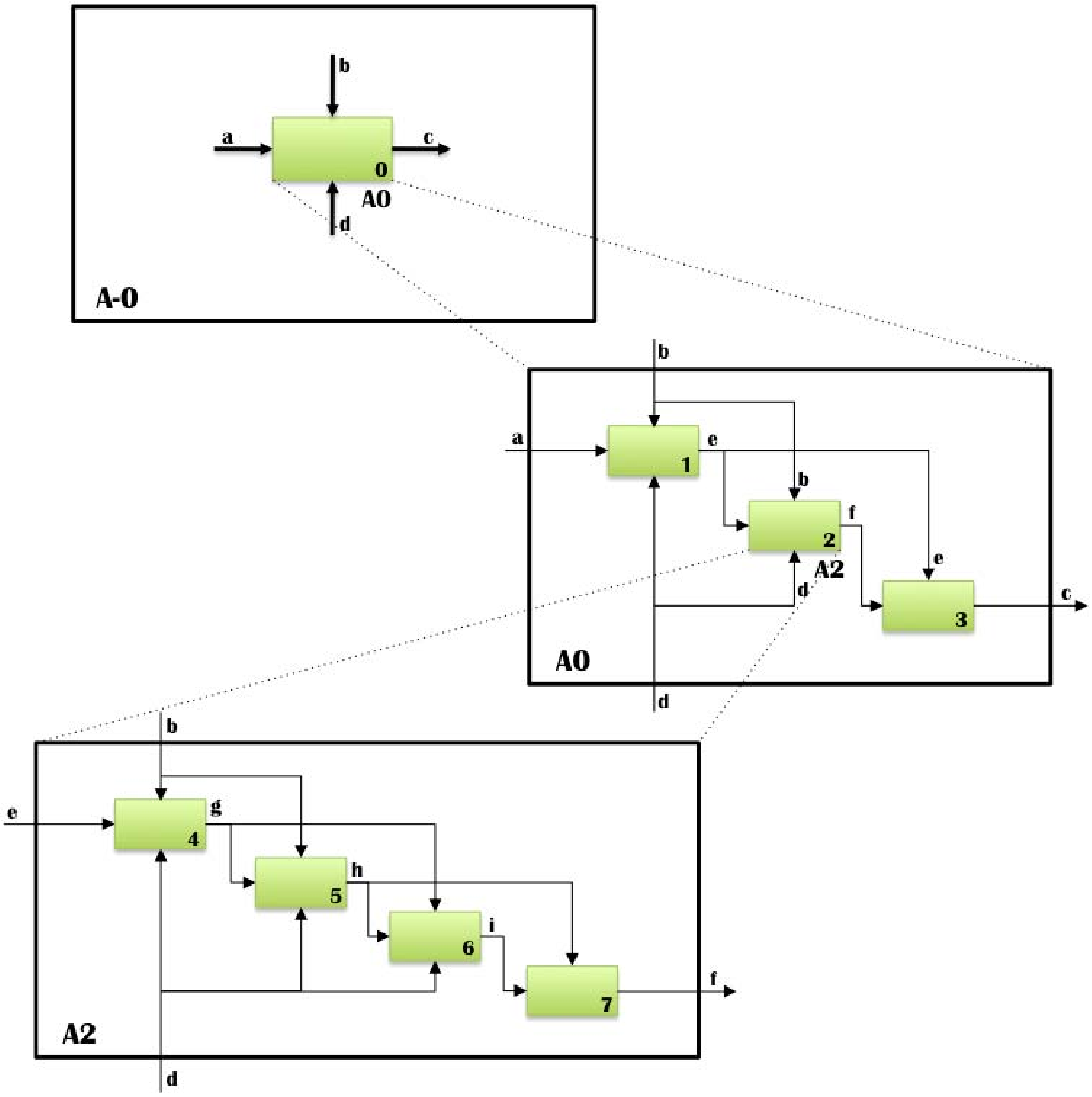


FIGURE III - 10 DECOMPOSITION OF FUNCTION (PUBS, 1993)

In MISE 2.0 functional model, IDEF0 is reused in two ways: main function model (the left hand part of Figure III-11) and function model (the right hand part of Figure III-11).

Main function model mainly presents main functions and the messages transferred among main functions (the messages has three types: objective information, feedback information and mean information, which have been presented in chapter II, section III.4 in the whole collaborative network or sub-network. In main function model, the box, control message in IDEF0 are reused. The box presents main function. Control message presents the message: objective information, feedback information and mean information. Because the main functions in the main function model are transferred from the collaborative objectives, which is defined in the collaborative network model. The collaborative objectives are classified to three types: strategy, operation and support, which are the three collaborative levels in the collaborative situation framework. This means that the messages exchanged among main functions are the messages, which is exchanges among different levels of collaboration. According to the section III.4 of chapter II, among different collaborative levels, there are different messages to trigger the collaborative processes in each level. So the messages of the main functions are defined as control message, neither input/output nor mechanism or call.

The function model is defined to gather the shared functions of partners. Each partner has to define the functions by using the box and the input/output/control arrows.

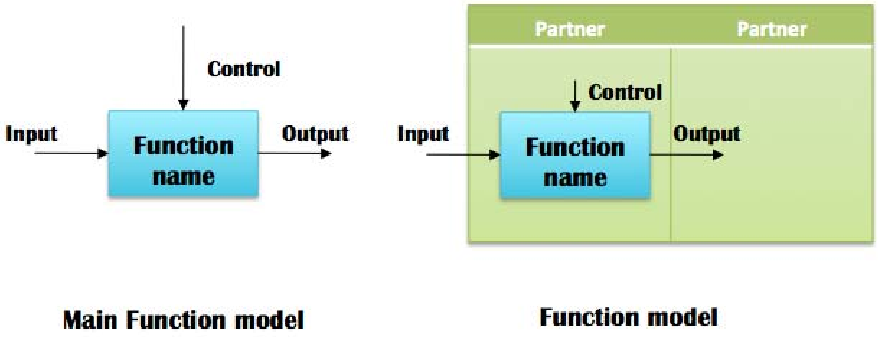


FIGURE III - 11 MODELING UNIT OF FUNCTION MODEL

After analysis and evaluation, we found out that the main function model could be partially transferred from objective model. User only fulfills control messages among main functions. Transformation equations from objective model to main function model are defined in first-order logic (Smullyan, 1995). Due to particularity of transformation rules, first order logic still needs to be expended as followed:

- Class: X is collaborative network → collaborative network(X)
- Association: Y is association implement which is between collaborative network X1 and objective X2 → implement(Y) (collaborative network(X1), objective(X2))
- If-then-else: if (X) → then (Y), else if (X1) → then (Y1), else → then (Y2)
- A set of variables: from X1, X2, X3 to Xn → X1 … Xn

TABLE III - 2 TRANSFORMATION RULE 1: OBJECTIVE TO FUNCTION

Rule 1: Main Collaborative Objectives → Main Function	
$\forall \text{CollaborativeNetwork}(X)\ (\forall \text{ObjectiveRelationship}(\text{CollaborativeNetwork}(X), \text{Objective}(X_1)))$	**(1)**
$\rightarrow \exists \text{MainFunctionModel}(X) \wedge \exists \text{MainFunction}(X_1) \in \text{MainFunctionModel}(X)$	

The transformation rule (1) in Table III-2 defines how the objective is transferred to main function. If there exits a collaborative network, and the collaborative network have one collaborative objective, then there exists one main function model, this main function model has one main function, which has the same name as the collaborative objective. The results of transformation are shown in Figure III-12. The example of collaborative network model presented in section II (Figure III-3) is reused to show the results. At the left side of Figure III-8, the models, with collaborative networks, are considered. They are *0 network* and *2 sell product*. The main function models are created for them. The objectives, which are defined in *0 network* and *2 sell product*, are transferred to the main functions in the main function models.

To summarize the transformation rules (1), there are three tasks: i) find the collaborative network model with main network or sub network, ii) create the main function model for the collaborative network model and iii) transfer the objectives of the network to the main functions of the main function model.

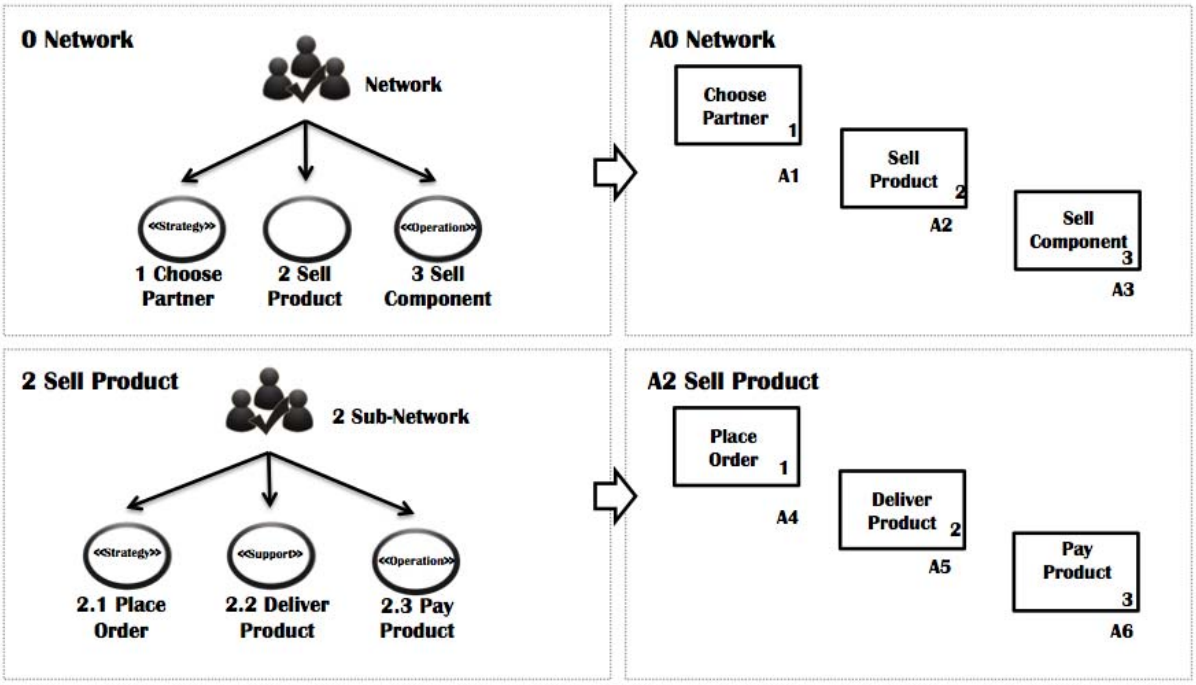


FIGURE III - 12 RESULTS OF TRANSFORMATION RULE (1)

After the transformation of main function model, users have to add control message for the main functions. As shown in Figure III-13, the *A0 Network* is completed. The control messages are added to the main functional model. The function *Choose Partner* triggers *Sell Product* by sending a control message named *Wait for order trigger*. If *Wait for order trigger* equals to true, it means that a partner has been chosen form Supplier 2 and Supplier 3, the whole collaborative network has been settled and the assembler could start to take order from client. In order to complete the *Sell Product*, a message is needed: *Component sold feedback*. If *Component sold feedback* equal to true, it means that all needed components have been bought, the assembler could start to assembly products and deliver them. These controlling messages are added manually. Combining with the business service selection in chapter IV, the controlling messages may be added automatically. For now in MISE 2.0, the controlling messages are added manually.

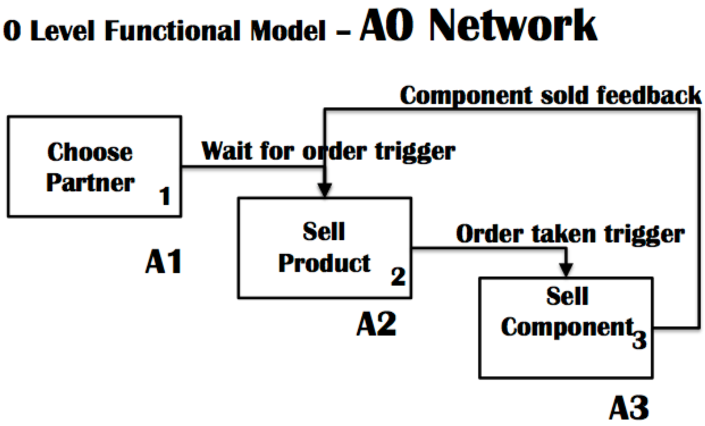


FIGURE III - 13 COMPLETED A0: NETWORK

Figure III-14 shows the completed *A2 Sell Product*. The function *Place Order* launches with the control message: *Wait for order trigger*. If the main function: *Place Order* is proceeded, then an output message: *Order taken trigger* is sent to the function: *Sell Component* in *A0 Network*. The function: *Deliver Product* is triggered by the control message: *Component sold feedback*. If the function is finished successfully, the output message: *Product delivered feedback* is sent to the function: *Pay Product* to trigger the function: *Pay Product*.

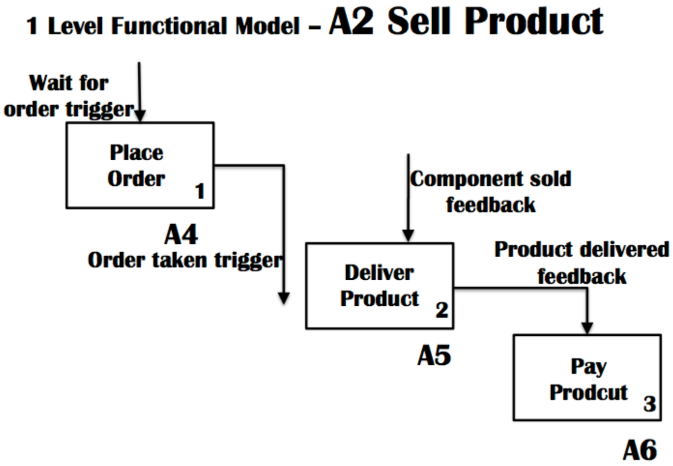


FIGURE III - 14 COMPLETED A2: SELL PRODUCT

To represent the main function model, in this section, users manually complete the control messages of the main function models. In chapter IV, combining with the business service selection, the controlling messages may be added automatically. The control messages can be added automatically by studying the control messages of selected partner functions. For now in MISE 2.0, the controlling messages are added manually. In next section, the function model of example is presented.

III.2. Example of Function Model in MISE 2.0

The function model in MISE 2.0 is an IDEF0 based functional model. IDEF 0 has been reused in two styles: main function model and function model. The organization model is made of network elements and objective elements (e.g. 0 Network and 2 Sell Product in Figure III-12) and reused to initialize the main functional model. The main function model is a kind of middleware, which separates the functions by collaborative objectives. But the shared functions of partners are not presented. In this section, we will reuse the example and build the function model by using the modeling unit, which have been represented at the right side of Figure III-11.

Figure III-15 and Figure III-16 represent the function model of example. The modeling method is really simple. The partners just list their shared business function with function name and the input/output/control messages. For example, in Figure III-15, the assembler provides the function: *Receive application report's requirement* with the input message: *Application report's requirement*, the function: *Send decision* with the output messages: *Partner chosen decision* and *Wait for order trigger*, the function: *Wait for order* with the control message: *Wait for order trigger* and the output message: *Order* and so on. In chapter V, the Mediator modeling 2ool is presented. The user can define the function model by using the Mediator modeling 2ool.

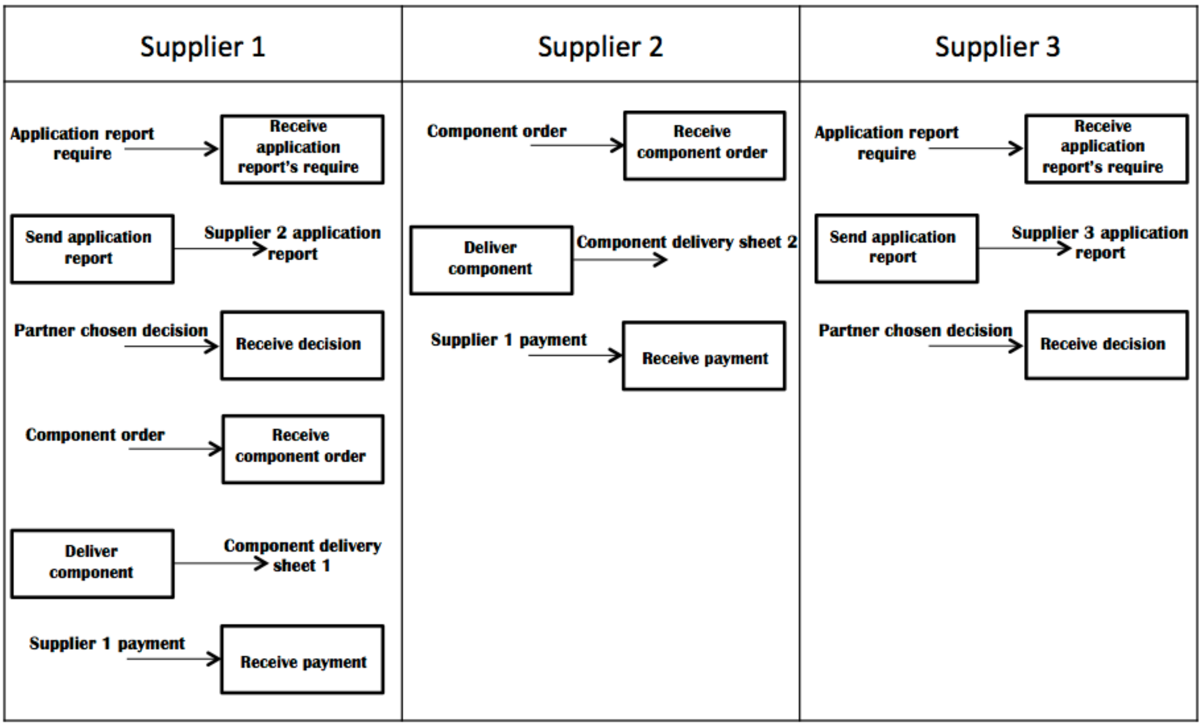


FIGURE III - 15 FUNCTION MODELS OF SUPPLIER 1, 2 AND 3

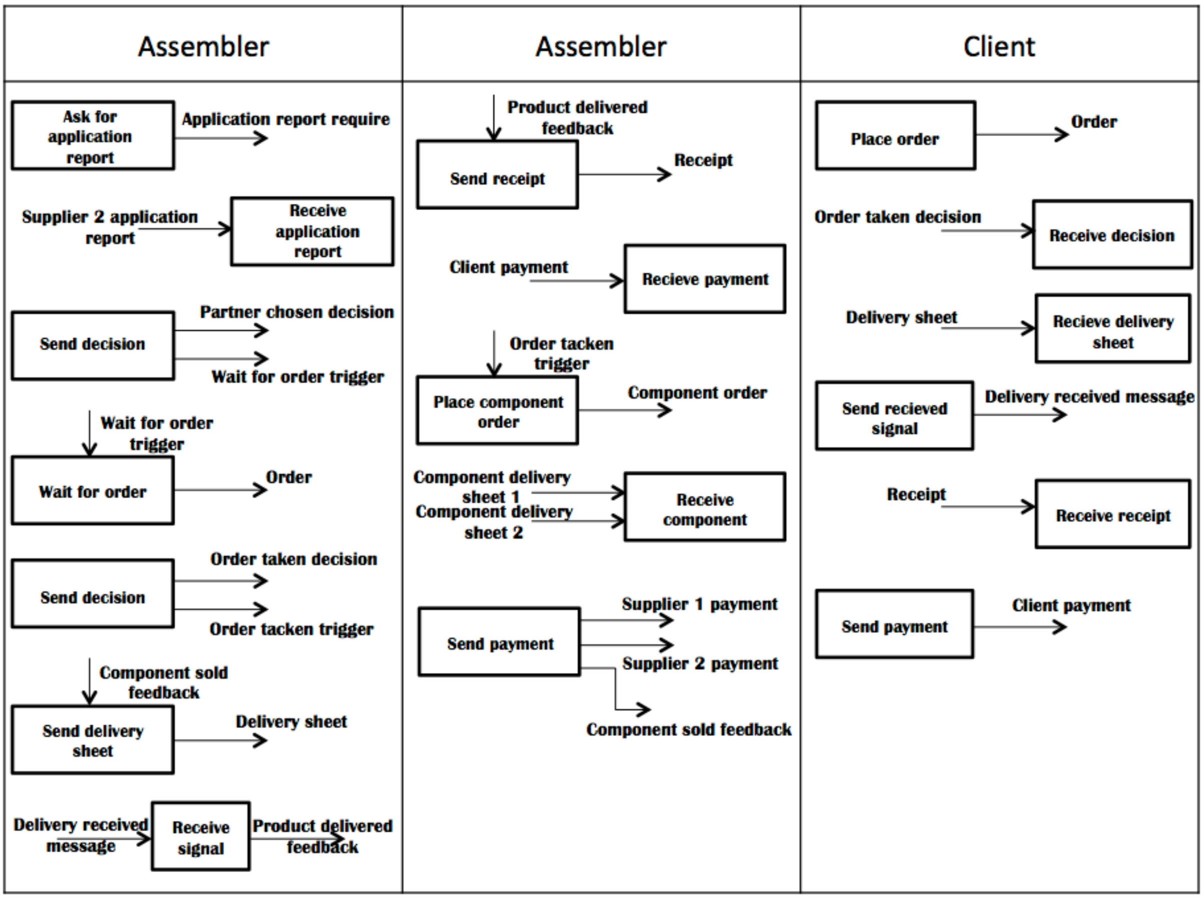


FIGURE III - 16 FUNCTION MODELS OF ASSEMBLER AND CLIENT

IV. Collaborative Process Model

In this section, the deduction of the BPMN (Business Process Model Notation) collaborative processes of MISE 2.0 abstract level is presented. The main goal of collaborative processes is to provide the process cartography (contains strategy, operation and support types) and to represent the collaborative behavior in detail. In section IV.1, firstly, the definition of collaborative process model of MISE 1.0 is addressed. Secondly, the collaborative process cartography and the collaborative process of MISE 2.0 are introduced. Section IV.2 proposes one example of collaborative processes.

IV.1. Definition of Collaborative Process Model

In the process modeling domain, a number of models have been defined, for example, flow charts, Petri nets, Event Process Chains of ARIS, activity diagrams of UML and more recently BPMN. But as mentioned by Touzi (Touzi, 2007), using the advanced formalisms to model a process can cover several aspects of processes including actors (organizational view) and information (informational view). Furthermore, one of the objectives of MISE 2.0 is to derive a BPEL file, which is deployed on the ESB to execute the technical process. Both (Truptil, 2011) and (White, 2005) introduced methods to translate BPMN models into BPEL. They both agree that with the goal of BPEL derivation, BPMN is an effective way to model business processes. Rajsiri et al. even provided a BPMN based Collaborative process model with several partner pools and one mediator pool (Rajsiri et al., 2010). BPMN (Business Process Model and Notation) 2.0 (OMG, 2011) has been developed by OMG (Object Management Group) in 2010. In BPMN 2.0, the tasks are classified as service tasks, send tasks, receive tasks, user tasks, manual tasks, business rule tasks etc. MISE 2.0 aims to deduce BPEL file and select web services with the help of semantic annotations. So BPMN 2.0 becomes a good choice to express collaborative process models.

The use of so-caffed BPMN based collaborative process, as introduced in (Rajsiri et al., 2010) is preferred. As shown in Figure III-17, the collaborative process model has one mediator pool and several partner pools. The mediator pool could be a strategy mediator, operation mediator or support mediator. Different mediators can communicate through data objects, which are linked through event messages. Tasks in mediator pools invoke tasks in the partner pool according to a defined collaborative process. Consequently, the target metamodel is not BPMN (or BPMN 2.0) itself, but the specific collaborative process metamodel presented in (Touzi et al., 2009), (Rajsiri et al., 2010) and (Touzi, 2007). This is a very important point because translating classical BPMN models directly into BPEL files is not always feasible. Börger has clearly addressed "*an unmediated gap between conceptual and executable BPMN model (in particular if obtained through compilation to more detailed languages like BPEL or to code)*" (Börger, 2011). The considered target metamodel has been built especially on this purpose: restricting BPMN expressivity to a sub-space, specifically dedicated to a collaborative situation and that could be translatable in BPEL.

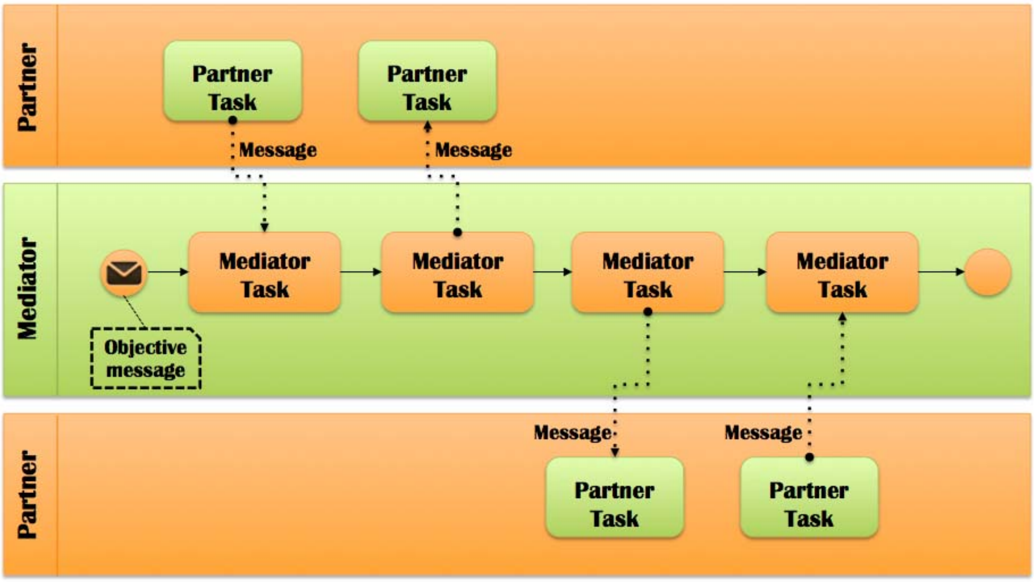


FIGURE III - 17 COLLABORATIVE PROCESS MODEL OF MISE 1.0

To build the collaborative process cartography, the collaborative process model in Figure III-17 stills needs to be expanded. Figure III-18 shows the expanded collaborative process model, which is the collaborative process cartography. The collaborative process cartography can be seen as the main collaborative process model. It separates the collaborative process by strategy pool, operation pool, support pool and general pool. If one main function is strategy function, then it is in the strategy pool. If one main function is operation function, then it is in the operation pool. If one main function is support function, then it is in the support pool. If one main function includes two of strategy, operation or support function or all of them, the main function is in the general pool. The main functions of different pools communicate with objective message, feedback message and mean message. These messages have been introduced in chapter II section III.4. A strategy task in strategy pool sends objective message to operation pool and support pool. It receives feedback message from operation pool and mean message from support pool. An operation task in operation pool sends feedback message to strategy pool and support pool. It receives objective message from strategy pool and mean message from support pool. A support task in support pool sends mean message to strategy pool and operation pool. It receives objective message from strategy pool and feedback message from operation pool. A general task in general pool could send all three kinds of messages and receive all three kinds of messages. As the process cartography, all the functions of the process cartography can be decomposed to sub collaborative process or another process cartography. (Figure III-18) The sub collaborative process likes the collaborative process, which is presented in Figure III-17. It contains one mediator pool and several partner pools.

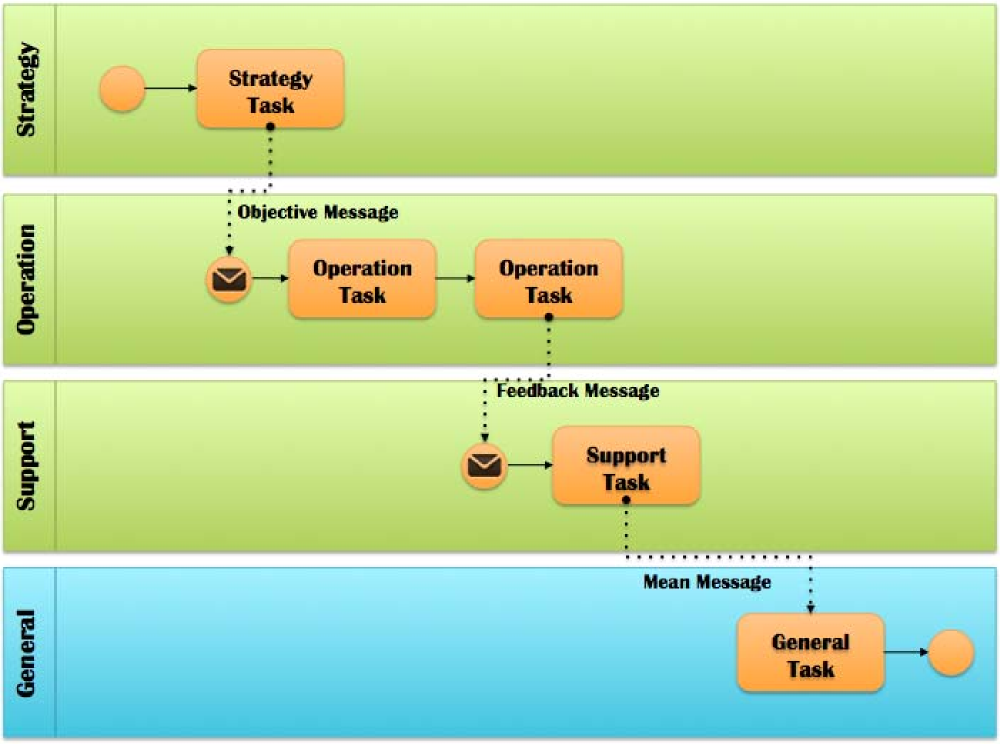


FIGURE III - 18 ADDITIONAL PROCESS CARTOGRAPHY OF MISE 2.0

In Figure III-19, the collaborative process can be decomposed to another collaborative process cartography or the sub collaborative process model. The tasks in strategy, operation and support pool can be decomposed to the sub collaborative process model. But the task in general pool can only be decomposed to another collaborative process cartography until the decomposed process cartography does not include any general function in general pool. Next section illustrates all these principles through an example.

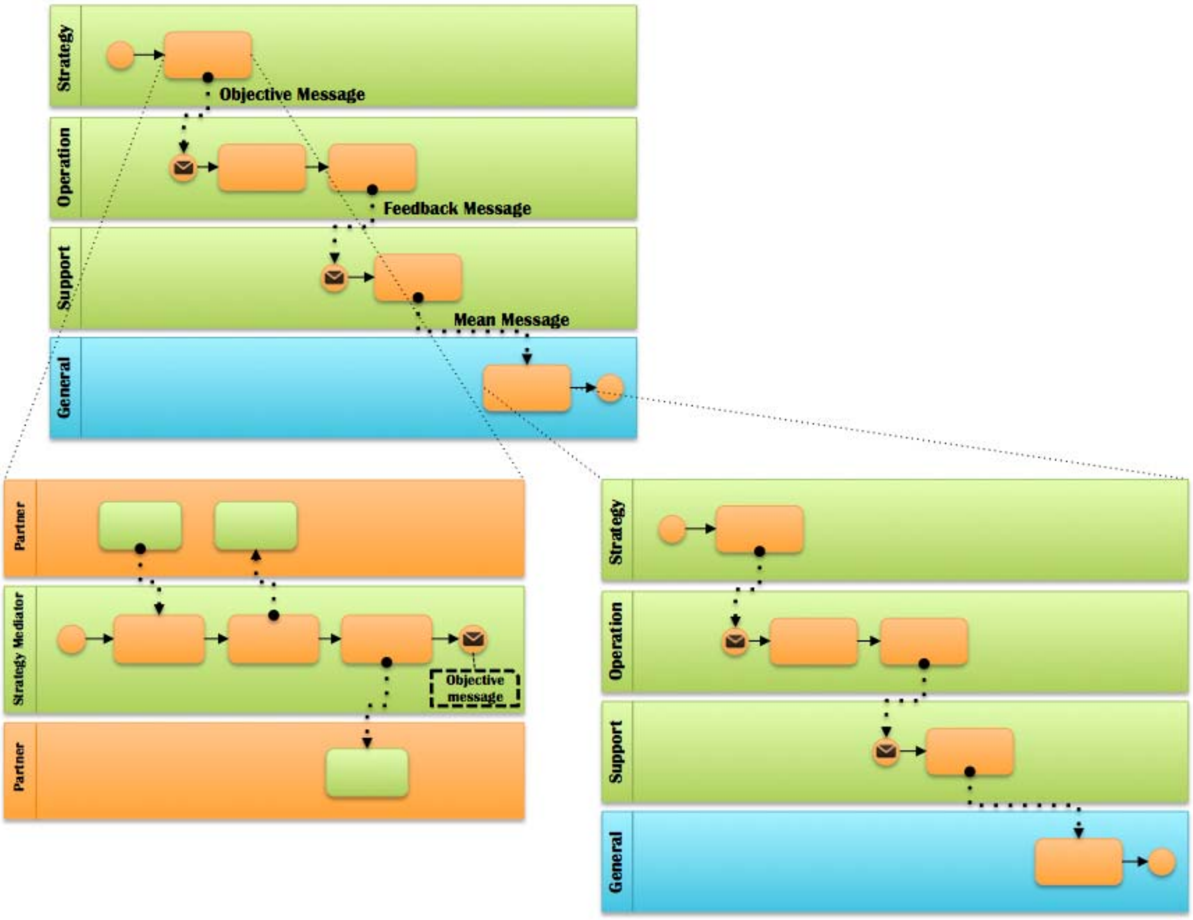


FIGURE III - 19 RELATIONS OF COLLABORATIVE PROCESS CARTOGRAPHY AND COLLABORATIVE PROCESS MODEL

IV.2. Example of Collaborative Process Model

This section presents the example of collaborative process cartography and collaborative process model. The example of the collaborative process cartography is represented in Figure III-20 and Figure III-21. Figure III-20 represents the collaborative process cartography, which is translated from the main function model of A0 Network in Figure III-13.

In Figure III-13, there are three main functions: *Choose Partner*, *Sell Product* and *Sell Component*. First, the main function: *Choose Partner* is transferred from the strategy objective in Figure III-12. The main function: *Choose Partner* is a strategy task. So the main function: *Choose Partner* is represented in the strategy pool in Figure III-20. Second, the main function: *Sell Product* is transferred from the general objective in Figure III-12. The main function: *Sell Product* is a general task, which includes strategy, operation or support tasks. So the main function: *Sell Product* is represented in the general pool in Figure III-20. Third, the main function: *Sell Component* is transferred from operation objective in Figure III-12. The main function: *Sell Component* is an operation task. So the main function: *Sell Component* is represented in the operation pool in Figure III-20. Finally, because there is no main function, which is transferred from the support objective, there does not exist the support pool in Figure III-20.

About the exchanged messages, in Figure III-13, user defined the control/input/output message. The main function: *Choose Partner* sends control message: *Wait for order trigger* to the main function: *Sell Product*. But in the collaborative process cartography, as defined in chapter II section III.4, the strategy pool can only send objective message to the other pools. In Figure III-20, the control message: *Wait for order trigger* has been transferred to an objective message: *Wait for order trigger*. The objective message is represented as output message of *Choose Partner* and input message for the message event, which trigger *Sell Product*. In Figure III-13, the main function: *Sell Product* sends control message: *Order taken trigger* to the main function: *Sell Component*. As defined in chapter II section III.4, the operation pool can only receive the objective message or the mean message. In Figure III-20, the control message: *Order taken trigger* has been transferred to an objective message or a mean message: *Order taken trigger*. The message is represented as output message of *Sell Product* and input message for the message event, which trigger *Sell Product*. In Figure III-13, the main function: *Sell Component* sends control message: *Component sold feedback* to the main function: *Sell Product*. As defined in chapter II section III.4, the operation pool can only send feedback message to the other pools. In Figure III-20, the control message: *Component sold feedback* has been transferred to a feedback message. The message is represented as output message of *Sell Component* and input message for the message event, which trigger *Sell Product*.

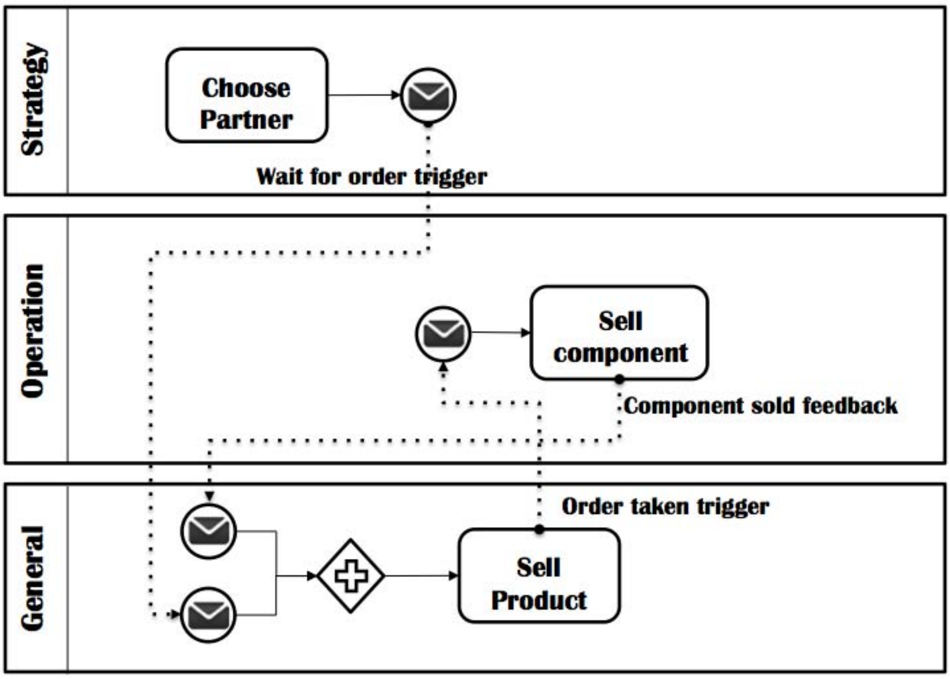


FIGURE III - 20 PROCESS CARTOGRAPHY OF A0 NETWORK

Figure III-21 represents the decomposed collaborative process cartography of the main function: *Sell Product* in Figure III-20. Because the main function: *Sell Product* in Figure III-20 is a general task, which means that the function can be decomposed into strategy, operation or support task, which means that the function can be decomposed to another collaborative process cartography. The collaborative process cartography of *Sell Product* is translated from the main function model of *A2 Sell Product* in Figure III-14.

In Figure III-14, there are three main functions: *Place Order*, *Send Payment* and *Deliver Product*. First, the main function: *Place Order* is transferred from the strategy objective in Figure III-12. The main function: *Place Order* is a strategy task. So the main function: *Place Order* is represented in the strategy pool in Figure III-21. Second, the main function: *Send Payment* is transferred from the operation objective in Figure III-12. The main function: *Send Payment* is an operation task. So the main function: *Send Payment* is represented in the operation pool in Figure III-21. Third, the main function: *Sell Component* is transferred from operation objective in Figure III-12. The main function: *Deliver Product* is a support task. So the main function: *Deliver Product* is represented in the support pool in Figure III-21. Finally, because there is no main function, which is transferred from the general objective, there does not exist the general pool in Figure III-21, which means that the decomposition of the collaborative process cartography has been completed.

About the exchanged messages, in Figure III-14, user defined the control/input/output message. The main function: *Place Order* receives control message: *Wait for order trigger* out. But in the collaborative process cartography, as defined in chapter II section III.4, the strategy pool can only receive feedback or mean message from the other pools. In Figure III-21, the control message: *Wait for order trigger* has been transferred to a feedback message: *Wait for order trigger*. The message is represented as input data object of the message event, which triggers the task: *Place Order*. In Figure III-14, the main function: *Place Order* sends control message: *Order taken trigger* out. As defined in chapter II section III.4, the strategy pool can only send objective message to the other pools. In Figure III-21, the control message: *Order taken trigger* has been transferred to an objective message: *Order taken trigger*. The message is represented as output message of a message event, which executes after the task: *Place Order*. In Figure III-14, the main function: *Send Payment* receives control message: *Product delivered feedback* from the main function: *Deliver Product*. As defined in chapter II section III.4, the operation pool can only receive mean message from the support pool. In Figure III-21, the control message: *Product delivered feedback* has been transferred to a mean message. The message is represented as output message of *Deliver Product* and input message for the message event, which triggers *Send Payment*. In Figure III-14, the main function: *Deliver Product* receives the control message: *Component sold feedback*. As defined in chapter II section III.4, the support pool can only receive the feedback message from the operation pool. In Figure III-21, the control message has been transferred to a feedback message. The message is represented as input message of the message event, which triggers *Deliver Product*.

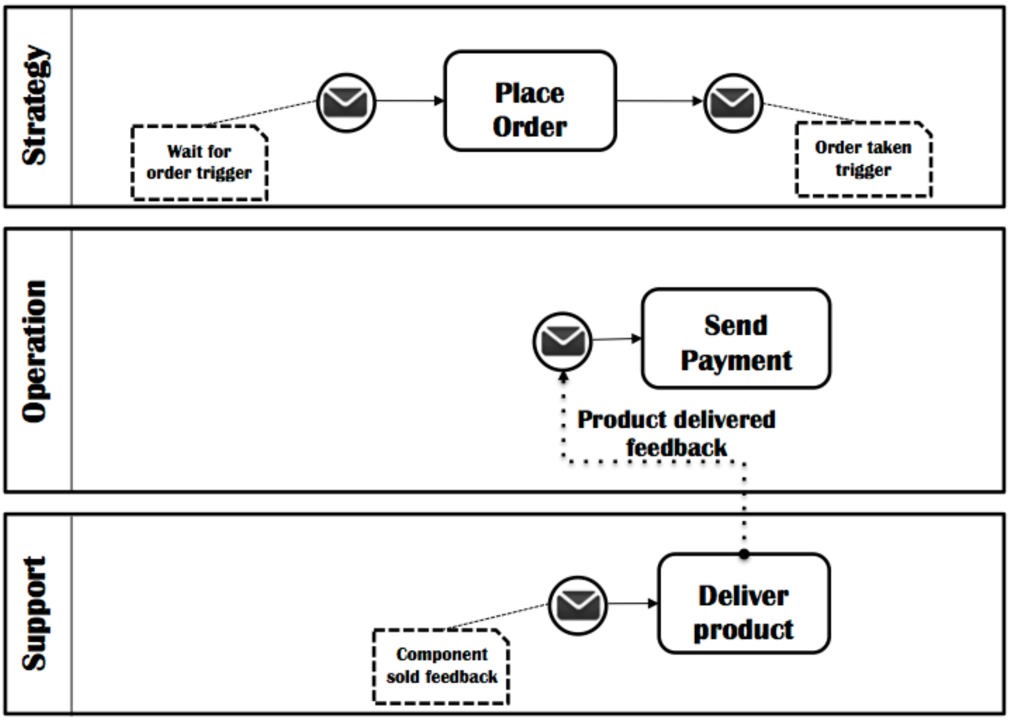


FIGURE III - 21 PROCESS CARTOGRAPHY OF A2 SELL PRODUCT

The collaborative process models, which are the decomposition of the main function in the collaborative process cartography, are represented in Figure III-22, Figure III-23 and Figure III-24. To show the collaborative process models clearly, three main functions (one strategy function, one operation function and one support function) in Figure III-20 and Figure III-21 have been chosen to be decomposed into the collaborative process models. The collaborative process model of *Choose Partner* is shown in Figure III-22. Figure III-7 defines the collaborative network of *Choose Partner*. The network contains: *Assembler*, *Supplier 2* and *Supplier 3*. So the partner pool: *Assembler*, *Supplier 2* and *Supplier 3* are represented in Figure III-22. In the partner pools, there are only tasks or functions, which are provided by partners and represented in Figure III-15 and Figure III-16. There is no sequence follows among these partners' tasks. These partners' tasks only receive messages. Additionally, there is a strategy mediator pool. Because the main function: *Choose Partner* is a strategy function, in the collaborative process model of *Choose Partner*, there is a strategy mediator. The strategy mediator pool has mediator functions, which invokes partners' tasks by the collaborative process, which is the sequence flows in the mediator pool. Because the main function: *Choose Partner* sends out the objective message: *Wait for order trigger*. In Figure III-22, at the end of the collaborative process, the *Wait for order trigger* is represented as the output data objective of the end message event.

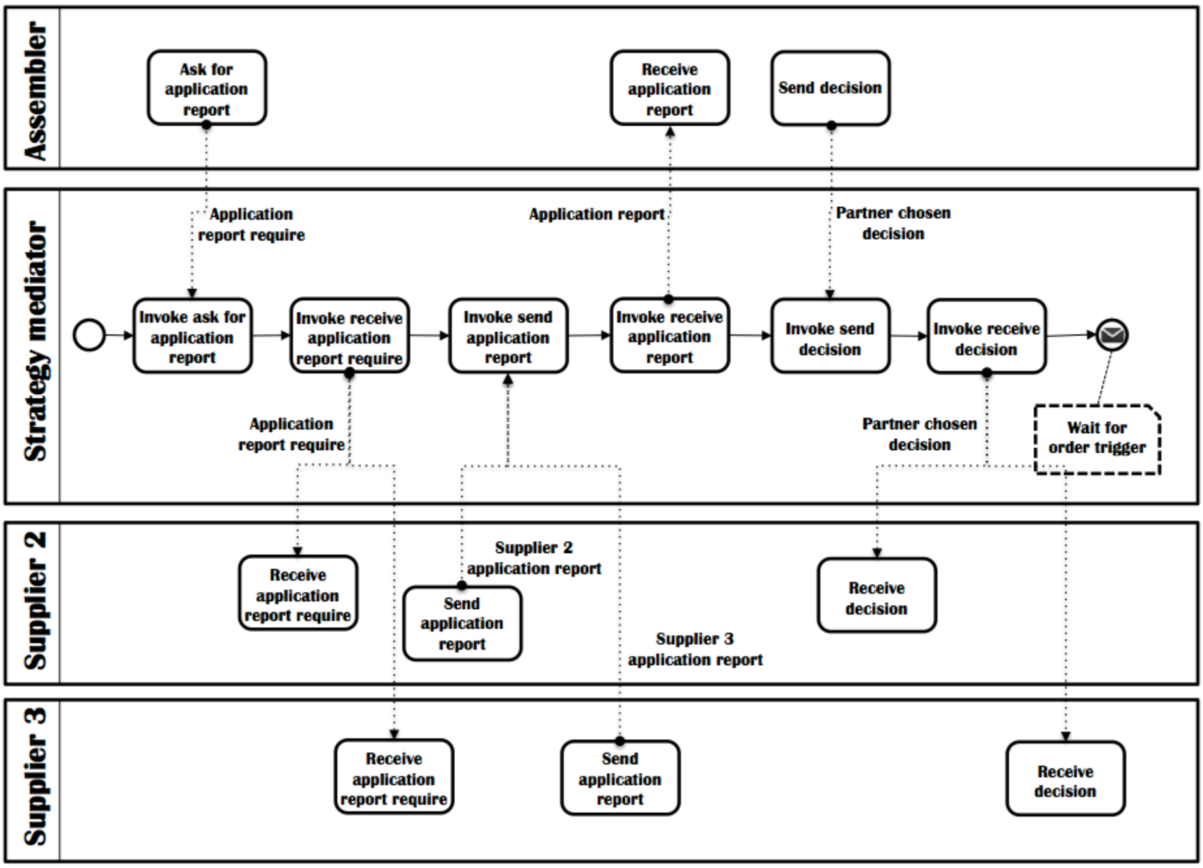

Assembler
Ask for application report
Receive application report
Send decision
Strategy mediator
Application report require
Application report
Partner chosen decision
Invoke ask for application report
Invoke receive application report require
Invoke send application report
Invoke receive application report
Invoke send decision
Invoke receive decision
Application report require
Partner chosen decision
Wait for order trigger
Supplier 2
Receive application report require
Supplier 2 application report
Send application report
Supplier 3 application report
Receive decision
Supplier 3
Receive application report require
Send application report
Receive decision

FIGURE III - 22 COLLABORATIVE PROCESS OF CHOOSE PARTNER

The collaborative process model of *Pay Product* is shown in Figure III-23. Figure III-8 defines the collaborative network of *Pay Product*. The network contains: *Assembler* and *Client*. So the partner pool: *Assembler* and *Client* are represented in Figure III-23. In the partner pools, there are only tasks or functions, which are provided by partners and represented in Figure III-15 and Figure III-16. There is no sequence follows among these partners' tasks. These partners' tasks only receive messages. Additionally, there is an operation mediator pool. The operation mediator pool has mediator functions, which invokes partners' tasks by the collaborative process, which is the sequence flows in the mediator pool. Because the main function: *Send Payment* receives the mean message: *Product delivered feedback*. In Figure III-23, at the beginning of the collaborative process, the *Product delivered feedback* is represented as the input data objective of the start message event.

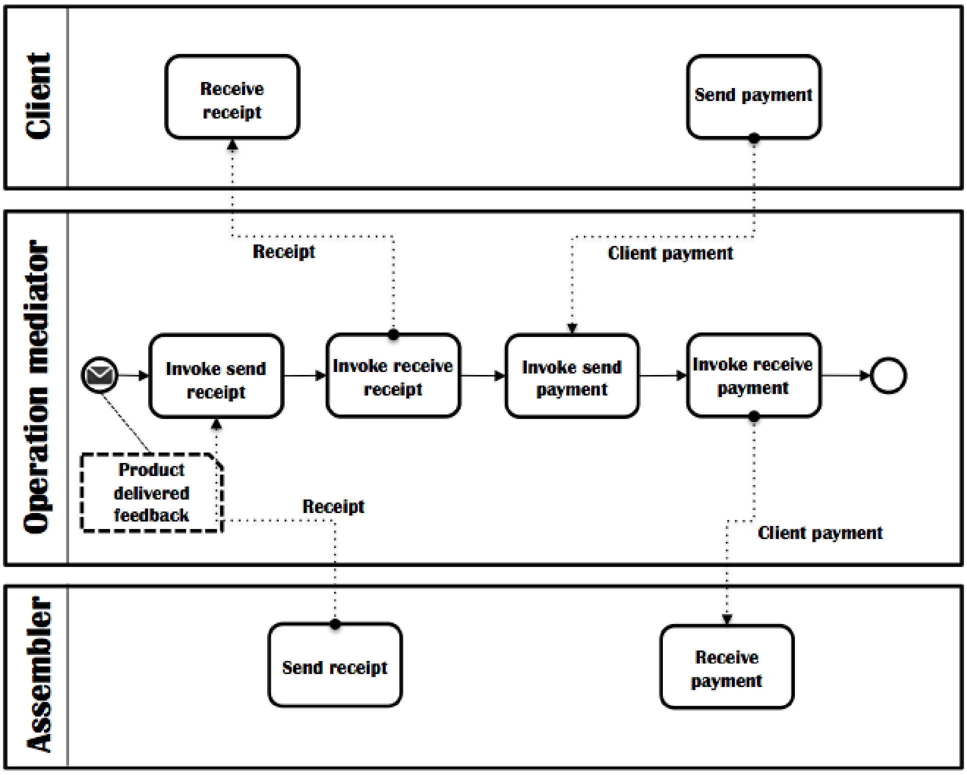


FIGURE III - 23 COLLABORATIVE PROCESS OF SEND PAYMENT

The collaborative process model of *Deliver Product* is shown in Figure III-24. Figure III-8 defines the collaborative network of *Deliver Product*. The network contains: *Assembler* and *Client*. So the partner pool: *Assembler* and *Client* are represented in Figure III-24. Same as Figure III-22 and Figure III-23, there is no sequence follows among these partners' tasks. These partners' tasks only receive messages. Additionally, there is a support mediator pool. Because the main function: *Deliver Product* is a support task. The support mediator pool has mediator functions, which invokes partners' tasks by the collaborative process, which is the sequence flows in the mediator pool. Because the main function: *Deliver Product* receives the feedback message: *Component sold feedback*. In Figure III-24, at the beginning of the collaborative process, the *Component sold feedback* is represented as the input data object of the start message event. The main function: *Deliver Product* sends the *Product delivered feedback*. In Figure III-24, at the end of the collaborative process, the *Product delivered feedback* is represented as the output data object of the end message event.

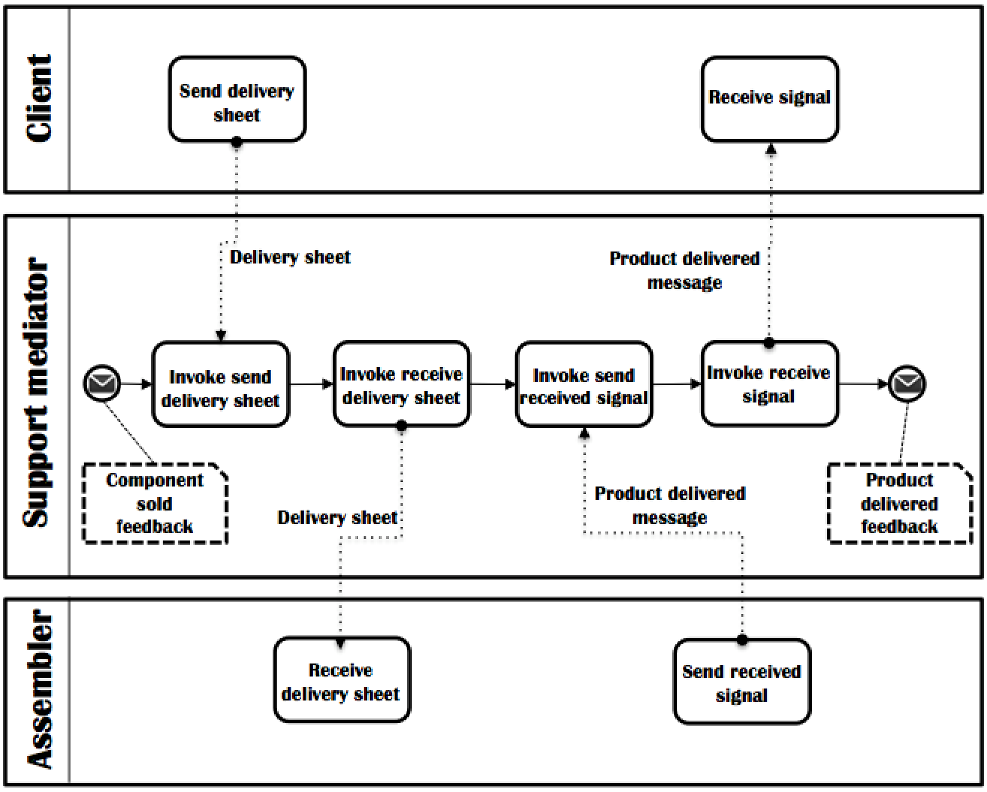


FIGURE III - 24 COLLABORATIVE PROCESS OF DELIVER PRODUCT

To summarize, the collaborative process includes the collaborative process cartography and the collaborative process model. It normally starts with a collaborative process cartography, which contains strategy, operation, support and general pools. The tasks in general pool can be decomposed to another collaborative process cartography until that there is no general pool in the collaborative process cartography. The tasks in strategy, operation and support pools can be decomposed into collaborative process models, which contain one mediator pool and several partner pools. This is the target collaborative process, which is deduced by the research work of this thesis.

V. Conclusion

This chapter has presented how the collaborative knowledge is collected in MISE 2.0. And how it is possible to deduce from it. The collaborative network model and IDEF0 based function model are defined to collect collaborative knowledge. The collaborative network model focus on the collection of information, which concerns partners, collaborative network and collaborative objectives. The collaborative network model is a pyramid model. It defines the collaborative objectives of the whole network, and decomposed the network into sub networks by the collaborative objective. The IDEF0 based function model mainly collects the information about the shared functions of partners. The IDEF0 is reused in two ways: the main function model and function model. The main function model is transferred from the collaborative network model. The main functions of the main function model come from the collaborative objectives of the collaborative network and sub network. The control message among the main functions can be added manually or completed after the business service selection in chapter IV. But for now, the message is added manually. The function model is just a list of functions of partners. So the input models (the collaborative network model and the function model) of the research work in the thesis are simple to define and easy to use. It decreases the user's workload. About the output model (the collaborative process model), the collaborative process cartography has been introduced. The process cartography separates collaborative process to strategy, operation and support process.

For the tasks of strategy, operation and support, can be decomposed into sub collaborative processes with one mediator pool and several partner pools. In the collaborative process, there exits correspondence mediator to orchestrate the collaborative process.

In chapter II section III.2, the relations of collaborative situation elements in the collaborative situation framework have been introduced (Figure II-12). The collaborative network model, function model and collaborative process model, which are presented in this chapter, can be located in the relations of collaborative situation elements. The positions of MISE 2.0 models are represented in Figure III-25. The collaborative network model presents the organizational elements. The IDEF0 based function model includes the functional elements. The knowledge of function model contains enough informational element (the input/output/control message). As the function model in business level, the detailed message information (e.g., attributes, entity-relationship, data types and so on) is not needed. The extra informational model is not defined here. The BPMN based collaborative process model is located in the process element. The goal of the thesis is to transfer the collaborative network model and the function model to into the collaborative process model. There are two questions: how to gather this knowledge? And how to present the target knowledge? They have been solved in this chapter. The next question would be how to transfer the gathered knowledge to the process model? And what are the ontology and the transformation rules? These two questions are answered in the next chapter.

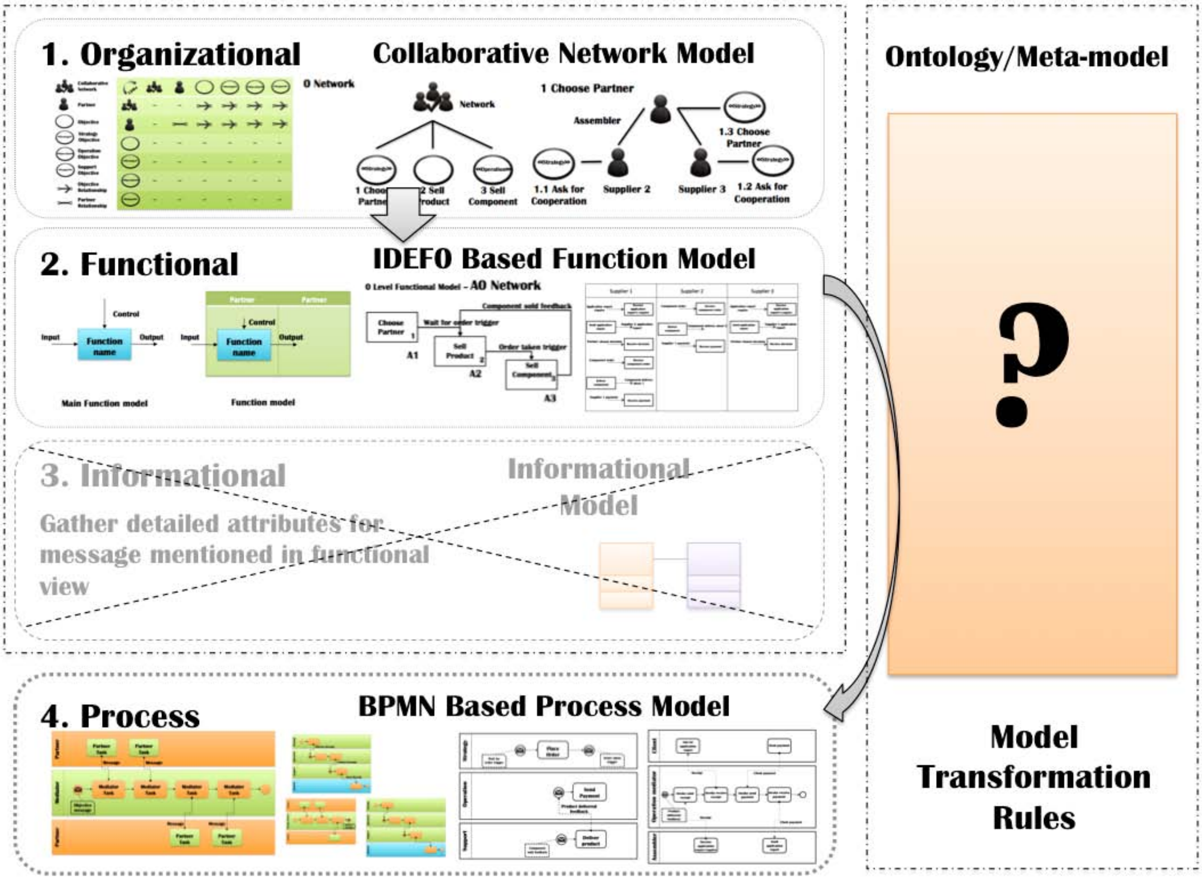


FIGURE III - 25 POSITIONS OF MODELS IN MISE 2.0 FRAMEWORK

Chapter IV:
Model Transformation

I. Introduction ... 78

II. Reference Ontologies ... 79

II.1. Definition of Ontology ... 79

II.2. Ontology of MIT Process Handbook ... 80

II.3. Collaborative Network Ontology of MISE 1.0 ... 82

III. MISE 2.0 Collaborative Ontology ... 84

III.1. The Collaborative Concepts ... 85

III.2. The Mediation Concepts ... 88

III.3. Transformation Rules ... 90

IV. Service Selection and Sequence Deduction ... 100

IV.1. Business Service Selection ... 101

IV.2. Process Sequence Deduction ... 105

IV.2.1. Process Deduction ... 105

IV.2.2. Sequence Deduction ... 109

V. Conclusion ... 111

I. Introduction

In this chapter, the vision is focus on the transformation from the collaborative network model and the function model to the collaborative process model. Figure IV-1 shows the position of this chapter in UJ picture. To complete the mission of model transformation, there are two ways: metamodel and ontology to support model transformation. According to (Mellor, 2004), "*a metamodel is a model of a modeling language. The metamodel defines the structure, semantics and constraints for a family of models*". However, (Clark et al., 2004) define that: "*a metamodel is a model of a language that captures its essential properties and features. These include the language concepts it support, its textual and/or graphical syntax and its semantics (what the models and programs written in the language mean and how they behave)*". (Bézivin, 2005) considers "*the notion of ametamodel is strongly related to the notion of ontology. A metamodel is a formal specification of an abstraction, usually consensual and normative. From a given system we can extract a particular model with the help of a specific metamodel. A metamodel acts as a precisely defined filter expressed in a given formalism*". A metamodel typically defines the languages and processes from which a model may be formed. A model is an abstraction of phenomena in the real world; a metamodel is yet another abstraction, highlighting properties of the model itself. A model always conforms to a unique metamodel.

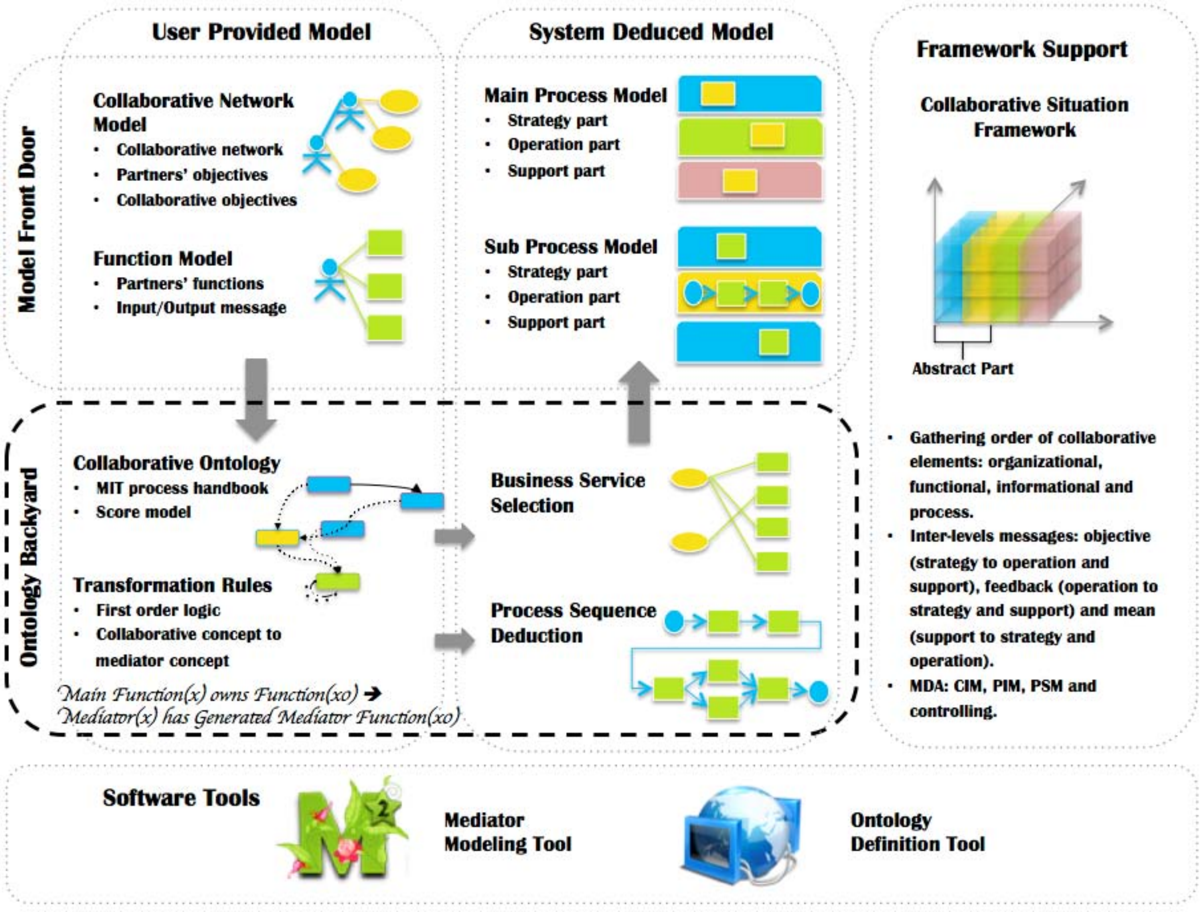

User Provided Model
System Deduced Model
Framework Support
Model Front Door
Collaborative Network Model
• Collaborative network
• Partners' objectives
• Collaborative objectives
Function Model
• Partners' functions
• Input/Output message
Main Process Model
• Strategy part
• Operation part
• Support part
Sub Process Model
• Strategy part
• Operation part
• Support part
Collaborative Situation Framework
Abstract Part
Ontology Backyard
Collaborative Ontology
• MIT process handbook
• Score model
Transformation Rules
• First order logic
• Collaborative concept to mediator concept
Main Function(x) owns Function(xo) →
Mediator(x) has Generated Mediator Function(xo)
Business Service Selection
Process Sequence Deduction
• Gathering order of collaborative elements: organizational, functional, informational and process.
• Inter-levels messages: objective (strategy to operation and support), feedback (operation to strategy and support) and mean (support to strategy and operation).
• MDA: CIM, PIM, PSM and controlling.
Software Tools
Mediator Modeling Tool
Ontology Definition Tool

Figure IV - 1 Position of Chapter IV in UJ

But the metamodel cannot store instances inside. During the research of MISE 2.0 abstract level, a metamodel without instances can just transfer the gathered knowledge to a part of the collaborative process model. It cannot solve the following problems:

- How to select shared functions from partners for each collaborative objective?
- Assuming that functions may be selected, how to create sequences or orders for these functions to make up a collaborative process?

So the design of the MISE 2.0 abstract level is focused on the collaborative ontology. It can confirm the models and also stores instances of the concepts. The process instances of MIT process handbook are transferred into the collaborative ontology of MISE 2.0. Based on the collaborative ontology of MISE 2.0, the transformation rules in the format of first-order logic are defined. The transformation rules can transfer part of the collaborative process. To complete the whole transformation of the collaborative process, as shown in Figure IV-1, the methods of business service selection and process sequence deduction are define. These methods solve the problems, which has been mentioned above.

In this chapter, section II provides the state of art of ontology. It explains the definition of ontology, the usages of ontology, the languages to define ontology, the collaborative ontology of MISE 1.0 and the ontology of MIT process handbook. Section III introduces the collaborative ontology of MISE 2.0 and the transformation rules. Section IV addresses the principle and the algorithms of business service selection and the methodology of business sequence deduction to solve both problems, exposed above.

II.Reference Ontologies

II.1. Definition of Ontology

Ontologies have been developed to provide a machine accessible semantics of information sources that can be communicated between different agents (software and humans). According to (Neches et al., 1991), "*an ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms, and relations to define extensions to the vocabulary*". (Gruber, 1993) defines that *"An ontology is a formal, explicit specification of a shared conceptualization"*. (Gruber, 1993) explains that, a "conceptualization" refers to an abstract model of some phenomenon in the world, which identifies the relevant concepts of the phenomenon. "Explicit" means that the type of concepts used and the constraints on their use are explicitly defined. "Formal" refers to the fact that the ontology should be machine-readable. (Velardi et al., 2001; Missikoff et al., 2002) summarize that ontology is a formal and explicit description of concepts of a particular domain, together with characteristics of these concepts and relations between them. Ontology is referred to as a representation of knowledge that can be used and reused in order to facilitate the comprehension of concepts and relations as well as the communication between different domain actors. (Rajsiri et al., 2010) summarized the general reasons of building ontologies as follows: i) sharing common understanding of the structure of information among people or software agents; ii) enabling reuse of domain knowledge; iii) making explicit domain assumptions: explicit specifications of domain knowledge are useful for new users who must learn what the terms in the domain mean; iv) separating the domain knowledge from the operational knowledge; v) analyzing domain knowledge: formal analysis of terms is extremely valuable both when attempting to reuse existing ontologies and when extending them.

According to (Navigli, 2004), the domain ontology (or domain-specific ontology) models a specific domain, or part of the world. It represents the particular meanings of terms as they apply to that domain. For example the word card has many different meanings. Ontology about the domain of poker would model the "playing card" meaning of the word, while ontology about the domain of computer hardware would model the "punched card" and "video card" meanings. An upper ontology (or foundation ontology) is a model of the common objects that are generally applicable across a wide range of domain ontologies. It contains a core glossary in whose terms objects in a set of domains can be described.

Contemporary ontologies share many structural similarities, regardless of the language in which they are expressed. As mentioned above, most ontologies describe individuals (instances), classes (concepts), attributes, and relations. Common components of ontologies are summarized as following:

- Individuals: instances or objects (the basic or "ground level" objects).
- Classes: sets, collections, concepts, classes in programming, types of objects, or kinds of things.
- Attributes: aspects, properties, features, characteristics, or parameters that objects (and classes) can have.
- Relations: ways in which classes and individuals can be related to one another.

An ontology language is a formal language used to encode the ontology. There are a number of such languages for ontologies, both proprietary and standards-based. There are different classifications of languages: the traditional ontology languages (Common Logic, DOGMA, KIF[1], etc.), by syntax (OIL[2], OWL, RDF, etc.) and by structure (F-logic[3], OKBC[4], KM[5], etc.). Considering the usability and practicability, the common languages of ontologies are explained as followed:

- DOGMA (developing Ontology-Grounded Methods and Applications) is an ontology approach and framework that is not restricted to a particular representation language. This approach has some distinguishing characteristics that make it different from traditional ontology approaches. (De Moor et al., 2006)
- IDEF5 (Integrated Definition for Ontology Description Capture Method) is a software engineering method to develop and maintain usable, accurate, domain ontologies. This standard is part of the IDEF family of modeling languages in the field of software engineering. (Benjamin et al., 1994)
- RDF (Resource Description Framework) is a W3C[6] recommendation that defines a general-purpose language for defining meta-data in the web. RDF is particularly intended for representing meta-data about web resources (e.g. title, author, copyright, etc.). It does not require that resources be retrievable on the web and is therefore suitable for representing any kind of meta-data. (Godoy, 2005)
- The RDF Schema is an extension to RDF. It provides vocabularies for RDF. While RDF is used to relate resources by means of properties, the RDF Schema introduces the notions of resource classes, subclasses, and properties. It can also impose restrictions on the domain and range of properties. (Lee et al., 2001)
- OWL (Web Ontology Language) endorsed by the W3C, is a family of knowledge representation languages for authoring ontologies. Intuitively, OWL can represent information about categories of objects and how objects are interrelated. It can also represent information about objects themselves. (McGuinness et al., 2004)

The general introduction of ontology has been presented. In section II.2 and section II.3, two special ontologies will be explained. The first one is the ontology of MIT process handbook. In the collaborative ontology of MISE 2.0, there are instances from MIT process handbook. The second one is the collaborative network ontology of MISE 1.0. This ontology deals with deduction of collaborative process in MISE 1.0.

II.2. Ontology of MIT Process Handbook

The very first research work specifically on the Process Handbook project began in 1991. (Malone et al., 2003) The project has been one of the primary projects in the MIT Center for Coordination Science. In 1996, several members of the project team start an MIT spin-off company. Under a license of MIT, Phios[7] develops commercial versions of the Process Handbook software tools and extended the knowledge base. They also worked on projects that integrate the Process Handbook with other tools for visualizing supply chain processes, analyzing organizational change, and classifying company's business models. The goal of the Process

[1] Knowledge Interchange Format, Ontolingua based on KIF
[2] Ontology Inference Layer
[3] Frame Logic
[4] Open Knowledge Base Connectivity
[5] Knowledge Machine
[6] The World Wide Web Consortium (W3C) is the main international standards organization for the World Wide Web
[7] Phios Corporation: http://www.phios.com/

Handbook project is to provide a theoretical and empirical foundation for such tasks as enterprise modeling, enterprise integration, and process re-engineering. According to (Bernstein et al., 1995), the project includes: i) collecting examples of how different organizations perform similar processes, and ii) representing these examples in an on-line "Process Handbook" that includes the relative advantages of the alternatives. To represent the large number of processes, the MIT process handbook ontology is defined. Figure II-2 illustrates the Process Handbook Ontology.

The Process Handbook ontology provides a specialization hierarchy of processes and their inter-relationships in the form of properties, which connect the process to its attributes, parts, exceptions, and dependencies to other processes. All major parts of the Process Handbook, such as Process, Bundle, Goal, Exception, Resource, Dependency, and Trade-offs are represented as OWL classes. The business processes in the Process Handbook have been written in OWL and stored as instances in their own files. The key elements of the Process Handbook ontology are (Rajsiri, 2010):

- Process: Like most process-modeling techniques, the process handbook allows processes to be annotated with attributes that capture such information as a textual description, typical performance values (e.g. how long a process takes to execute), as well as conditions. A process is modeled as a collection of activities that can in turn be broken down into sub-activities.
- Resource: A process consumes and produces resources. In other word, resources describe input and output of processes they are related to.
- Dependencies: Another key concept is that coordination can be viewed as the management of dependencies between processes. Every dependency can include an associated coordination mechanism, which is simply the process that manages the resource flow and thereby coordinates the activities connected by the dependency.
- Goal: allows business processes to be composed, or monitors their execution
- Exceptions: It is possible that processes can fail because of exceptions. Therefore we have to anticipate, avoid, or detect and resolve them.
- Bundle: this is a group of related specializations. In general, it is often very useful to create bundles based on the basic questions for asking about any activity.

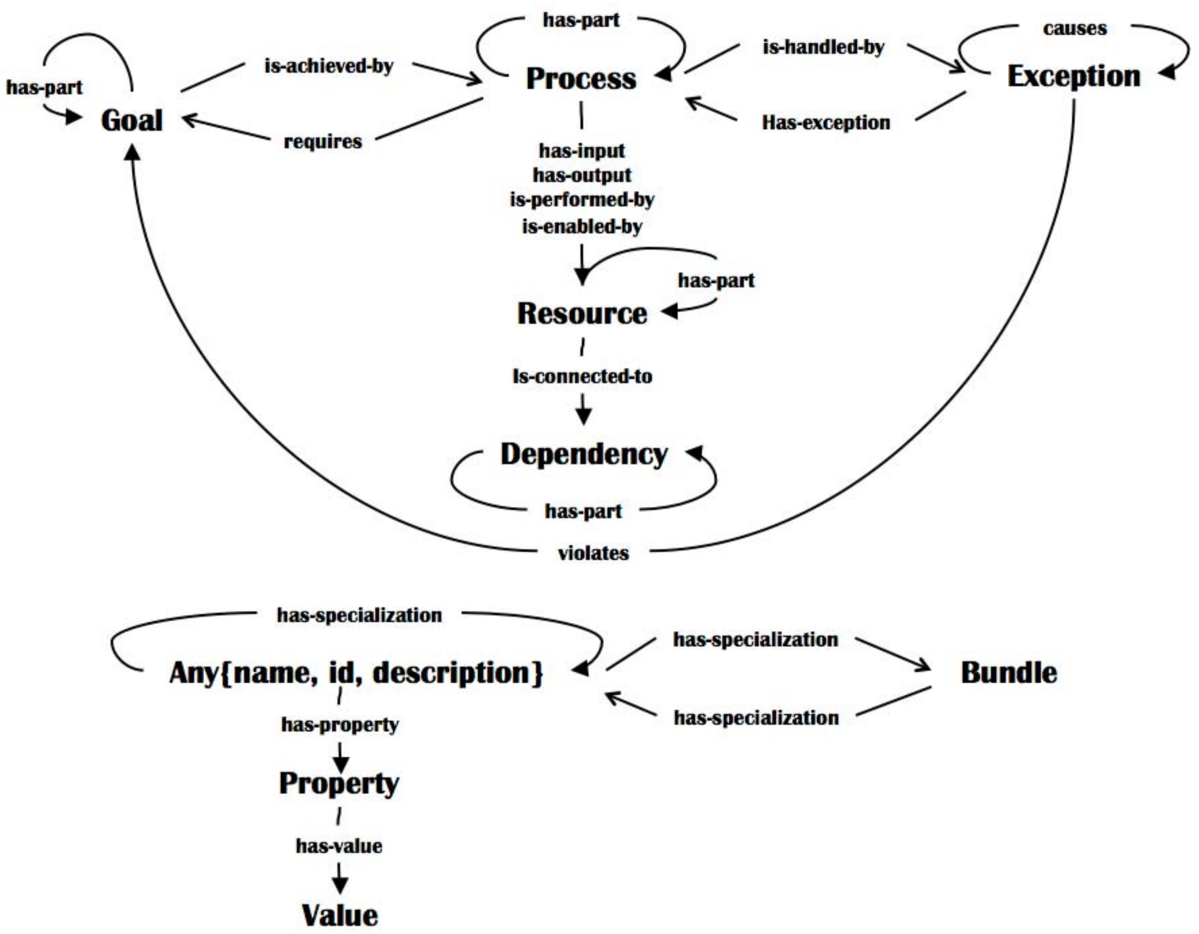


FIGURE IV - 2 GRAPHICAL REPRESENTATION OF THE MIT PROCESS HANDBOOK ONTOLOGY SCHEMA[1]

II.3. Collaborative Network Ontology of MISE 1.0

The Collaborative Network Ontology (CNO) of MISE 1.0 deals with collaboration as well as providing common definitions in collaboration, and network domains. The CNO has two parts: the collaborative ontology (CO) and the collaborative process ontology (CPO) (Figure IV-3).

The CO refers to the conceptualization of enterprise collaboration, and the collaborative network characteristics. The concepts, which are defined in the CO, have been listed as follows:

- A Participant can be a physical actor or an enterprise that joins the network in order to achieve a common goal collaboratively with other participants.
- A role defines the responsibility of a participant in the network. For example, seller, buyer or producer. Role refers to a resource as defined in the MIT Process Handbook.
- The Abstract service is a high-level service that explains the competencies or the know-how of the participant. For example: marketing and sales, procurement. This concept comes from the BAM concept of the MIT Process Handbook.
- A Collaborative network is a group of at least two participants who would like to work together in response to one or multiple common goals and a set of relationships between the participants.
- A Common goal describes the reason why the network is established in terms of products or services to deliver to customers. It gives the direction the partners have to head for and achieve.

[1] https://files.ifi.uzh.ch/ddis/oldweb/ddis/research/completed-projects/semweb/ph-owl/index.html

- A Relationship defines the interaction between two participants. It describes how partners connect to each other. Three types of relationship have been classified: competition, supplier-customer, and group of interest.
- Topology describes the relationships between partners at high level and the overall structure of the network. Three basic forms of topology based on the circulation flow in the network have been presented in (Katzy et al., 2005):chain, star, and peer-to-peer. The form of topology can be distinguished by the orientation of decision-making power and duration of collaboration in the network.
- Decision-making power describes the behavior and the orientation of decision-making in the network. Three decision-making powers are distinguished: central, equal or hierarchic. These three kinds are inspired from the topology characteristics.
- Duration describes the frequency of interactions that occur during the collaboration in the network. (Zaidat, 2005) distinguished two kinds of duration: continuous or discontinuous.

The CPO refers to the conceptualization of a collaborative process. It addresses business service, flow of resources between services and management of flows. It covers the concepts of business service, resource, dependency, coordination service, and MIS service. The definitions of these concepts are described as follows:

- Business service explains the task at functional level. An abstract service is composed of some business services. For example: assemble components of computer, obtain order. This concept is inspired from the functional level activity described in the BAM concept of the MIT Process Handbook.
- Resource can be data, machine, software, tool or material used or produced by business service. For example: message, order, machine, container, and technology. Resource concerns the resource concept defined in the MIT Process Handbook Coordination service is in charge of managing the dependency of resources. For example: manage flow of material, manage accessibility of documents. This concept comes from the model of collaborative process concept of the MIT Process Handbook.
- MIS service is defined in the meta-model of collaborative process (Touzi, 2007). We consider a coordination service as a MIS service because both are collaborative services provided by the collaborative platform (or MIS).
- Dependency between business services (message flow) is a flow from one business service to another when they have a resource in common. The two business services linked by this kind of flow do not belong to the same participant. It can be seen as a movement of a resource between business services.
- Dependency between MIS services (sequence flow) is a flow from one MIS service to another when they have a resource in common. It can be seen as a movement of a resource between MIS services.

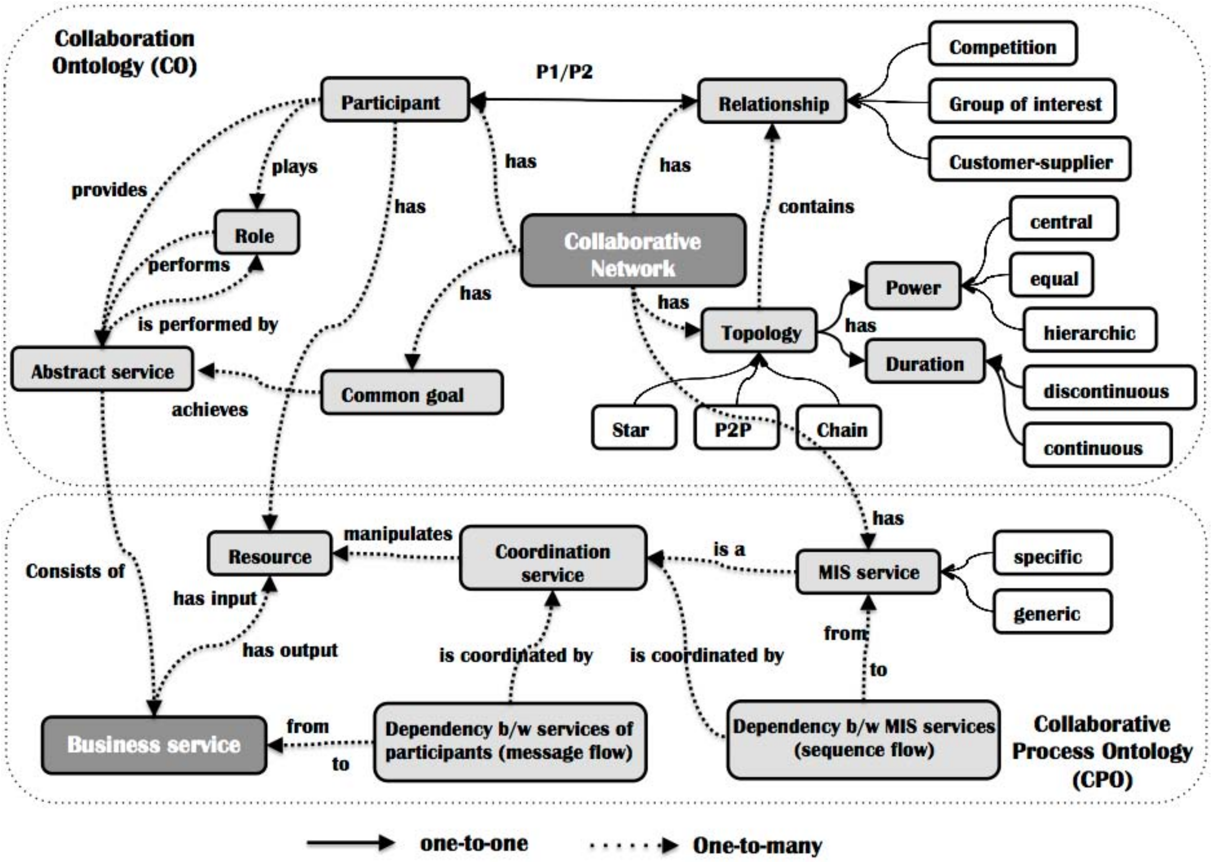


FIGURE IV - 3 THE COLLABORATIVE ONTOLOGY OF MISE 1.0 (RAJSIRI ET AL., 2010)

III. MISE 2.0 Collaborative Ontology

The collaborative ontology is designed to transfer the gathered collaborative knowledge to the collaborative process cartography and the collaborative process. The collaborative ontology contains collaborative concepts and mediation concepts. The collaboration concepts are the gray boxes (or the green boxes if the document is printed in color). The mediation concepts are the black boxes. The collaboration concepts define the concepts and relations of the gathered knowledge in the collaborative network model and the function model. The mediation concepts define the concepts and relations of the deduced knowledge in the collaborative process cartography and model.

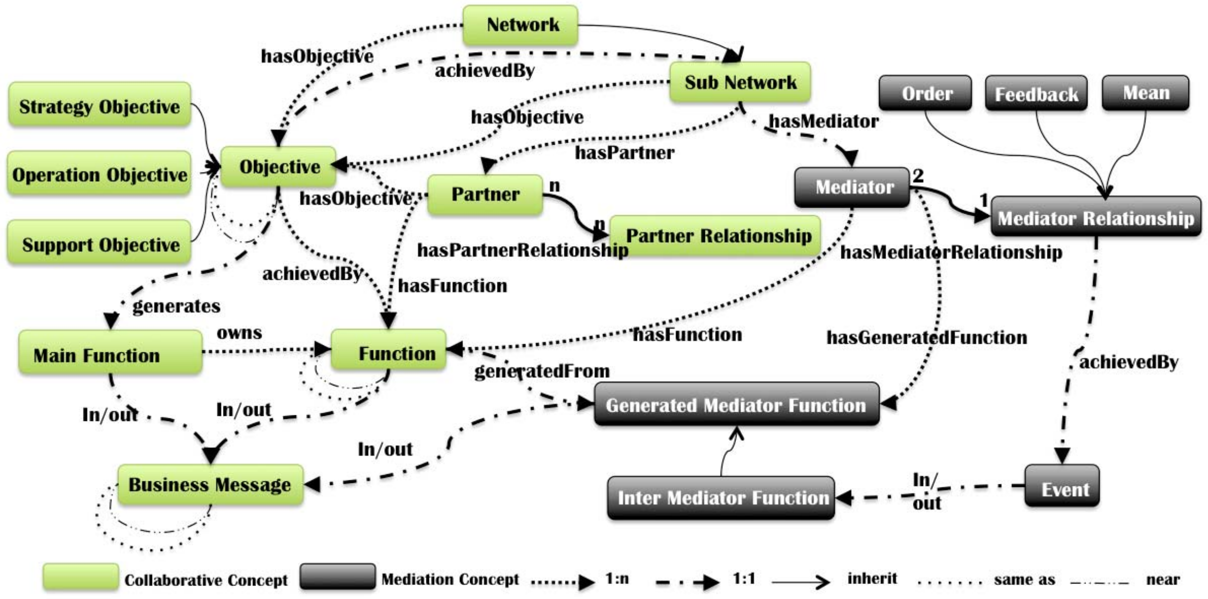


Figure IV - 4 Collaborative Ontology of MISE 2.0

III.1. The Collaborative Concepts

The collaborative concepts refer to the conceptualization of organizational and functional knowledge, which is gathered in the collaborative network model and the function model. The collaborative concepts are explained by two categories: Organizational Concepts and Functional Concepts.

Organizational Concepts

This sub part of concepts refers to the organizations and networks of collaborative situation (Figure IV-5). The following paragraphs detail the concepts of these two categories, together with relations between concepts.

- The *Partner* is an actor, an organization or an enterprise, which joins the collaboration with their individual objectives to achieve a collaborative objective. A *Partner Relationship* defines the interaction between two *Partners*. It provides how partners communicate with each other. The *Partner* is similar with “Participant” in CO of MISE 1.0.
- The *Network* is a set of partners interconnected by communication paths (here is *Partner Relationship*). The *Network* represents the whole organizational system of all the partners and the relationships of partners. The *Network* is similar with “Collaborative Network” in CO of MISE 1.0.
- The *Sub Network* is a part of the *Network*. It contains some *Partners* of the collaboration, but not all of them. It represents the small group of *Partners*, whom should cooperate together to achieve the same collaborative objective.
- The *Objective* is a way for the collaboration to define its goals and direction. The *Objective* here is the collaborative objective, which describes the reason of establishing the collaborative network. The *Objective* can be classified into three types: the *Strategy Objective*, the *Operation Objective* and the *Support Objective*. As defined in chapter III section II.1, the collaborative network model provides three kinds of objectives: strategy, operation and support.

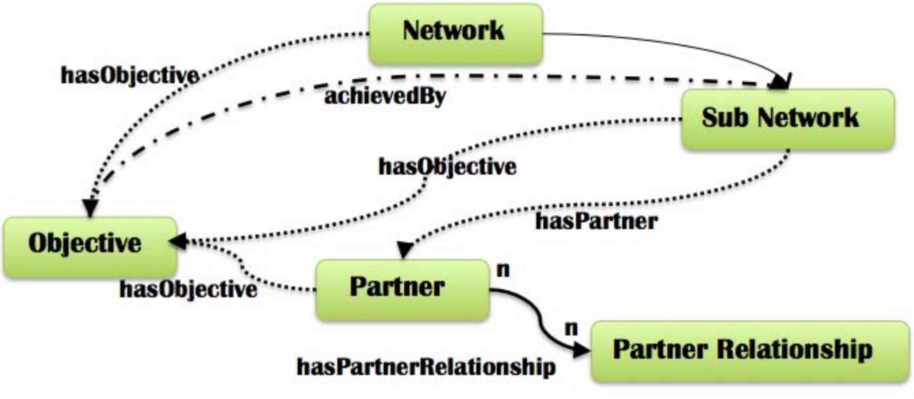


FIGURE IV - 5 PARTNER, NETWORK AND OBJECTIVES

Figure IV-5 illustrates conceptual relations of that sub part of the ontology. The explanations of each relation are listed as following:

- The relation: *hasObjective* between the *Network* and the *Objective* represents that the *Network* may have several *Objectives*. The *Objectives* is the collaborative objectives, which are the reasons to set up the collaboration.
- The relation: *achievedBy* between the *Objective* and the *Sub Network* represents that the *Sub Network* can achieve the *Objective*. It means that the collaborative *Objective* can be accomplished by a small group of the whole *Network*. By this way, the collaborative network can be decomposed to small groups by different collaborative objectives. These small groups (sub networks) may have some partners in common, which means they may cross.
- The relation: *hasObjective* between the *Sub Network* and the *Objective* represents that the *Sub Network* may have several *Objectives*. The *Sub Network* is a part of the collaborative network. It can be seen as a smaller collaborative situation, which may have its own collaborative objectives.
- The relation: *hasPartner* between the *Sub Network* and the *Partner* represents that the *Partners* make up the *Sub Network*. It means the *Sub Network* may be decomposed to the *Partners*.
- The relation: *hasObjective* between the *Partner* and the *Objective* represents that the *Partner* has its individual objectives.

For the *Objective*, there are two special relations. These are relations from one *Objective* to another one. They are the *Same As* and the *Near By* (Figure IV-6). If one instance of a concept is *Same As* another instance of the concept, it means that the two instances are the same object, but with different names or different presentation. For example, "Place Order" and "Send Order", they are different, but they have the same function. So "Place Order" is same as "Send Order". If one instance of a concept is *Near By* another instance of the concept, it means that the two instances do not point to the same objective, but they have something in common. For example, "Place Order" and "Place Order to Supplier A", they both have the function to place order. But the second one precisely defined the information of supplier. They are the relation: *Near By*. The relations: *Same As* and *Near By* help to build a network of instances of ontology in a semantic way. With the large number of interconnected instances, the data of ontology is very useful for selecting partner functions to achieve collaborative objectives (explains in detail in chapter IV section IV.1).

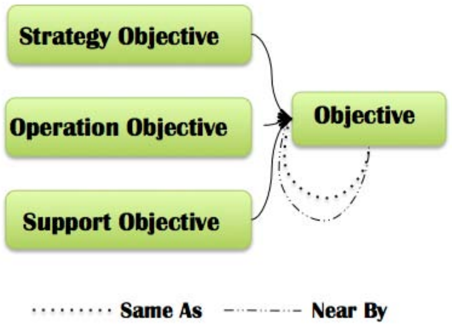


FIGURE IV - 6 SAME AS AND NEAR BY

Functional Concepts

This sub part of concepts refers to the shared functions of partners in the collaborative situation (Figure IV-7). The following paragraphs detail the concepts of these two categories, together with relations between concepts.

- The *Main Function* is higher-level function. It organizes the functions by groups. The Equation (1) in chapter III section III.1 generates the *Main Function* from the *Objective*. The *Main Function* is similar to the "Abstract service" in CO of MISE 1.0.
- The *Function* is the shared task, activity or business service of partner. It takes a set of inputs and provides a set of permissible outputs. The *Function* is similar with the "Business service" in CO of MISE 1.0. The *Function* also has the *Same As* and the *Near By* relations.
- The *Business Message* is the input or output data of the *Function*. It is required or provided by the *Function*. The *Business Message* is similar with the "Resource" in CO of MISE 1.0. The difference is that the "Resource" can be data, machine, software, tool and so on. But the *Business Message* is just business data or abstract message, but not technical data. The *Business Message* also has the *Same As* and the *Near By* relations.

Figure IV-6 illustrates the relations between the functional concepts. The explanations of each relation are listed as following:

- The relation: *owns* between the *Main Function* and the *Function* represents that the *Main Function* have several *Functions*. The combination of several *Functions* can complete the *Main Function*.
- The relation: *In/out* between the *Main Function* and the *Business Message* represents that there are input and output business messages for the *Main Function*.
- The relation: *In/out* between the *Function* and the *Business Message* represents that there are input and output business messages for the *Function*.

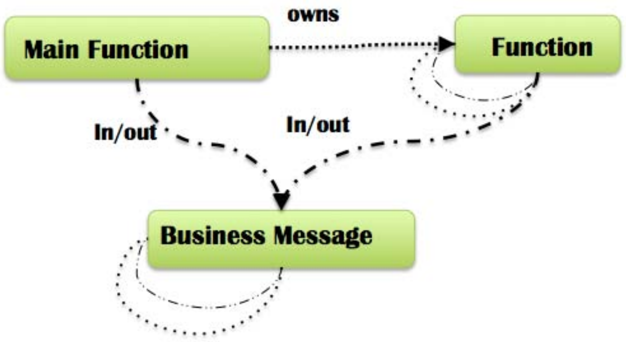


FIGURE IV - 7 MAIN FUNCTION, FUNCTION AND BUSINESS MESSAGE

Figure IV-8 shows the relations between organizational concepts and function concepts.

- The relation: *hasFunction* between the *Partner* and the *Function* represents that the *Partner* provides one or many shared *Functions* to the collaboration.
- The relation: achievedBy between the *Objective* and the *Function* represents that the *Objective* is achieved by one *Function* or a group of *Functions*.
- The relation: *generated* between the *Objective* and the *Function* shows that the *Main Function* can be transferred from the *Objective*.

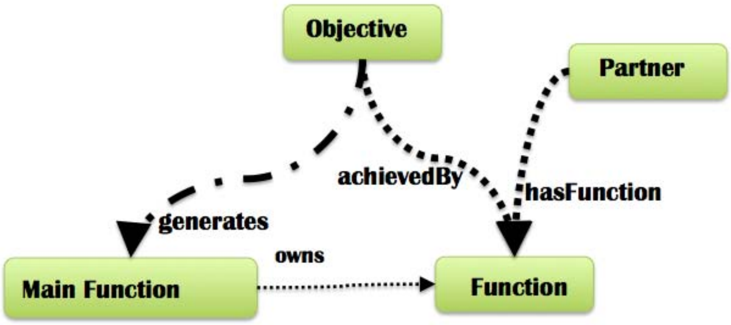


FIGURE IV - 8 RELATIONS AMONG ORGANIZATIONAL AND FUNCTIONAL

III.2. The Mediation Concepts

The mediation concepts refer to the conceptualization of a collaborative process. The collaborative process model has been introduced in section IV of chapter III. The mediation concepts of collaborative ontology come from the definition of collaborative process. The concepts are represented in Figure IV-9 and listed as following:

- The *Mediator* is a participant of the collaboration. It likes the *Partner*. Both of them are the actors in the collaboration. But the *Mediator* manages and orchestrates the collaborative process. From the viewpoint of actors, the *Mediator* is the center of all the *Partners*.
- The *Mediator Relationship* defines the relationships among *Mediators*. As defined in chapter II section III.4, there are the objective, feedback and mean messages, which are transferred among different types of collaborative processes. One collaborative process has one *Mediator*. So the relationships among *Mediators* are classified as *Order* (Objective Message in chapter II section III.4), *Feedback* and *Mean*.
- The *Generated Mediator Function* is the function or business service of *Mediator*. Because the *Mediator* orchestrates the functions of partners. The functions of *Mediator* can be generated from the functions of partners. So here, the functions of *Mediator* are called *Generated Mediator Function*.
- The *Inter Mediator Function* is one special type of the *Generated Mediator Function*. If one *Generated Mediator Function* sends or receives *Objective*, *Feedback* or *Mean* messages, the *Generated Mediator Function* is an *Inter Mediator Function*.
- The *Event* respects to the *Mediator Relationship*. If the *Mediator Relationship* is in the abstract level of ontology, then the *Event* is in the concrete level of the ontology.

Figure IV-9 illustrates the relations between the concepts. The explanation of the relations are listed as following:

- The relation: *hasMediatorRelationship* between the *Mediator* and the *Mediator Relationship* represents that one *Mediator* may relates to another one through one *Mediator Relationship*.
- The relation: *hasGeneratedFunction* between the *Mediator* and the *Generated Mediator Function* represents that the *Mediator* may have several *Generated Mediator Functions*. This relation shows the *Mediator* can access to partners' functions by invoking the *Generated Mediator Functions*. It also shows that what are the functions in the mediator pool of collaborative process model?
- The relation: *achievedBy* between the *Mediator Relationship* and the *Event* the *Mediator Relationship* can be realized by the Event. In section IV of chapter III, between different kinds of collaborative processes

(strategy, operation or support), the message is exchanged by start message event, intermediate message event or end message event. This relation shows the *Event*, which can deliver the message.

- The relations: *In/out* between the *Event* and the *Inter Mediator Function* represents that the *Inter Mediator Function* has input or output *Event*. As mentioned above, the *Inter Mediator Function* sends or receives the exchanged message among different types of collaborative processes. The *Event* represents the message. So for the *Inter Mediator Function*, it must relate to one input or output message event.

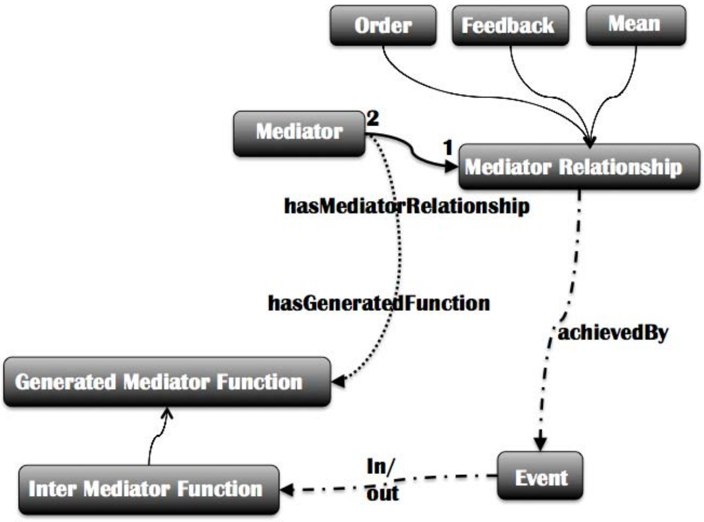


FIGURE IV - 9 MEDIATOR AND MEDIATOR RELATIONSHIP

Figure IV-10 shows the relations between collaborative concepts and mediator concepts. These relations is explained as following:

- The relation: *hasMediator* between the *Sub Network* and the *Mediator* represents that one *Sub Network* has one *Mediator*. In section IV of chapter III, each collaborative process has one mediator. The collaborative process is the decomposition of main function in the collaborative process cartography. The main function comes from the objective of the collaborative network of the collaborative network model. So one *Sub Network* has one Mediator.
- The relation: *hasFunction* between the *Mediator* and the *Function* represents that one *Mediator* can provide several *Functions*. It means that the *Mediator* can play as a partner of the collaboration to provide the functions or business service. If the partners cannot provide enough services, the mediator could have additional or supplementary service. If the mediator uses one of function, but it involves too many exchanges of messages, the mediator could provide a service in the mediator pool to improve the efficiency of the collaborative process, For example, merge documents, select a supplier and so on.
- The relation: *generatedFrom* between the *Function* and the *Generated Mediator Function* represents that the *Function* generated the *Generated Mediator Function*.
- The relation: *In/out* between the *Business Message* and the *Generated Mediator Function* represents that the *Generated Mediator Function* has input or output *Business Message*.

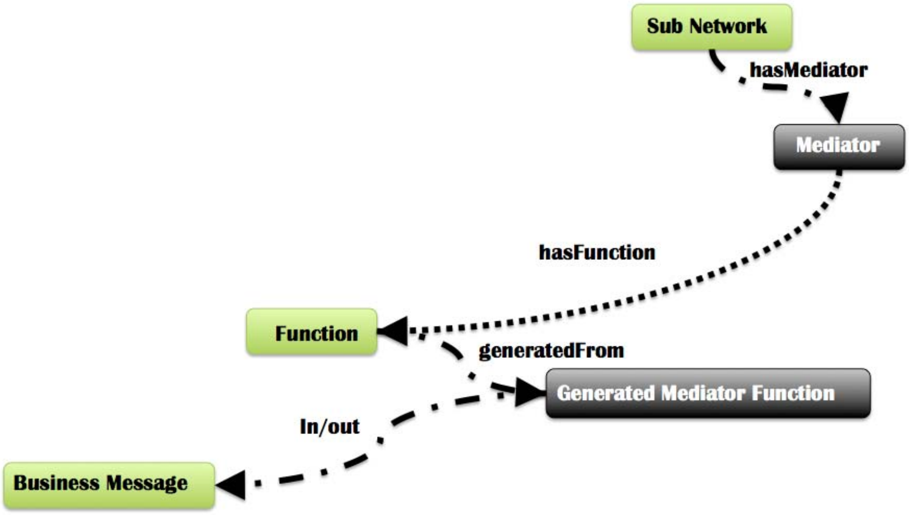


FIGURE IV - 10 MEDIATOR FUNCTION AND INTER MEDIATOR FUNCTION

III.3. Transformation Rules

The transformation rules of the collaborative ontology are defined in first-order logic (Smullyan, 1995), which has been introduced in chapter III. To remind the rules, here, the first-order logic rules are repeated. Due to particularity of transformation rules, first order logic still needs to be expended as followed:

- Class: X is collaborative network → collaborative network(X)
- Association: Y is association implement which is between collaborative network X1 and objective X2 → implement(Y) (collaborative network(X1), objective(X2))
- If-then-else: if (X) → then (Y), else if (X1) → then (Y1), else → then (Y2)
- A set of variables: from X1, X2, X3 to Xn → X1 … Xn

The Transformation rules of collaborative ontology are specified to six groups of rules. The rules are summarized in Table IV-1. These transformation rules aim to transfer the collaboration concepts to the mediation concepts.

The 6 groups of transformation rules deal with the deduction of mediation concepts. Due to the knowledge of the collaboration concepts is already collected by the collaborative network model and the function model, the objective of transformation rules is to transfer the un-known mediation concepts from the known collaboration concepts. The group 1, 2 and 3 create *Mediator*, *Mediator Relationship* and *Generated Mediator Function*. The group 4 creates the relation: *hasGeneratedFunction* to link *Mediator* with the deduced *Generated Mediator Function*. The group 5 transfers the *Mediator Relationship* to the Event. With the *Event*, the group 5 can recognize the *Inter Mediator Function* from the *Generated Mediator Function*.

In this section the transformation rules are presented group by group. The explanation of each group contains: i) the principle of the transformation rules, ii) the involved concepts of collaborative ontology, iii) the equations of transformation rules, and iv) the example of transformation rules.

TABLE IV - 1 TRANSFORMATION RULES

No.	Group Name	Description	Equation	Total No.
Group 1	Create *Mediator*	Deduce *Mediator* from *Sub Network*	(1)	1
Group 2	Create *Mediator Relationship*	Deduce *Mediator Relationship* through *Sub Network*, *Objective*, *Main Function* and *Business Messages*	(2), (3), (4), (5), (6) and (7)	6
Group 3	Create *Generated Mediator Function*	Deduce *Generated Mediator Function* and *In/out* relation through *Function*	(8) and (9)	2
Group 4	Link *Generated Mediator Function* to *Mediator*	Deduce the relation: *hasGeneratedFunction* through *Mediator*, *Objective*, *Main Function* and *Function*. This group based on that the relation: *owns* exists.	(10)	1
Group 5	Create *Inter Mediator Function*	Deduce *Event* from *Mediator Relationship*, Separate *Inter Mediator Function* from *Generated Mediator Function* and link it to *Event*	(11)	1
			Total:	**11**

Group 1: Create Mediator

The Group 1 deals with the creation of *Mediator*. From business point of view, mediator orchestrates the collaborative process. The mediator is an important role in the collaborative network. It manages the collaborative network to achieve the collaborative objective.

The theory is that, if one *Sub Network* contains several *Partners*, then for the *Sub Network*, there is one *Mediator* to manage the collaboration of the *Sub Network*. This group of transformation rule contains one equation. Table IV-2 presents the equation (1) in extended first-order logic. The equation tells that, if there exists an instance X of *Sub Network* and the X links to several instances of *Partner* (X_1, X_2 … X_n) by the relations: *hasPartner*, then there exists an instance X of *Mediator* and the instance links to the instance of *Sub Network* by relations: *hasMediator*.

TABLE IV - 2 GROUP 1 CREATE MEDIATOR

Group 1: Create *Mediator*	
Sub Network → Mediator	
$\forall \text{Sub Network}(X)\ (\forall \text{hasPartner}(\text{Sub Network}(X), \text{Partner}(X_1)) \wedge (\forall \text{hasPartner}(\text{Sub Network}(X), \text{Partner}(X_2)) \wedge \ldots \wedge (\forall \text{hasPartner}(\text{Sub Network}(X), \text{Partner}(X_n)))$	**(1)**
$\rightarrow \exists \text{Mediator}(X) \wedge \exists \text{hasMediator}(\text{Sub Network}(X), \text{Mediator}(X))$	

To explain the transformation rule clearly, the example of transformation is proposed in Figure IV-11. The grey boxes are the existed instances. The black boxes are the deduced instance. “Network 1” is an instance of *Sub Network*. “Client”, “Supplier” and “Assembler” are instances of *Partners*. They are the partners of the “Network 1”. With these instances, the instance “Mediator 1” of *Mediator* is deduced for the “Network 1”. To show that the “Mediator 1” belongs to the “Network 1”, the relation “hasMediator” is created between the “Network 1” and the “Mediator 1”.

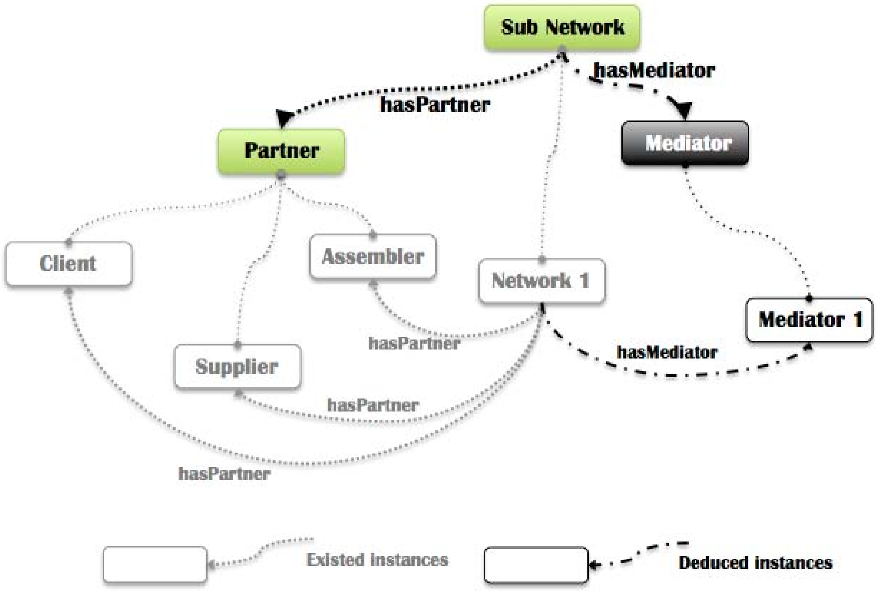


FIGURE IV - 11 EXAMPLE OF GROUP 1

Group 2: Create Mediator Relationship

This group concerns the deduction of *Mediator Relationships* by recognizing the in/out of business message of main function. From business point of view, in one collaborative situation, there may exist several mediators, which manage different collaborative processes and achieve different collaborative objectives. But among different collaborative processes, there exists communications. These communications (has been introduced in chapter II, section III.4 as objective, feedback and mean) are among mediators. The creation of relationships among mediators is crucial.

The group 2 has 6 equations as shown in Table IV-3. The equations are based on the theory of communications among collaborative processes (chapter II section III.4). The theory is that **i) the strategy process sends objective information to operation and support process, ii) the operation process sends feedback information to strategy and support process and iii) the support process sends the mean information to strategy and operation process**. As mentioned in section III.2 of chapter IV, in the collaborative ontology, *Mediator Relationship* is defined to represent the communications among collaborative processes. There are three types of *Mediator Relationship*. First, the Order represents objective information. The *Feedback* represents feedback information. The *Mean* represents mean information. The equation (2) and (3) deals with the deduction of the *Order*. The equation (3) and (4) deals with the deduction of the *Feedback*. The equation (5) and (6) deals with the deduction of the *Mean*. Because these equations are based on the same principle, the equation (2) is taken and explained in detail as an example.

As shown in Table IV-3, the equation (2) starts from *Strategy* and *Operation Objective*. If there is an instance X_1 of *Main Function* transferred from an instance of *Strategy Objective*, and if there is an instance X_2 of *Main Function* transferred from an instance of *Operation Objective*, and if there is an instance m of *Business Message*, the m is the output of X_1, and the m is also the input of X_2, then an instance m of *Order* is deduced and the relations: *hasMediatorRelationship* with *Mediator* *(X_1)* (this is the instance of *Mediator*, that has been generated from the *Sub Network*, which achieves the *Main Function (X_1)*) and *Mediator (X_2)* (this is the instance of *Mediator*, that has been generated from the *Sub Network*, which achieves the *Main Function (X_2)*) are deduced. The equation (3) creates *Order* between *Mediators*, which are transferred from *Strategy* and *Support Objective*. The equation (4) creates *Feedback* between *Mediators*, which are transferred from *Operation* and *Strategy Objective*. The equation (5) creates *Feedback* between *Mediators*, which are transferred from *Operation* and *Support Objective*. The equation (6) creates *Mean* between *Mediators*, which are transferred from *Support* and *Strategy Objective*. The equation (7) creates *Mean* between *Mediators*, which are transferred from *Support* and *Operation Objective*.

TABLE IV - 3 GROUP 2 CREATE MEDIATOR RELATIONSHIP

Group 2: Create *Mediator Relationship*

Strategy* and *Operation Objective* → *Main Function* → *Business Message* → *Order	
If $\forall$Strategy Objective (X_1) ($\forall$generates (Strategy Objective (X_1), Main Function (X_1))) $\wedge$ $\forall$Operation Objective (X_2) ($\forall$generates (Operation Objective (X_2), Main Function (X_2))) If $\forall$Main Function (X_1) ($\forall$out (Main Function(X_1), Business Message (m))) $\wedge$ $\forall$Main Function (X_2) ($\forall$in (Main Function(X_2), Business Message (m)))	**(2)**
$\rightarrow\exists$ Order (m)(hasMediatorRelationship (Mediator (X_1), Order (m))) $\wedge$ $\exists$ Order (m)(hasMediatorRelationship (Mediator (X_2), Order (m)))	

Strategy* and *Support Objective* → *Main Function* → *Business Message* → *Order	
If $\forall$Strategy Objective (X_1) ($\forall$generates (Strategy Objective (X_1), Main Function (X_1))) $\wedge$ $\forall$Support Objective (X_2) ($\forall$generates (Support Objective (X_2), Main Function (X_2))) If $\forall$Main Function (X_1) ($\forall$out (Main Function (X_1), Business Message (m))) $\wedge$ $\forall$Main Function (X_2) ($\forall$in (Main Function (X_2), Business Message (m)))	**(3)**
$\rightarrow\exists$ Order (m)(hasMediatorRelationship (Mediator (X_1), Order (m))) $\wedge$ $\exists$ Order (m)(hasMediatorRelationship (Mediator (X_2), Order (m)))	

Operation* and *Strategy Objective* → *Main Function* → *Business Message* → *Feedback	
If $\forall$Operation Objective (X_1) ($\forall$generates (Operation Objective(X_1), Main Function(X_1))) $\wedge$ $\forall$Strategy Objective (X_2) ($\forall$generates (Strategy Objective(X_2), Main Function(X_2))) If $\forall$Main Function (X_1) ($\forall$out (Main Function(X_1), Business Message (m))) $\wedge$ $\forall$Main Function (X_2) ($\forall$in (Main Function(X_2), Business Message (m)))	**(4)**
$\rightarrow\exists$ Feedback (m)(hasMediatorRelationship (Mediator (X_1), Feedback (m))) $\wedge$ $\exists$ Feedback (m)(hasMediatorRelationship (Mediator (X_2), Feedback (m)))	

Operation* and *Strategy Objective* → *Main Function* → *Business Message* → *Feedback	
If $\forall$Operation Objective (X_1) ($\forall$generates (Operation Objective(X_1), Main Function(X_1))) $\wedge$ $\forall$Strategy Objective (X_2) ($\forall$generates (Strategy Objective(X_2), Main Function(X_2))) If $\forall$Main Function (X_1) ($\forall$out (Main Function Business Message (m))) $\wedge$ $\forall$Main Function (X_2) ($\forall$in (Main Function, Business Message (m)))	**(5)**
$\rightarrow\exists$ Feedback (m)(hasMediatorRelationship (Mediator (X_1), Feedback (m))) $\wedge$ $\exists$ Feedback (m)(hasMediatorRelationship (Mediator (X_2), Feedback (m)))	

Support **and** ***Strategy Objective → Main Function → Business Message → Mean***	
If ∀Support Objective (X_1) (∀generates (Support Objective(X_1), Main Function(X_1))) ∧	**(6)**
∀Strategy Objective (X_2) (∀generates (Strategy Objective(X_2), Main Function(X_2)))	
If ∀Main Function (X_1) (∀out (Main Function, Business Message (m))) ∧	
∀Main Function (X_2) (∀in (Main Function, Business Message (m)))	
→∃Mean (m)(hasMediatorRelationship (Mediator (X_1), Mean (m))) ∧	
∃ Mean (m)(hasMediatorRelationship (Mediator (X_2), Mean (m)))	

Support **and** ***Operation Objective → Main Function → Business Message → Mean***	
If ∀Support Objective (X_1) (∀generates (Support Objective(X_1), Main Function(X_1))) ∧	**(7)**
∀Operation Objective (X_2) (∀generates (Operation Objective(X_2), Main Function(X_2)))	
If ∀Main Function (X_1) (∀out (Main Function, Business Message (m))) ∧	
∀Main Function (X_2) (∀in (Main Function, Business Message (m)))	
→∃Mean (m)(hasMediatorRelationship (Mediator (X_1), Mean (m))) ∧	
∃ Mean (m)(hasMediatorRelationship (Mediator (X_2), Mean (m)))	

To clarify the equations of group 2, the example in Figure IV-12 is made for equation (2). The instance "Choose Partner" of *Strategy Objective* generates the instance "Choose Partner" of *Main Function*. The instance "Sell Product" of *Operation Objective* generates the instance "Sell Product" of *Main Function*. The instance "Order trigger" of *Business Message* is the output message of "Choose Partner" of *Main Function* and the input message of "Sell Product" of *Main Function*. Then the "Order trigger" of *Business Message* is the communication message from strategy process to operation process. The "Mediator 1" belongs to "Sub Network 1", which achieves *Strategy Objective* "Choose Partner". So the "Mediator 1" manages the strategy collaborative process for "Choose Partner". The "Mediator 2" belongs to "Sub Network 2", which achieves *Operation Objective* "Sell Product". So the "Mediator 2" manages the operation collaborative process for "Sell Product". Because *Business Message* "Order trigger" is output of "Choose Partner" and input of "Sell Product", the *Mediator Relationship* "Order trigger" is created and the *hasMediatorRelationship* with "Mediator 1" and "Mediator 2" are created.

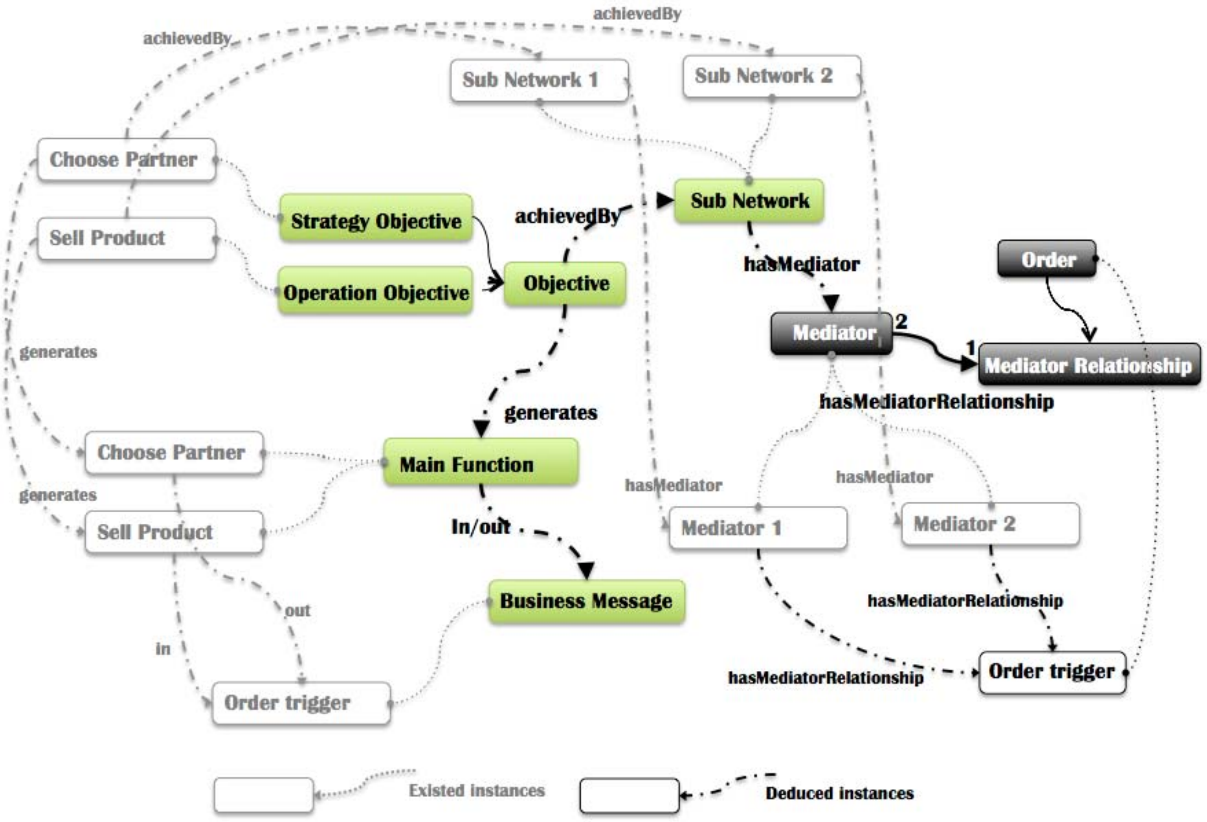


FIGURE IV - 12 EXAMPLE OF GROUP 2

Group 3: Create Generated Mediator Function

The rules in this group are dedicated to deduce the *Generated Mediator Function* and *input/output* relations. From business point of view, the mediator needs mediator functions to invoke partner functions. The input and output messages between mediator functions and partner functions have to be added. This group deals with the creations of mediator functions and input/output messages of mediator functions.

This group contains two equations (Table IV-4). The equation (8) defines how the *Generated Mediator Function* and *Out* relation are created. The equation (9) defines how the *Generated Mediator Function* and *In* relation are created.

For equation (8), if *Function* (X) has input *Business Message (m)*, then *Generated Mediator Function (X)* and relation: *genereatedFrom* between *Function (X)* and *Generated Mediator Function (X)* are created, and the *Generated Mediator Function (X)* has output *Business Message (m)*. For equation (9), if *Function (X)* has output *Business Message (m)*, then *Generated Mediator Function (X)* and relation: *generatedFrom* between *Function (X)* and *Generated Mediator Function (X)* are created, and the *Generated Mediator Function (X)* has input *Business Message (m)*.

TABLE IV - 4 GROUP 3 CREATE GENERATED MEDIATOR FUNCTION

Group 3: Create *Generated Mediator Function*	
Function → In Business Message → Generated Mediator Function → Out Business Message	
∀Function (X) (∀In (Function (X), Business Message (m)))	**(8)**
→∃ Generated Mediator Function (X) (Out (Generated Mediator Function (X), Business Message (m))) ∧ ∃ Generated Mediator Function (X) (generatedFrom (Function (X), Generated Mediator Function (X)))	

Function → Out Business Message → Generated Mediator Function → In Business Message	
∀Function (X) (∀In (Function (X), Business Message (m)))	**(9)**
→∃ Generated Mediator Function (X) (Out (Generated Mediator Function (X), Business Message (m))) ∧ ∃ Generated Mediator Function (X) (generatedFrom (Function (X), Generated Mediator Function (X)))	

The example of equation (8) and (9) is presented in Figure IV-13. The "Payment" and "Order" are the instances of *Business Message*. The "Place Order" and "Receive Payment" are the instances of *Function*. The "Payment" is the input of "Receive Payment". The "Order" is the output of "Place Order". According the equations, the "Place Order" and "Receive Payment" are generated to *Generated Mediator Function*: "Invoke Place Order" and "Invoke Receive Payment". The "Payment" is the output message of "Invoke Receive Payment". The "Order" is the input message of "Invoke Place Order".

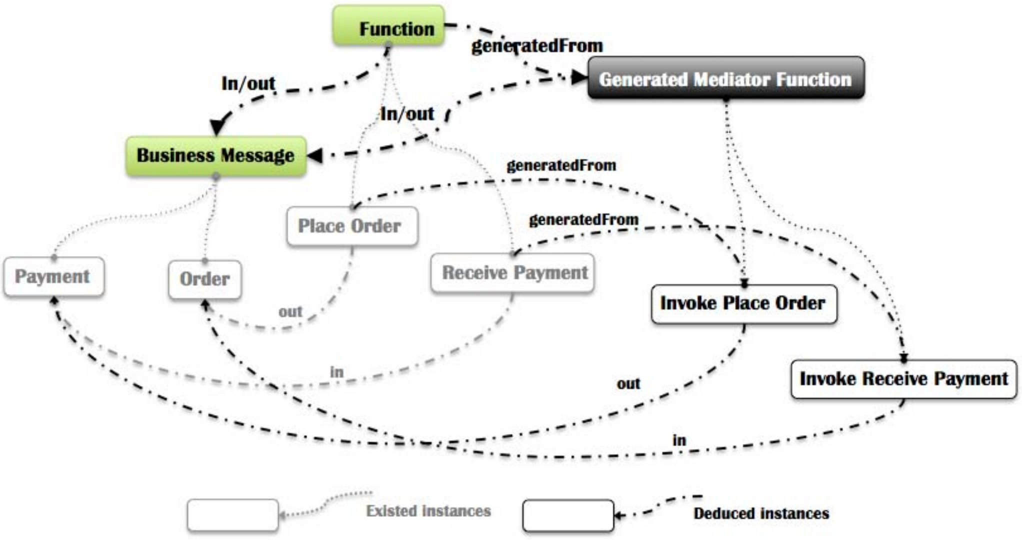


FIGURE IV - 13 EXAMPLE OF GROUP 3

Group 4: Link Generated Mediator Function to Mediator

This group is dedicated to deduce the relation: *hasGeneratedFunction* between *Generated Mediator Function* (deduced by group 3) and *Mediator*. From business view, to deduce the collaborative process, we have to know which mediator functions belong to which mediator, so that the mediator can use its functions to invoke partner functions.

This group of transformation rules is based on one assumption. The assumption is that the relation: *owns* already exists in the collaborative ontology. But in the collaborative ontology and the gathered knowledge, the relation does not exist. The section IV.1 of chapter IV explains how is the relation: owns created without transformation rules.

The equation (10) of this group (Table IV-5) defines the deduction method. If *Main Function (X)* has *Function (X_1)*, then the *Mediator (X)*, which manages the *Sub Network* of *Main Function (X)*, has the *Generated Mediator Function (X_1)*, which is generated from *Function (X_1)*.

Table IV - 5 Group 4 Link Generated Mediator Function to Mediator

Group 4: Link *Generated Mediator Function* to *Mediator*	
Main Function → Function → Mediator → Generated Mediator Function	
$\forall$Main Function (X) ($\forall$owns (Main Function (X), Function (X_1)))	**(10)**
→$\exists$ Mediator (X) ($\forall$hasGeneratedFunction (Mediator (X), Generated Mediator Function (X_1)))	

Figure IV-14 illustrates the example of the equation (10). The *Main Function* “Sell Component” is generated from the *Operation Objective* “Sell Component”, which is achieved by *Sub Network* “Network 1”. The *Sub Network* “Network 1” is manages by *Mediator* “Mediator 1”. So the “Mediator 1” is for the *Main Function* “Sell Component”. Because the *Main Function* has *Function* “Place Order” and “Deliver Product” and the “Invoke Place Order” and “Invoke Deliver Product” are generated from “Place Order” and “Deliver Product”, the *Generated Mediator Function* “Invoke Place Order” and “Invoke Deliver Product” belong to the “Mediator 1”. So the two relations: *hasGeneratedFunction* are created between “Mediator 1” and “Invoke Place Order”/“Invoke Deliver Product”. Additionally, the question mark on relation: *owns* is to represent that the instances of the relation are not created in the ontology. The section IV.1 of chapter IV provides a methodology to created the relations.

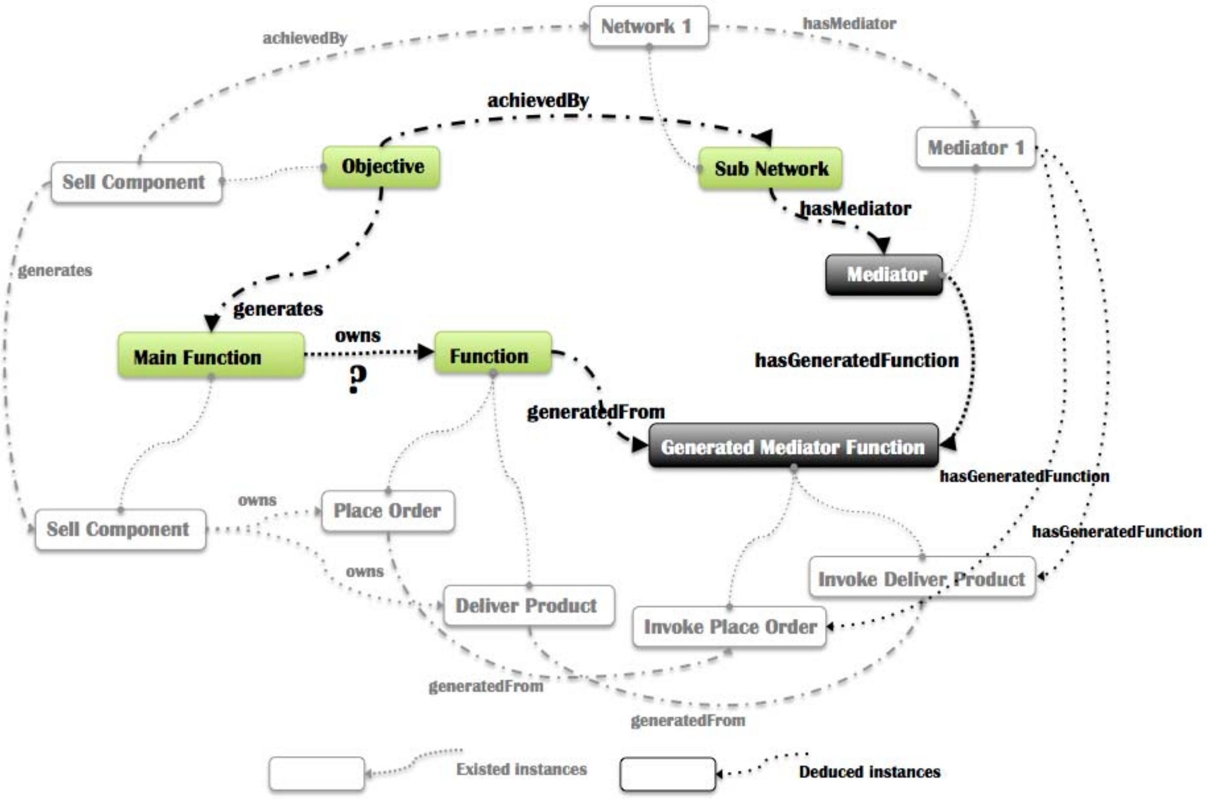


FIGURE IV - 14 EXAMPLE OF GROUP 4

Group 5: Create Inter Mediator Function

This fifth group concerns the deductions of *Inter Mediator Function.* As explained in the group 2, one collaborative situation may have several mediators. There are communications among mediators. This group helps to identify specific the mediator function, which is communicated with other mediators.

Table IV-6 represent the equation (11) of the group 5. If *Generated Mediator Function (X)* of *Mediator (Y)* has the *Business Message (m)*, which generates *Mediator Relationship (m)* of *Mediator (Y)* by equation (2) or (3) of the group 2, then *Event (m)* is created to achieve *Mediator Relationship (m)* and *Generated Mediator Function (X)* is also an *Inter Mediator Function (X).* If the Business *Message (m)* is the input message of *Generated Mediator Function (X)*, then the *Event (m)* is also the input of *Inter Mediator Function (X).* If the *Business Message (m)* is the output message of *Generated Mediator Function (X)*, then the *Event (m)* is also the output of *Inter Mediator Function (X).*

TABLE IV - 6 GROUP 5 CREATE INTER MEDIATOR FUNCTION

Group 5: Create *Inter Mediator Function*	
Generated Mediator Function → Mediator Relationship → Event → Inter Mediator Function	
If ∀Generated Mediator Function (X) (In/out (Generated Mediator Function (X), Business Message (m))) ∧ ∃ Mediator Relationship (m) (hasMediatorRelationship (Mediator (Y), Mediator Relationship (m))) ∧ Mediator (Y) (hasGeneratedFunction (Mediator (Y), Generated Mediator Function (X)))	**(11)**
→∃ Event (m) (achievedBy (Mediator Relationship (m), Event (m))) ∧ ∃ Inter Mediator Function (X) (In/out (Event (m), Inter Mediator Function (X)))	

Figure IV-15 defines an example for the equation (11). The *Generated Mediator Function* "Invoke wait for order" belongs to "Mediator 1" and has input *Business Message* "Order trigger", which generates the "Order trigger" of *Mediator Relationship* by equation (2) or (3). So the "Invoke wait for order" is also an *Inter Mediator Function*. The *Event* "Order trigger" is created to achieve the *Mediator Relationship*. Because "Order trigger" of *Business Message* is the input of "Invoke wait for order", the *Event* "Order trigger" is the input of "Invoke wait for order".

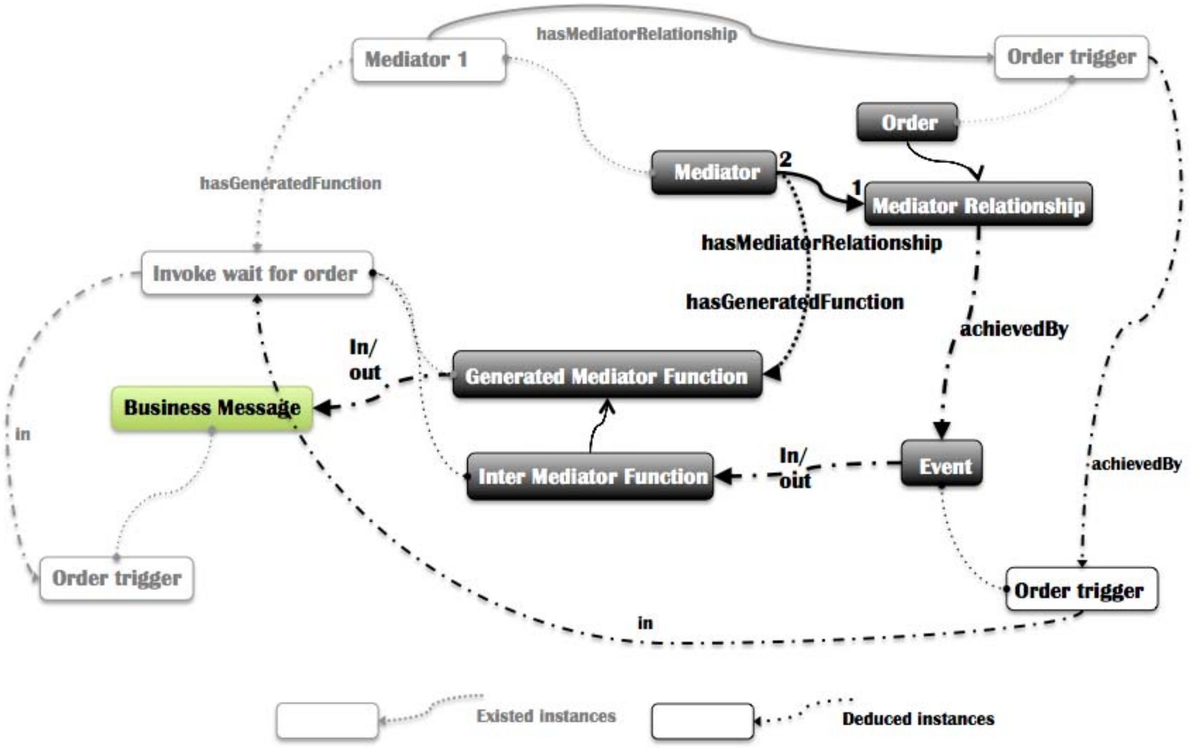


FIGURE IV - 15 EXAMPLE OF GROUP 5

IV. Service Selection and Sequence Deduction

In this section, the problems, which cannot solve by the collaborative ontology, are addressed in detail. The first one is the selection of business service (or to add the relation: *owns* in Figure IV-14). The solution of this problem is explained in section IV.1. The second one is the deduction of collaborative process. Section IV.2 explains the deduction rules of collaborative process and the deduction methods of sequences and gateways. To better explain the two solutions, an example is built in Figure IV-16.

Figure IV-16 upper left part presents the initial collaborative situation of example. There are three partners: client, factory and subcontractor in the collaboration. The client places order to factory, which makes decision about outsourcing order for the subcontractor. Concerning product delivery, factory provides client and subcontractor a support service of transportation and storage.

The first step of objective model is to model the common goals in this collaboration. As shown in Figure IV-16 upper right hand side, there are two main objectives (ellipses in Figure IV-16): selling products and outsourcing production are defined as the common goals of the whole collaborative network. Second step is to define the sub-networks to achieve each main objective. For example, for outsourcing production, partners: factory and subcontractor are grouped to cooperate as a sub-network. If the collaborative situation is complex, step 1 and step 2 could be repeated several times. Finally, we model the individual objectives of each partner (for example, on the collaborative network of Figure IV-16 the right hand side, subcontractor owns two objectives: delivery support and selling products).

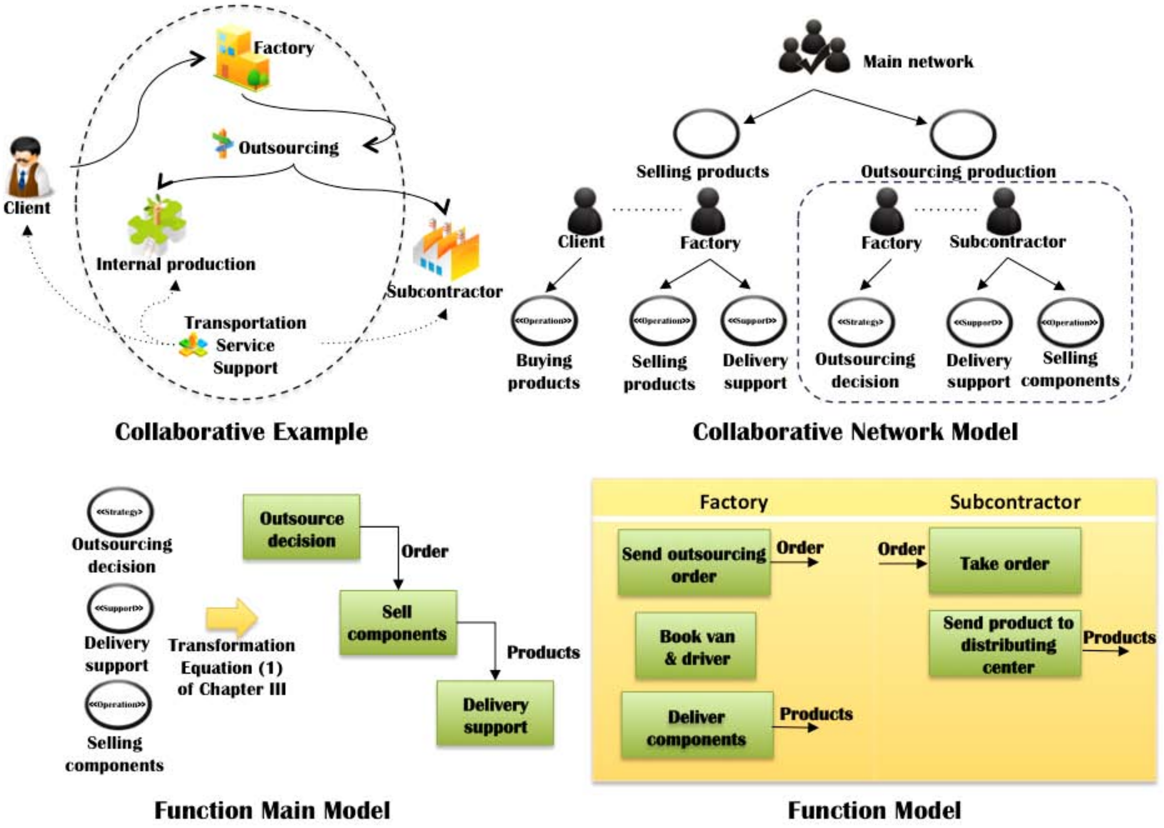


FIGURE IV - 16 EXAMPLE OF CHAPTER IV

After the definition of objective model, we can start transformation of function main model and the definition of functional table. Here, we use outsourcing production sub-network in objective model (black dash-line box in Figure IV-16) to show how do we use transformation equation 1 of chapter III and how do users provide their business functions. Figure IV-16 the left hand side at the bottom is an example for transformation Equation (1) of chapter III. If a sub-network exists, then a function main model exists. All the defined objectives in the sub-network are seen as main functions and put in function main model. In the example, objective: “Outsourcing decision”, “Delivery support” and “selling products” are transferred to main function. User defines controlling messages “Order” and “Products”. Main function “Outsource decision” sends controlling message “Order” to trigger main function “Sell components”. Receiving controlling message “Products” from main function “Sell components” launches Main function “Delivery support”. User provides the controlling messages among these functions. Figure IV-16 the right hand side at the bottom presents functional table, each partner fills correspondence column with the list of functions (with input and output message).

IV.1. Business Service Selection

The principle of business service selection is based on the collaborative ontology, which have been introduced in last section. In the ontology, there are large numbers of *Objective* and *Function* instances with relationship: *achievedBy*. If we could link business objectives and business function in the model to *Objective* and *Function* instances in collaborative ontology, then we could indirectly link business objective to business function by the help of relationship: *achievedBy*. With above theory, we can complete business service selection task.

As shown in Figure IV-17, there are two parts: ontology and model. In ontology part, we choose several instances of *Function* and *Objective*. In model part, we take the objective and function model example in introduction of this section. Here for each business objective and function in the model, we want to find same or close instances in ontology. For example, in the model, “Book van & driver” is same or close to “Book transportation” in the ontology, then we make a link: *Same As/Near By* between them.

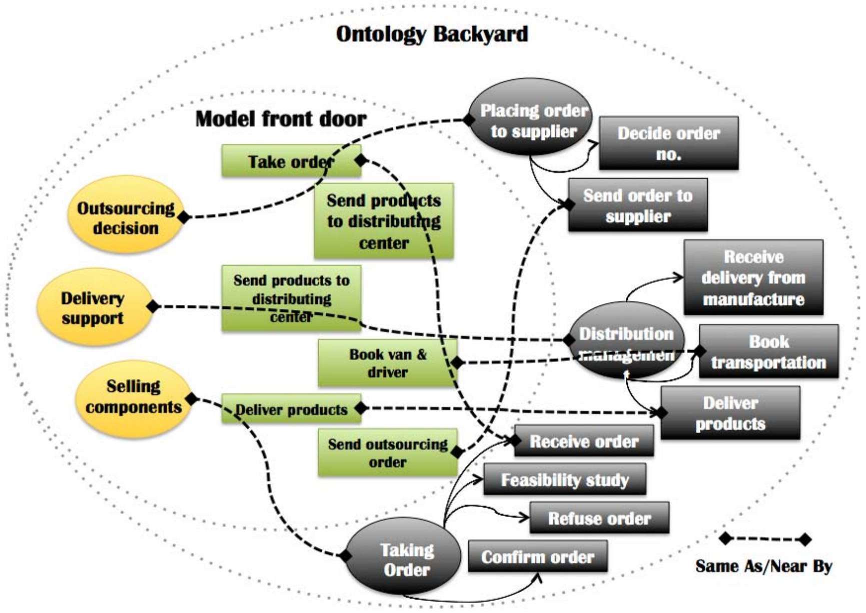


FIGURE IV - 17 INSTANCES OF COLLABROATIVE ONTOLOGY AND MODELS

With all the relationship: *Same As/Near By* in Figure IV-17, suitable business functions are selected for business objectives. For example, “Outsourcing decision” is same or close to “Placing order to supplier”. “Send outsourcing order” is same or close to “Send order to supplier”. Because “Placing order to supplier” is achieved by “Send order to supplier”, “Outsourcing decision” can be achieved by “Send outsourcing order”. Figure IV-18 shows all the resulting additional relationship. Business objectives is linked to business functions by relationship: *achievedBy*.

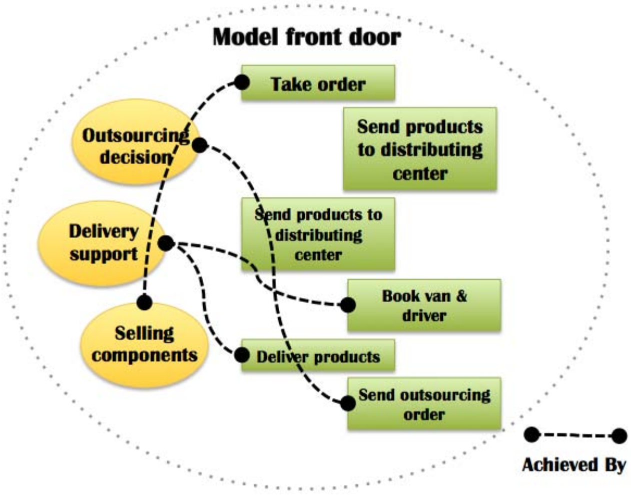


FIGURE IV - 18 RESULTS OF DEDUCTION

Even though the basic principal of business service selection is defined, there are still some remarks to consider:

- Making relationship: Same As/Near By for each business objective and function is quite hard for user. Because there may be instances with various names. We provide an Instance Suggestion Mechanism, which could provide suggested ontology instance for user. The mechanism is based on syntactic. Algorithm IV-1 presents the Instance Suggestion Mechanism algorithm.
- Business objectives and functions defined in the model could also be seen as Business and Function instances in collaborative ontology for future uses. There should be a self-update mechanism to enlarge the collaborative ontology. Algorithm IV-3 explains self-updating mechanism.

Instance Suggestion Algorithm

Instance suggestion algorithm deals with selecting same or nearest ontology Objective or Function instances for each business objective and function. The algorithm takes ontology Objective instances suggestion as example (ontology Function instances suggestion uses the same algorithm structure.). Algorithm (1) takes syntactic keyword of business objective as input, uses collaborative ontology as data and provides a list of suggested ontology instances. This algorithm has three main parts:

- Line 3: finding an Objective instance in collaborative ontology which owns the same keyword: $\text{objective}_{\text{key}}$ as business objective, the instance is added to suggestion list: $L_{\text{suggestion}}$;
- Line 6-Line 17: Taking frontal parts of keyword as a new list of keyword: $L_{\text{word}}[1]$ to $L_{\text{word}}[i]$ (for example, keyword: “send products to distributing center”, new keywords: “send products to distributing” and “send products to”), for each new keyword, if finding an Objective instance’s keyword in collaborative ontology which starts with or contains the new one or contains, then the Objective instance is added to suggestion list;
- Line 18-Line 23: Taking related two words, which are contained in keyword as a new list of keyword: L_{2words} (for example, keyword: “send outsourcing order”, new keywords: “send outsourcing” and “outsourcing order”), for each related two words, if finding an Objective instance’s keyword, which contains the two words, then the Objective instance is added to suggestion list.

ALGORITHM IV - 1 INSTANCE SUGGESTION ALGORITHM

Algorithm (1) Instance Suggestion: provide suggested collaborative ontology instances for business objective.

Input: $\text{objective}_{\text{key}}$, keyword for the business objective
Data: Collaborative Ontology: CO
Output: $L_{\text{suggestion}}$, list of suggested ontology instances

```
1  $A_{objective}$, array of all Objective instances of CO;
2  $L_{suggestion}$ ← Null;
3  if $A_{objective}$ contains objective.keyword = $Objective_{key}$ then
4      $L_{suggestion}$ adds objective;
5  else
6      $L_{word}$, list of words contained in $objective_{key}$;
7      i, counter for loop ← $L_{word}$.length;
8      word, store a part of keyword ← Null;
9      for i from $L_{word}$.length to 3
10         word = from $L_{word}$[1] to $L_{word}$[i] ;
11         if $A_{objective}$ contains objective.keyword starts with word then
12             $L_{suggestion}$ adds objective;
13         end;
14         if $A_{objective}$ contains objective.keyword contains word then
15             $L_{suggestion}$ adds objective;
16         end;
17     end
18     $L_{2words}$, list of 2 words contained in $Objective_{key}$;
19     while $L_{2words}$ has next element: word do
20         if $A_{objective}$ contains objective.keyword contains word then
21             $L_{suggestion}$ adds objective;
22         end;
23     end;
24 end;
25 return $L_{suggestion}$;
```

Objective-Function Mapping

Objective-Function mapping algorithm is the main part of business service selection. The principle has been explained above. As shown in Algorithm (2), it takes list of business objectives and list of business functions as input, uses collaborative ontology as data, and outputs list of relationships: achievedBy. The algorithm is explained as followed:

- Line 3-Line 5: starts the mapping from business functions side. If one business function: $E_{function}$ owns relationship: Same As/Near By with one ontology Function instance: $O_{function}$.
- Line 6-Line 7: if $O_{function}$ owns relationship: achievedBy with one ontology Objective instance: $O_{objective}$, and if $O_{objective}$ owns relationship: Same As/Near By with business objective: $E_{objective}$, then as result: $E_{objective}$ has relationship: achievedBy with $E_{function}$.
- Line 8: the relationship is added into the list: $L_{achievedBy}$.
- Line 13 and Line 16: if there is an $E_{function}$, which doesn't find $E_{objective}$, then a relationship: achievedBy from Null to $E_{function}$ is created. The relationship is added to $L_{achievedBy}$.
- Line 21: if an $E_{objective}$ is never achieved, then a relationship: achievedBy from $E_{objective}$ to Null is created. The relationship is added to $L_{achievedBy}$.

Algorithm IV - 2 Objective-Function mapping

Algorithm (2) Objective-Function mapping: find correspondence business functions for each business objective and create relationship: achievedBy.

```
Input: $L_{objective}$, list of business objectives
       $L_{function}$, list of business functions
Data: Collaborative Ontology: CO
Output: $L_{achievedBy}$, list of relationship
1  $L_{achievedBy}$ ← Null;
2  $L_{relatedObjectives}$, list of objectives with achievedBy ← Null;
3  while $L_{function}$ has next element: $E_{function}$ do
```

```
    if $E_{function}$.sameas/nearby!=null then
        $O_{function}$ = $E_{function}$.sameas/nearby;
        $O_{objective}$ = $O_{function}$.achievedby;
        if $L_{objective}$ contains element $E_{objective}$.sameas/nearby = $O_{objective}$ then
            $L_{achievedBy}$ adds achievedBy($E_{objective}$, $E_{function}$);
            if $L_{relatedObjective}$ doesn't contain $E_{objective}$ then
                $L_{relatedObjective}$ adds $E_{objective}$;
            end;
        else
            $L_{achievedBy}$ adds achievedBy(null, $E_{function}$);
        end;
    else
        $L_{achievedBy}$ adds achievedBy(null, $E_{function}$);
    end;
end;
while $L_{objective}$ has next element: $E_{objective}$ do
    if $L_{relatedObjective}$ doesn't contain $E_{objective}$ then
        $L_{achievedBy}$ adds achievedBy($E_{objective}$, null);
    end;
end;
return $L_{achievedBy}$;
```

Ontology Updating

Ontology updating algorithm deals with inserting business objectives and functions in collaborative ontology as Objective and Function instances with relationship: Same As/Near By. Because algorithms of Objective and Function updating are similar, here Algorithm (3) uses objective updating as example. It creates new ontology instance for each $E_{objective}$, creates relationship: Same As/NearBy and adds them in collaborative ontology. As shown in algorithm (3):

- Line 2 and Line 3: for each $E_{objecitve}$, if $E_{objective}$ owns relationship: Same As/Near By, then get $O_{objective}$ which is related to $E_{objective}$.
- Line 4 and Line 5: create new ontology instance: O_{new} for $E_{objective}$ and add O_{new} into collabroative ontology.
- Line 6: creates new Relationship: Same As/Near By between O_{new} and $O_{objective}$, and adds the relationship to collaborative ontology also.

ALGORITHM IV - 3 ONTOLOGY UPDATING

Algorithm (3) Ontology Updating: insert business objectives into ontology as instances and created relationship: Same As/Near By.

```
Input: $L_{objective}$, list of business objectives
Data: Collaborative Ontology: CO
1  while $L_{objective}$ has next element: $E_{objective}$ do
2      if $E_{objective}$.sameas/nearby!=null then
3          $O_{objective}$ = $E_{objective}$.sameas/nearby;
4          $O_{new}$ = change $E_{objective}$ to ontology instance;
5          CO adds $O_{new}$;
6          CO adds SameAs/NearBy($O_{objective}$, $O_{new}$);
7      end;
8  end;
```

IV.2. Process Sequence Deduction

In this section, the example at the beginning of chapter IV is taken to present the deduction of collaborative process. The deduction methods are cut into four parts: deduction of process cartography, deduction of collaborative process, deduction of partner pool and deduction of mediator pool. The four parts of deduction methods are presented in section IV.2.1. The section IV.2.2 presents the complementary method to add sequences and gateways to the collaborative process.

IV.2.1. Process Deduction

Part 1: Deduction of process cartography

The deduction of process cartography presents how the collaborative process cartography (presented in section IV of chapter III) is extracted from the collaborative ontology. Figure IV-19 represents the example of deduction of process cartography. There are four rules in this part of deduction.

- No.1-1: *Network/Sub Network* → Collaborative process cartography. If the *Network* or *Sub Network* has collaborative objectives, then there exists a correspondence collaborative cartography for the *Network* or *Sub Network*. In Figure IV-19, the "Main Network" is the instance of *Network*, which defines three main collaborative objectives. The "Main Network" is seen as a sign for the deduction of the collaborative process cartography, which means there exists the collaborative process cartography.
- No.1-2: *Objective* → Strategy/Operation/Support/General pool. Based on rule No.1-1, rule No.1-2 fills the collaborative process cartography with different types of pools (strategy, operation and support). If the *Network* has strategy objective, then there is one strategy pool. If the *Network* has operation objective, then there is one operation pool. If the *Network* has support objective, then there is one support pool. If the *Network* has objective, then there is one general pool. In Figure IV-19, the "Main Network" has the strategy objective: "Outsourcing decision", the operation objective: "Sell products" and the support objective: "Delivery support". The deduced collaborative process cartography has the strategy, operation and support pool.
- No.1-3: *Main Function* → Task. This rule fills the strategy/operation/support/general pool with tasks. These tasks are extracted from *Main Functions*, which are generated from collaborative objectives. In Figure IV-19, the *Main Function* "Outsourcing decision" is generated from strategy objective, the *Main Function* "Outsourcing decision" is transferred to the task: "Outsourcing decision" in strategy pool. The *Main Function* "Sell products" is generated from operation objective, the *Main Function* "Sell products" is transferred to the task: "Sell products" in operation pool. The *Main Function* "Delivery support" is generated from support objective, the *Main Function* "Delivery support" is transferred to the task: "Delivery support" in support pool.
- No.1-4: *Business Message* → Message flow. This rule deals with the transformation of message flow. The "Order" and "Products" are the instance of *Business Message*. The "Order" is the output of *Main Function* "Outsourcing decision" and the input of *Main Function* "Sell products". The "Order" is transferred to message flow between task "Outsourcing decision" and "Sell products".

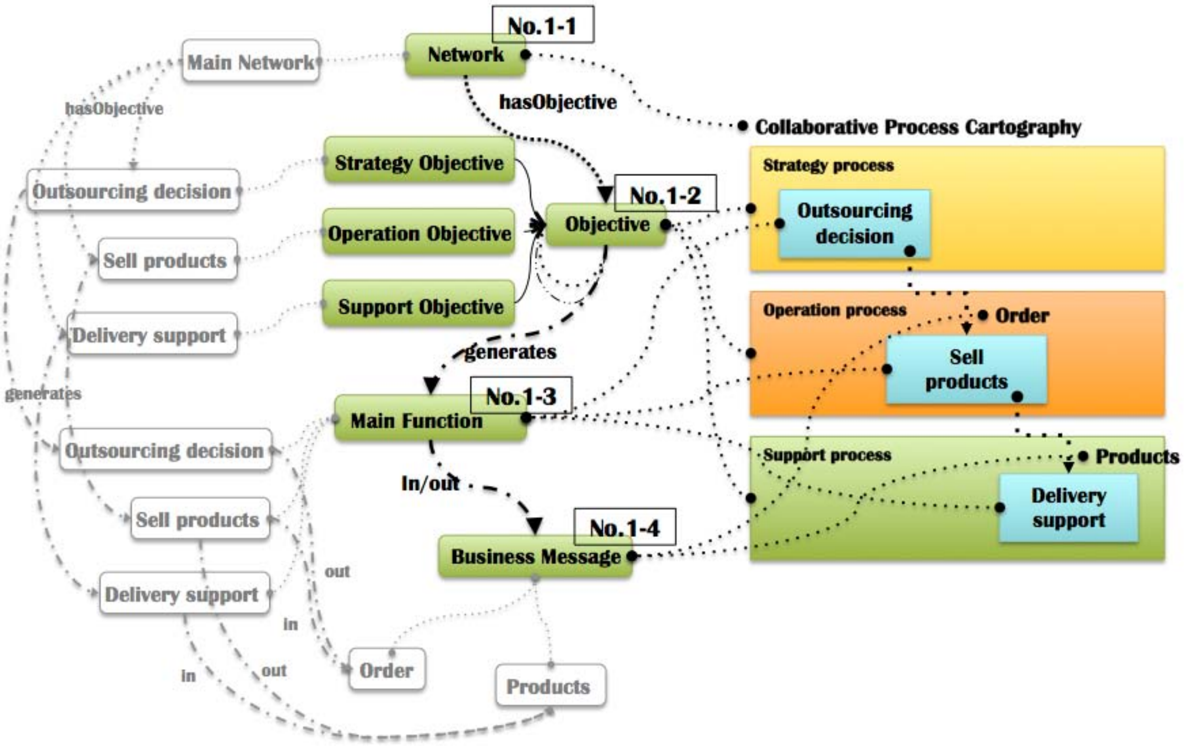


FIGURE IV - 19 DEDUCTION OF COLLABORATIVE PROCESS CARTOGRAPHY

Part 2: Deduction of collaborative process

Part 2 defines the rules of deduction of collaborative process (presented in section IV of chapter III). In the process cartography deduced as explained in Part 1, there are main tasks. For each task, there is a sub process, which presents the collaborative process of the task in detail. Part 2 shows how the sub collaborative process is transferred. Figure IV-20 represents the example of deduction of collaborative process. There are three rules in this part of deduction.

- No.2-1: *Main Function* → Collaborative process. If there is a *Main Function*, which has been transferred to task in collaborative process cartography, then there is a detailed collaborative process. For example, in Figure IV-20, there is *Main Function* “Delivery support”, which is transferred into the task of support pool in Figure IV-19. For this *Main Function*, there is a detailed collaborative process.
- No.2-2: *Partner* → Partner pool. Based on rule No.2-1, rule No.2-2 fills the collaborative process with partner pools. If the *Main Function* is generated from one *Objective*, which is achieved by *Sub Network*, the *Partners* of the *Sub Network* are transferred as partner pools in the collaborative process. In Figure IV-20, the “Sub Network 1” contains Partners: “Subcontractor” and “Factory”. The “Subcontractor” and “Factory” is transferred to subcontractor and factory pools in the Figure IV-20.
- No.2-3: *Mediator* → Mediator pool. If the *Sub Network* has one *Mediator*, then the *Mediator* is transferred into the mediator pool in the collaborative process. In Figure IV-20, the “Sub Network 1” has *Mediator* “Support Mediator”. The “Support Mediator” is transferred into the support mediator pool in the collaborative process of delivery support.

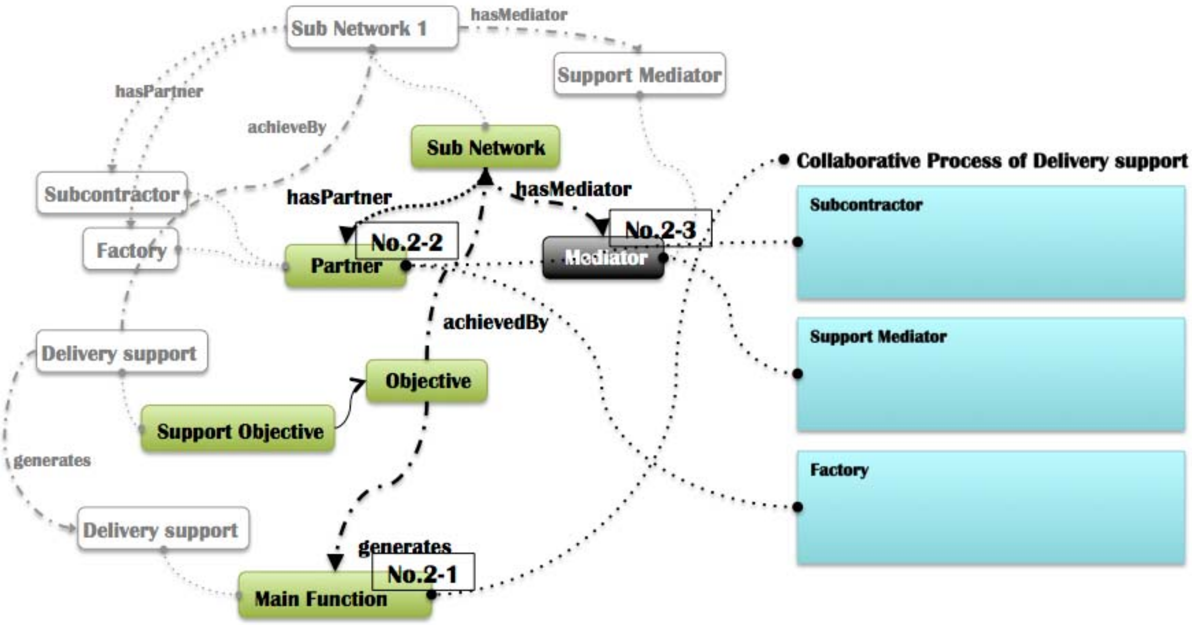


FIGURE IV - 20 DEDUCTION OF COLLABORATIVE PROCESS

Part 3: Deduction of partner pool

The rule No. 2-2 has deduced the partner pools of the collaborative process. But there is no task in these pools. Part 3 of the deduction rules is dedicated to fill the partner pools with tasks and message flows. Figure IV-21 represents the example of deduction of the tasks. There are two rules in this part of deduction.

- No.3-1: *Function* → Task in partner pool. This rules transfers all the functions of each partner that are defined to achieve the collaborative objective. If there is one *Function*, which is shared by one *Partner*, then the *Function* is transferred into task of the partner pool. For example, in Figure IV-21, the *Function* "Send products to distributing center" belongs to the *Partner* "Subcontractor". The *Function* "Send products to distributing center" is transferred to the task of subcontractor pool in the collaborative process. The *Function* "Book van & driver" and "Deliver products" belongs to the *Partner* "Factory". The *Function* "Book van & driver" and "Deliver products" are transferred to the tasks of the factory pool in the collaborative process.
- No.3-2: *Business Message* → Message flow. Based on rule No.3-1, rule No.3-2 adds messages flow to the tasks. If one *Function* has input/output *Business Message*, then the *Business Message* is transferred to the input/output message flows of the task, which is transferred from the *Function*. In the *Function* IV-21, the "Products" is the instance of *Business Message*. It is the output of "Send products to distributing center" and the input of "Deliver products". The "Products" is transferred to the output message flow of the task "Send distributing center" and the input message flow of the task "Deliver products".

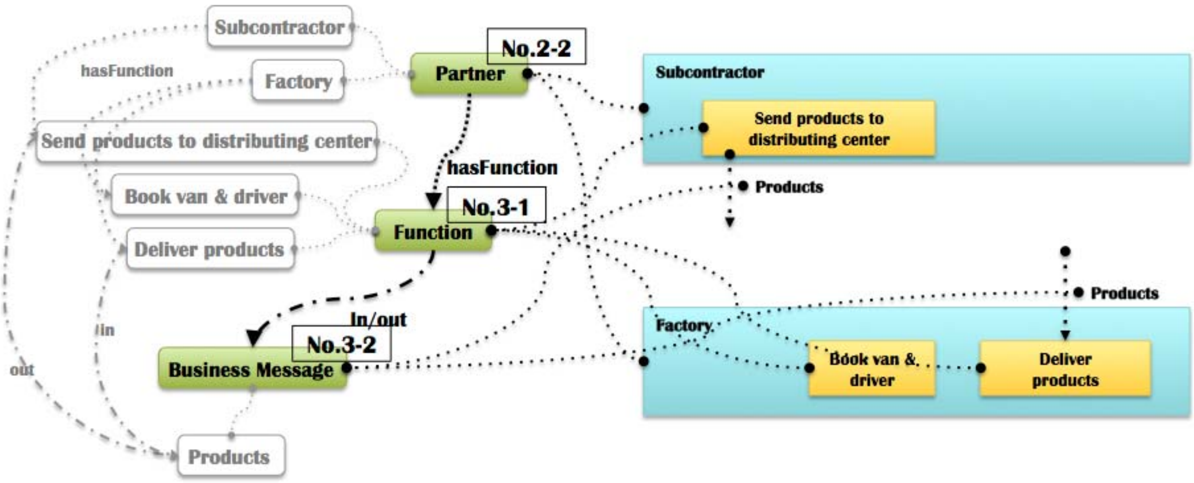


FIGURE IV - 21 DEDUCTION OF PARTNER POOL

Part 4: Deduction of mediator pool

The rule No. 2-3 has deduced the mediator pool. The part 4 fills the mediator pool with mediator tasks, message flows and events. Figure IV-22 represents the example of deduction of mediator tasks, message flows and events. There are three rules in this part of deduction.

- No.4-1: *Generated Mediator Function* → Task in the mediator pool. This rules transfers all the functions of mediator that are defined to achieve the collaborative objective. If the *Mediator* of the collaborative process has *Generated Mediator Function*, then the *Generated Mediator Function* is transferred to the tasks in the mediator pool. In Figure IV-22, the "Invoke Deliver products" and "Invoke Send products to distributing center" are the instances of *Generated Mediator Function*. They belong to the "Support Mediator". So these two *Generated Mediator Functions* are transferred to the mediator tasks in the mediator pool.
- No.4-2: *Business Message* → Message flow. Based on rule No.4-1, rule No.4-2 adds the message flows to the mediator functions. If one *Business Message* is the input/output of the *Generated Mediator Function*, then the *Business Message* is transferred to the input/output message flow of the mediator task, which is transferred from the *Generated Mediator Function*. Figure IV-22 shows that the "Products" is the output of the "Invoke Deliver products". So the "Products" is transferred to the input message flow of the "Invoke Deliver products" task in the mediator pool.
- No.4-3: *Event* → Start/Intermediate/End message event. If there is one *Event*, which is the input or output of one *Inter Mediator Function*, then the *Event* is transferred to start/intermediate/end message event before or after the task, which is transferred from the *Inter Mediator Function*. For example, the *Event* "Products" is the input event for the "Invoke Send products to distributing center". So the event is transferred to the start event before the "Invoke Send products to distributing center".

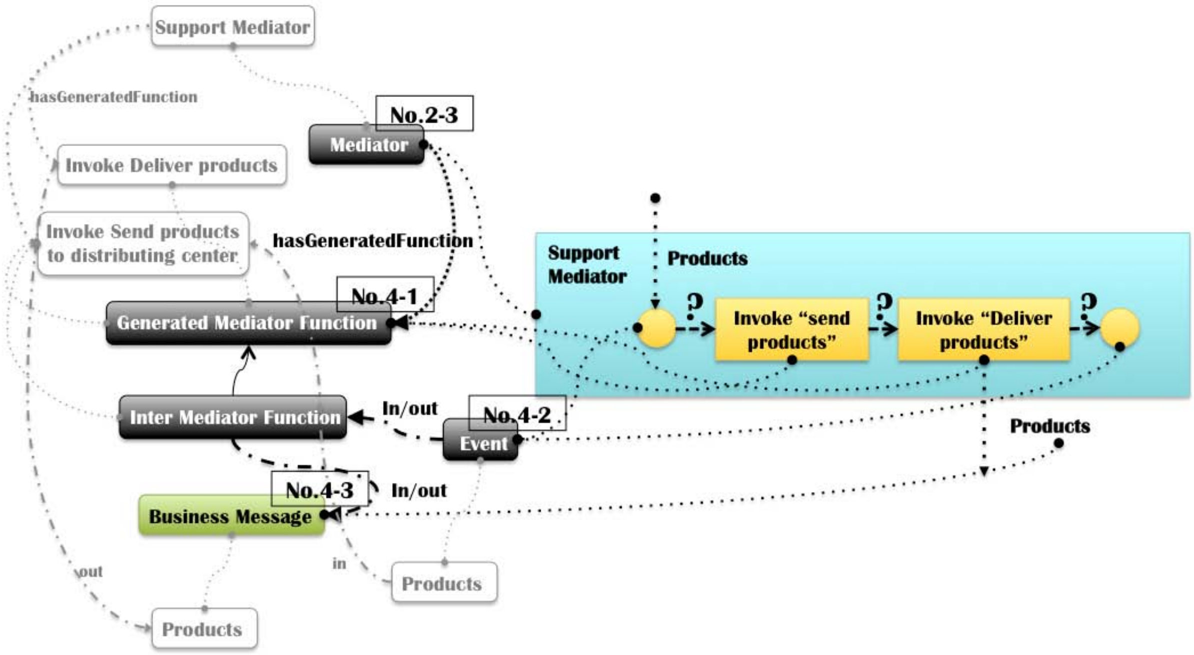


FIGURE IV - 22 DEDUCTION OF MEDIATOR POOL

The four parts of deduction rules deduce the collaborative process cartography from the collaborative ontology. But in the Figure IV-22, there are question marks on the sequence flow. In fact, the rules can't deduce the sequences. The sequences among the mediator tasks are still problems. In next section, the methodology of sequence deduction is presented.

IV.2.2. Sequence Deduction

In (Rajsiri, 2010), the sequence flow is deduced by linking the functions/tasks with input/output messages and then by "cleaning the deduced model manually". This method leads to a problem. As shown in Figure IV-23, on the top, there are functions and input/output messages of the functions. If the functions are linked together by input/output messages (at the bottom of Figure IV-23), the output "M2" of "F1" is the inputs of the "F3" and the "F4". There must be a gateway to manage the fork output sequence flows. In this condition, it is sure that there must be a gateway, but the type of gateway cannot be decided. There is also a potential problem. The linkage of input/output messages easily creates loops of functions. If the loops come out, it would be quit difficult to solve. Furthermore, it creates a lot of useless connections.

The linkage of input/output messages is useful when the functions are linked one by one as a line. If there are forks, loops and useless connections in the process, another solution has to be developed. For the forks, loops and useless connections in the collaborative process, a capability objective based method is developed. The "F3", "F4" and "F6" (in the dot line box of Figure IV-23) are taken as example.

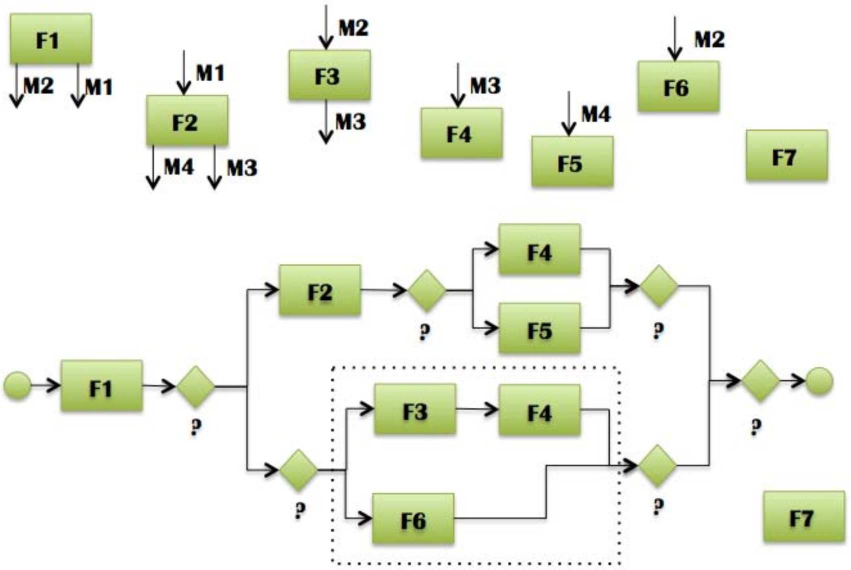


FIGURE IV - 23 DEDUCE SEQUENCES BY INPUT/OUTPUT MESSAGES

As shown in Figure IV-24, there are a list of functions and a list of objectives. There is one main objective: "O1". The main objective has sub objectives. The "O2" and "O3" are the sub objectives of "O1". All the functions are linked to the objectives. These linkages tell the functions which can achieve the objectives. To achieve "O1", there must functions, which achieve "O1" directly, or the functions, which achieve "O2" and "O3". So to obtain "O1", the function "F6" or the functions "F3" and "F4" are needed (the equation in Figure IV-24). By this way, the clue of gateways is found.

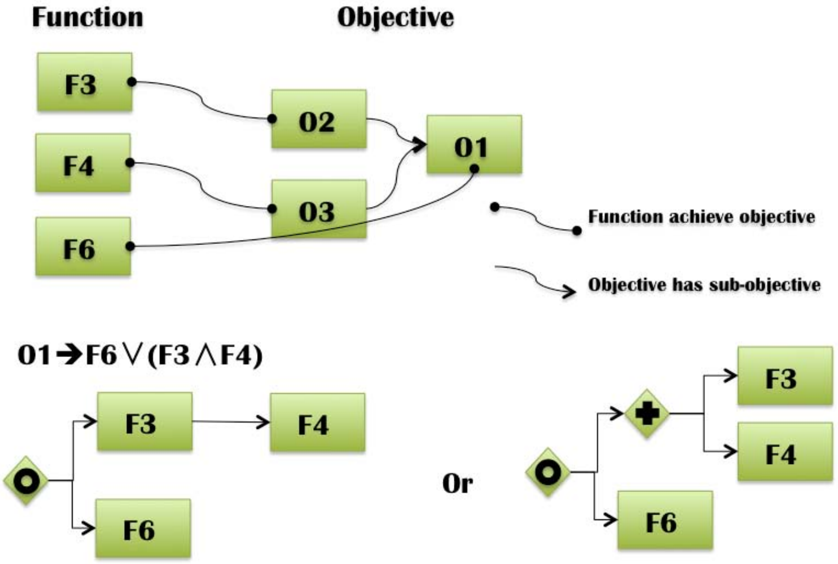


FIGURE IV - 24 LINK FUNCTIONS WITH OBJECTIVES

As shown at the bottom of Figure IV-24, two results are possible. First of all, the "F6" can obtain "O1". Or the "F3" and "F4" can obtain "O1". For both solutions, the inclusive gateway is used to launch "F6" or "F3" and "F4". But for "F3" and "F4", they can be invoked one after another or at the same time. For first result (the left hand side of Figure IV-24), the "F3" is invoked before "F4". For the second result, the parallel gateway is added to invoke the two functions at the same time. But in Figure IV-23, the output message of "F3" is the input message of "F4", so the first solution is taken.

Figure IV-25 illustrates an example of sequence deduction. There are functions: “Deliver goods”, “Receive goods” and “Close deal”. There is objective: “Delivery”, which has sub objectives: “Delivery success” and “Finish delivery”. “Deliver goods” and “Receive goods” can achieve objective “Delivery success”. “Close deal” can achieve “Finish delivery”. On the bottom of Figure IV-25, there are three solutions. Because the input of “Deliver goods” is the output of “Receive goods”, “Deliver goods” is before “Receive goods”. But the order of “Close deal” is un-known. There are three possibilities (manually, the solution in the dot line box should be selected).

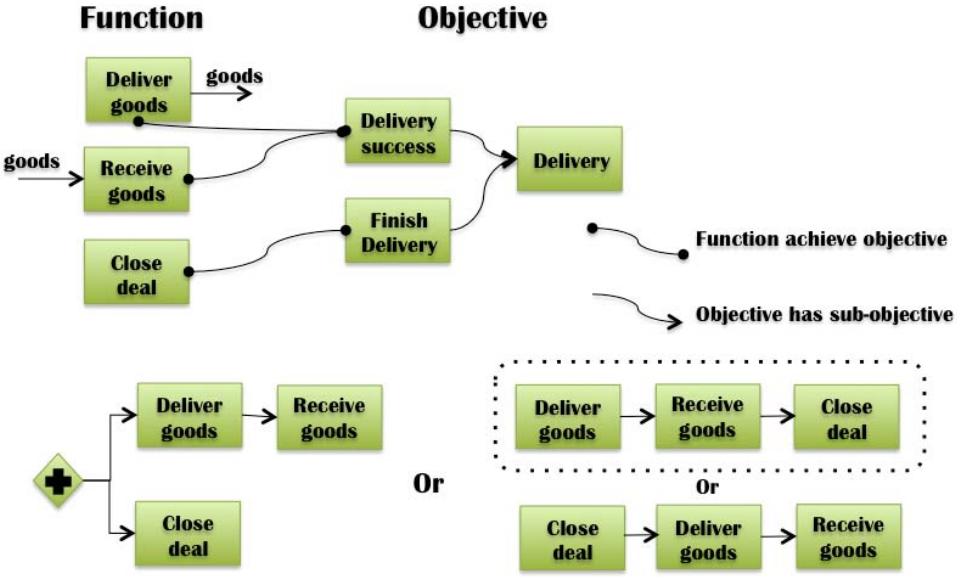


FIGURE IV - 25 EXAMPLE OF PROCESS DEDUCTION

In this research work, the linkage of messages and the objective based method are mixed to deduce the sequences and the gateways. First, the linkage of messages is used to get at global picture of the process. Second, for the special place (the gateways are needed) of the global picture is taken and re-done by using the objective based method. Finally, the linkage of messages checks the results of objective based method to get the best solution.

V. Conclusion

This chapter has presented the collaborative ontology of MISE 2.0 (CO2). The collaborative ontology contains the transformation rules from collaboration concepts to mediation concepts. Tow supplementary methods (business service selection and process sequence deduction) are provided to the remained problems of collaborative ontology and transformation rules.

The collaborative ontology of MISE 1.0 (CO1) is also presented in this chapter. They are both collaborative ontology. There might be some questions or confusions: i) what are the key elements for the collaborative ontology? ii) What are the same concepts in the two ontologies? iii) What are the differences of the two ontologies? iv) And why the collaborative ontology of MISE 2.0 is different with the one of MISE 1.0.

For the first two questions, a comparison of the two collaborative ontologies is made in Figure IV-26. They have some concepts in common. These concepts can be considered as the key elements of the collaborative ontology. The concepts would be: the Collaborative Network in CO1 or the Network in CO2, the Participant in CO1 or the Partner in CO2, the Abstract service in CO1 or the Main Function in CO2, the common goal in CO1 or the Objective in CO2, the Resource in CO1 or the Business Message in CO2, and the MIS service in

CO1 or the Generated Mediator Function in CO2. In summarize, one collaborative ontology may contain the concepts about: collaborative network, partner, objective, shared function, shared resource and mediation function.

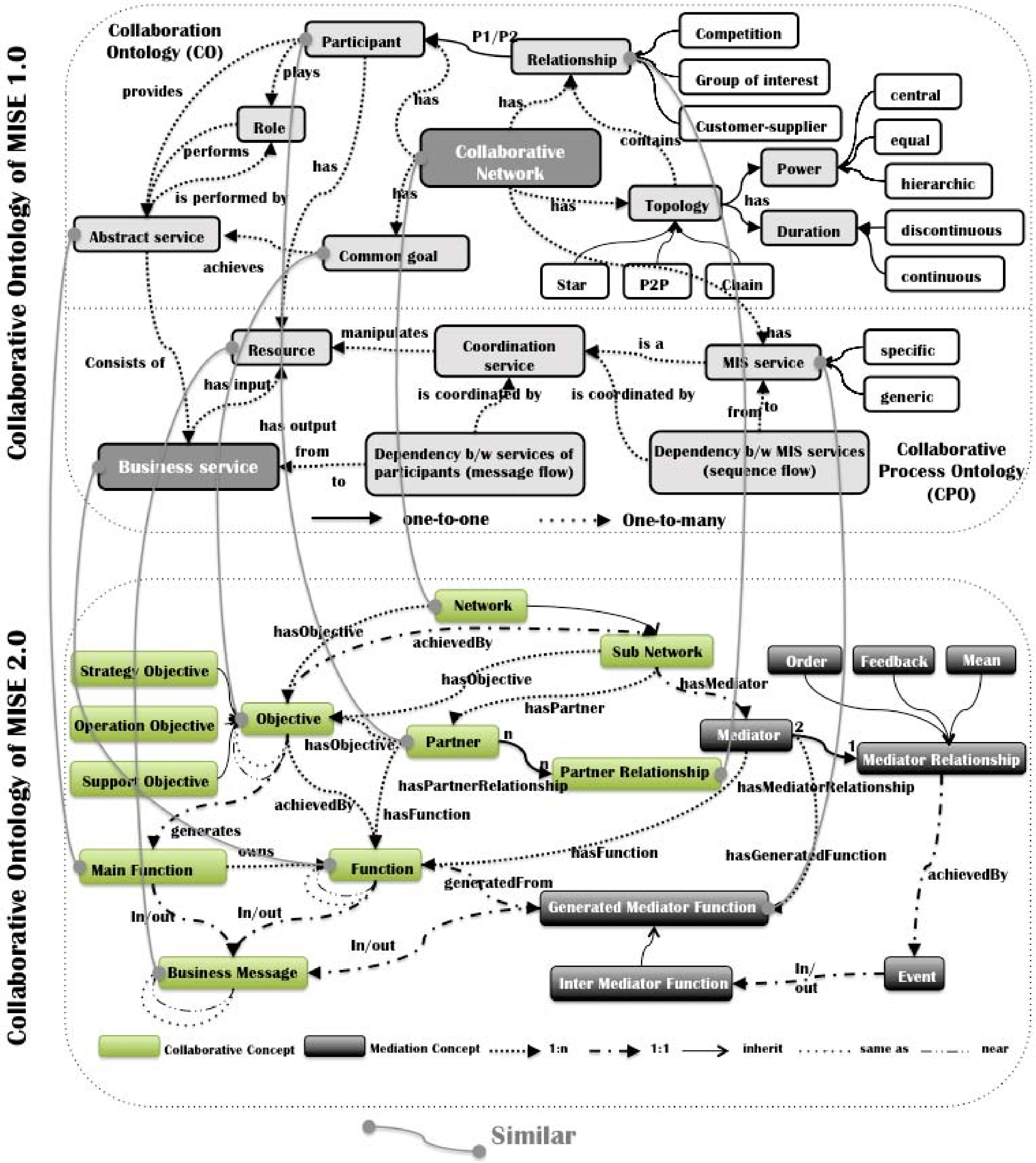


FIGURE IV - 26 COMPARISONS OF COLLABORATIVE ONTOLOGIES OF MISE 1.0 AND MISE 2.0

For the question three, there are two main differences for the CO1 and CO2. First, the CO1 defines concepts, which are the properties of the collaborative network, for example, the Topology of CO1. It precisely defines the performance and the properties of the collaborative network. CO2 focuses to the network and the sub network and to organization the collaborative network model by decomposition of main network. For CO2, the Topology is not really useful and practical. So the concepts concerned Topology are not considered in CO2.

Second, the objective of the two ontologies is different. The CO1 is to deduce a collaborative process. The CO2 is to deduce the collaborative process cartography. The difference leads to two things. For CO2, to build the collaborative process cartography the classification of objective is added to the collaborative ontology. The collaborative process cartography contains several collaborative processes; this means that there would be several mediators. So the CO2 considers mediator as a concept. It defines the sub network and the relations among network, sub network and objectives to complete mediators. Furthermore, the collaborative process is BPMN based process. So in the CO1, the sequence flow and the message flow are considered as concepts. But the CO2 is designed to be able to deduce the collaborative processes, which are not only BPMN based but also others (even though in MISE 2.0, the collaborative process is still BPMN based). There are no concepts such as sequence flow and message flow in CO2.

In next chapter, the software tool: Mediator Modeling 2ool, which supports the model definition, model transformation, business service selection and business process deduction, is presented. First, the global design of the software is addressed. Second, the usage of the software tool is explained through an example.

Chapter V: Final Result

I. Introduction 116

II. Mediator modeling 2ool 117

II.1. Development 118

II.2. Implementation 120

III. Example 123

IV. Conclusion 132

I. Introduction

This chapter aims to mainly introduce i) the software techniques, which implement the Mediator modeling 2ool of MISE 2.0 abstract level and ii) the usage and the functions of Mediator modeling 2ool through an example. In addition, the ontology definition tool, which is developed by Tiexin Wang doing his Master internship, is presented as a support tool of Mediator modeling 2ool. The position of this chapter in the UJ picture is represented in the dash-line box of Figure V-1. The main functions of Mediator modeling 2ool are i) define collaborative network model, ii) define function model, iii) link objectives with functions (business services selection), iv) transfer defined model to collaborative process model and v) update collaborative ontology. The ontology definition tool mainly deals with: i) insert or create instances to collaborative ontology in batch mode, ii) basic concepts mapping between two ontologies and iii) based on the mapping of concepts, inserts the instances of one ontology into another.

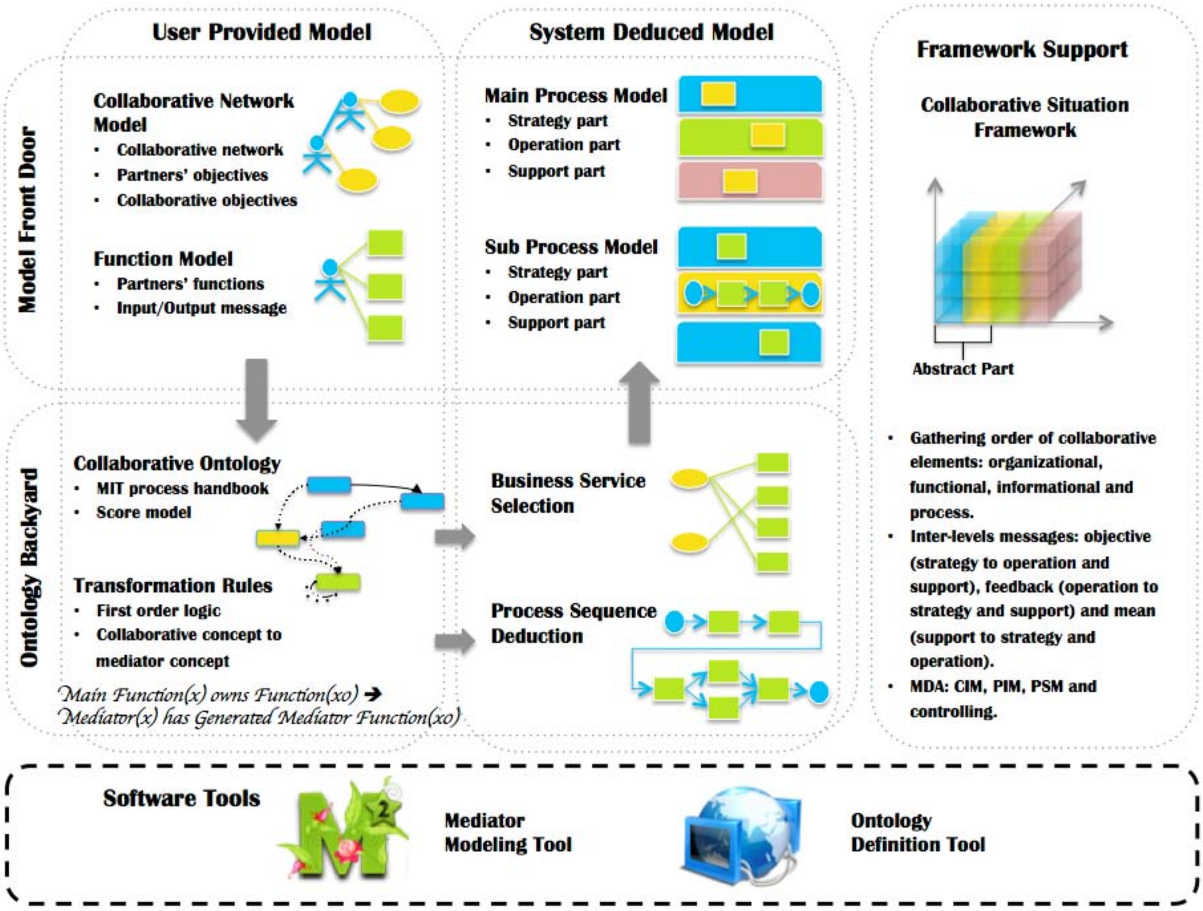

User Provided Model
System Deduced Model
Framework Support
Model Front Door
Collaborative Network Model
• Collaborative network
• Partners' objectives
• Collaborative objectives
Function Model
• Partners' functions
• Input/Output message
Main Process Model
• Strategy part
• Operation part
• Support part
Sub Process Model
• Strategy part
• Operation part
• Support part
Collaborative Situation Framework
Abstract Part
Ontology Backyard
Collaborative Ontology
• MIT process handbook
• Score model
Transformation Rules
• First order logic
• Collaborative concept to mediator concept
Main Function(x) owns Function(xo) → Mediator(x) has Generated Mediator Function(xo)
Business Service Selection
Process Sequence Deduction
• Gathering order of collaborative elements: organizational, functional, informational and process.
• Inter-levels messages: objective (strategy to operation and support), feedback (operation to strategy and support) and mean (support to strategy and operation).
• MDA: CIM, PIM, PSM and controlling.
Software Tools
Mediator Modeling Tool
Ontology Definition Tool

FIGURE V - 1 POSITION OF CHAPTER V IN THE UJ

In MISE 1.0, the CIM level uses GMF[1], ATL[2], XSLT[3] and Protégé[4] to develop the prototype of software tool. At the PIM level ATL completes the transformation from CIM to PIM. At PSM level, Sébastien Truptil develops the supports tool with ATL, Protégé, GMF, JavaScript and Petals ESB. But for MISE 1.0, these

[1] Graphical Modeling Framework
[2] Atlas Transformation Language
[3] Extensible Stylesheet Language Transformation
[4] http://protege.stanford.edu/

software tools cannot be deployed on the ESB as web services. Consequently, the flexibility and automation of the software tools have been fragilized. It indirectly leads to that the agility management cannot be implemented in 100% automatic way.To develop software tools based on web services, is the main goal of MISE 2.0 software design. So the vision of our team is to focus on SaaS (Software as a Service). The software tools of MISE 2.0 are designed and developed as SaaS, which can be loaded by web browser and used by any Internet user.

In this chapter, section II gives the core program design and the interface design of Mediator modeling 2ool. Section IV uses an example to show the modeling and transformation steps with Mediator modeling 2ool. In the conclusion, the detailed function and design plan of the ontology definition tool are provided.

II.Mediator modeling 2ool

The mediator modeling 2ool is designed to implement model definition and model transformation. In the model definition part (Figure V-2), the tool can define i) the collaborative network model (described in section I of chapter III) and ii) the function model (described in section II of chapter III).

To define the collaborative network model and the function model, the following detailed requirements (user friendly requirements) must be satisfied:

- The modeling elements are represented by symbol. The symbols of modeling elements can be selected, dragged, dropped, deleted, resized and connected by arrows.
- The modeling elements have property view (property as name, for example). Specially, for the properties of function and objective must have properties: same as and near by. For the function and message, the property of semantic annotations should be added.
- Both the collaborative network model and the function model can be exported and imported as files in XML format or picture in JPEG format. The XML files can be uploaded on the server side.

For the model transformation (Figure V-2), the tool can support i) the transformation rules defined in section II of chapter IV, ii) the algorithms of business service section in section III of chapter IV and iii) the process deduction method in section III of chapter IV.

To support the above main functions, the following detailed requirements (user friendly requirements) must be taken into account:

- To complete the selection of business service, the instances of objective, function and message in collaborative ontology must by shown or imported into the tool.
- After the accomplishment of business service selection, the mapping results between functions and objectives must be shown as a figure and saved as XML file. The XML file can be uploaded to the server side.
- To show the final results of process deduction, all the deduced XML files of collaborative process can be opened in Petals easy BPM[1]. The XML files can be uploaded to the server side.

To complete the main function, there is one thing. The definition of models and deduction of model can be seen as a project. So the software must create new project on the server side. Meanwhile, the files on the server side must be show as list or file tree on the client side.

[1] Petals BPM: http://research.petalslink.org/display/petalsbpm/Petals+BPM+-+Open+source+BPMN+2.0+modeler

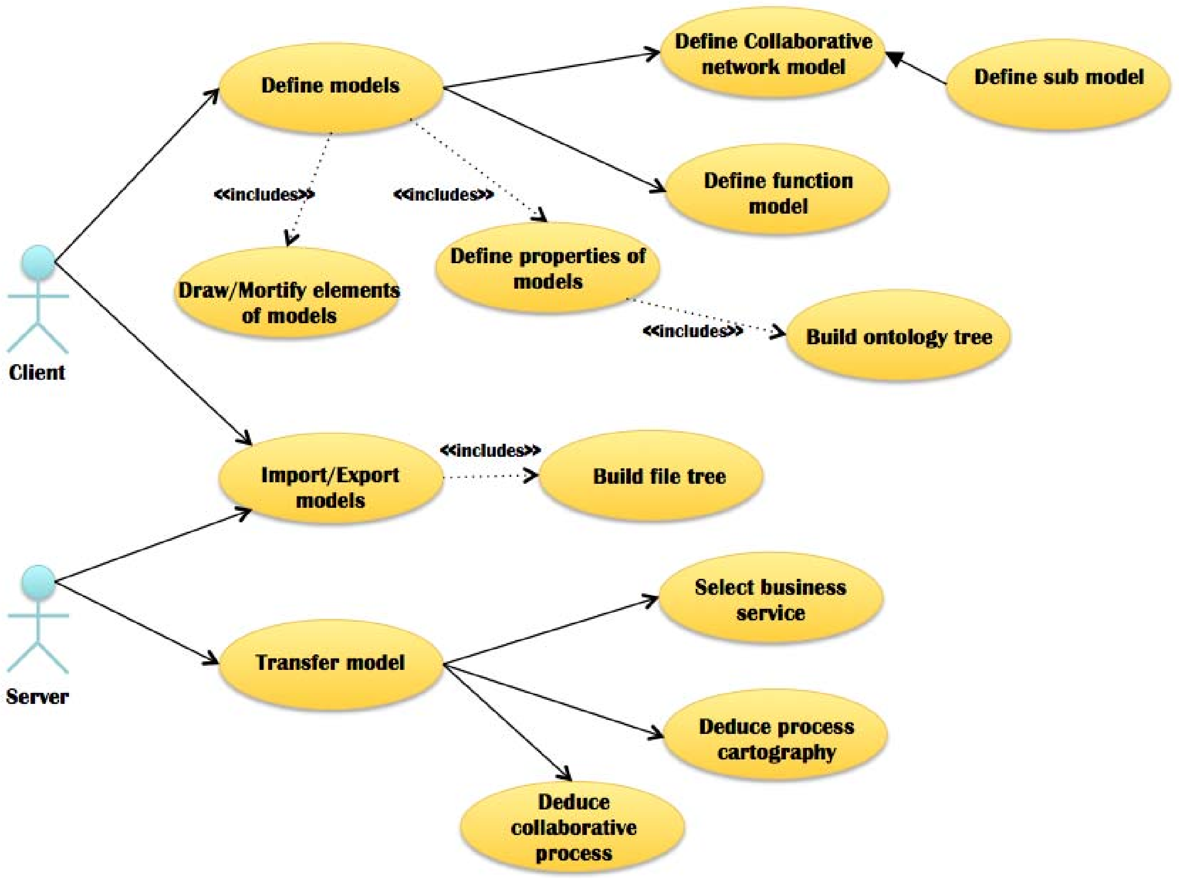


FIGURE V - 2 MAIN USE CASE DIAGRAM OF MEDIATOR MODELING 2OOL

II.1. Development

The Mediator modeling 2ool is designed as two main parts. First, it is “Define models”. As shown in Figure V-3, this part is represented as two sides: client side and server side. It defines the main modules on client side and server side and the communications between client side and server side. The first module is “Create project file” on the client side. The user uses the main interface on client side to create the new project. Then the client side passes the command to the server. The server creates the folder of the project and the initial XML files of the collaborative network model and the function model. Then the user on the client side can define the collaborative network model by using the palette and the canvas on the main interface. Once the user has finished the definition of the model on the client side, the user could export the XML file (.org) of the collaborative network model on the server side. The server side re-writes the XML file (.org) of the model. The user on the client side could also open the XML file (.org) of the model and re-draw the model on the canvas. So the communication between client and server in this phase can be unidirectional or bidirectional. The last phase of “Define model” is to define the functional model on the client side. Then the user can save the model as XML file (.fun) and upload to the server side. As the last phase, the communication between client and server side is a two-way street.

Up to this time, the knowledge-gathering phase is over. The organizational and functional knowledge has been defined. The next step is to transfer the knowledge to the target collaborative process model.

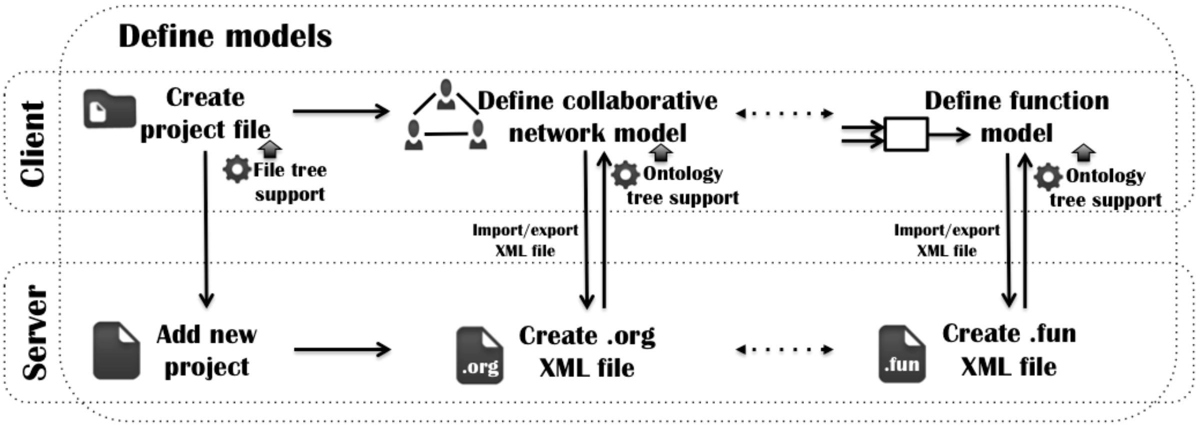


FIGURE V - 3 MAIN FLOWS OF "DEFINE MODELS"

The second part of the design of the Mediator modeling 2ool is "Transfer model". The first thing is to select business services, which means choose the shared functions of partners to achieve the collaborative objectives. The user could pass the command on the client side. The server launches the selection of business services and then writes all the results of selection in a XML file. The user on the client side could download the XML file and then import the file to the main interface as a model. And then the user could ask the server the transfer the gathered models to the collaborative process cartography. The server side launches the program of deduction and creates the XML file in BPMN format (.bpmn). The XML file can be imported, opened and shown by Petals BPM, which is also a GWT based modeling tool. With the collaborative process cartography, the user could ask the server to transfer the collaborative processes. When the server receives the order, the server creates several XML files (.bpmn) to represent the collaborative processes.

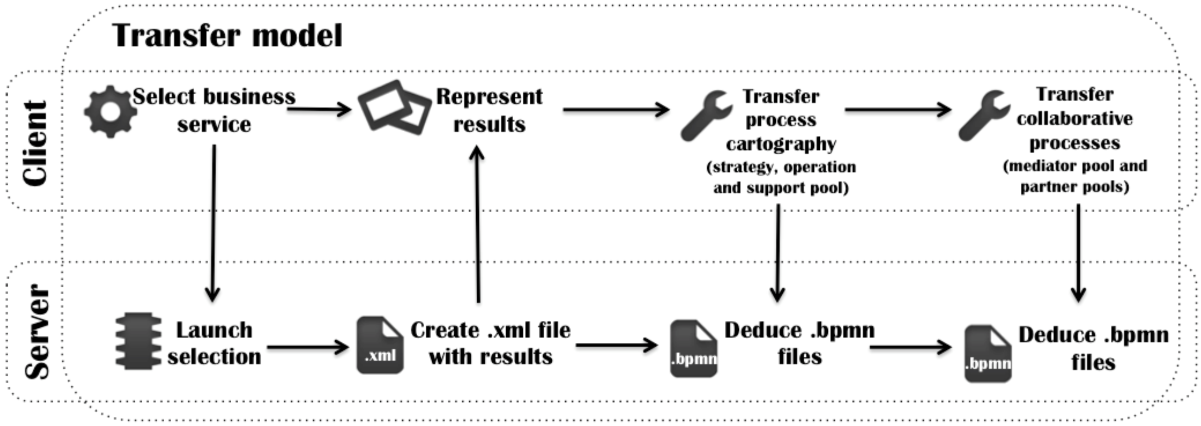

Transfer model
Client
Server
Select business service
Represent results
Transfer process cartography (strategy, operation and support pool)
Transfer collaborative processes (mediator pool and partner pools)
Launch selection
Create .xml file with results
.xml
Deduce .bpmn files
.bpmn
Deduce .bpmn files
.bpmn

FIGURE V - 4 MAIN FLOWS OF "TRANSFER MODEL"

To implement the designed modules and the communications between client and server, the coding of the software tool is designed into 14 packages, which are summarized in the Table V-4. The packages, which start with "../client" are developed for the client side. The package, which start with "../server" is developed for the server side. The packages: "../client/customicons", "../client/ui/panel", "../client/ui/tree" and "../client/submenu" implement the main interfaces. The packages: "../client/elements" and "../client/connector" defines the symbols of the modeling elements. They also define if the modeling elements can be dragged, dropped, connected, selected and resized. The package "../client/editormodel" implements the property views of the modeling elements. The packages "../client/palette" and "../client/view" defines the canvas of the models. The package "../client/sevice" includes the interface classes of server. They help the client side to invoke the server. The classes in the package "../server" implements the interfaces, which are defined in the package "../client/service".

TABLE V - 1 SUMMARIES OF PACKAGES

Package Name	Class No.	Server/Client	Function Description
../client/	2	Client	Main frame and entry point class
../client/customicons	2	Client	Information classes of icons in SVG and PNG format
../client/ui/panel	8	Client	The creations of sub windows of main frame
../client/ui/tree	9	Client	File tree and ontology tree
../client/elements	22	Client	Configuration (symbol, linked property view, belonged canvas and so on) of modeling elements and syntaxes of modeling elements
../client/connector	13	Client	Connector objective, e.g. partner relationship, objective relationship and in/out message flow
../client/editormodel	11	Client	The properties views for modeling elements and connectors
../client/template	1	Client	The template of property menu
../client/submenu	4	Client	Mouse over popped up menu for creation of sub collaborative network
../client/fileoperation	7	Client	Import and Export all XML files and creation of project
../client/palette	14	Client	Definition of palette and drag proxies of modeling elements
../client/view	2	Client	The factory of modeling elements (factory pattern of object-oriented design) and the canvas of the modeling elements
../client/service	2	Client	Server RPC interface
../server	5	Server	Import and export of XML files, the deduction of collaborative process and collaborative process cartography, and business service selection

In this section, the design of the Mediator modeling 2ool is briefly introduced. Because the subject of this thesis is more like model transformation and knowledge based information system in the enterprise interoperability domain and the thesis is not for software engineering, the detailed software design is not explained in the thesis. The aim of this section is to present the main modules of the software tool and the global design of the software tool. But the section I of Annex B presents the technologies, which are used in Mediator modeling 2ool. In the section II of Annex B, there are the detailed designs of classes for each package. In the section III of Annex B, the design of XML files (.org, .fun and .bpmn) are addresses. The section IV of Annex B shows some main codes (Java codes) of the software tool.

II.2. Implementation

In this section, the main implementation results (the main interface of the Mediator modeling 2ool) are presented. The main interface is design into five windows (Figure V-5 and Figure V-6). The first one is the one on the top. It shows the logo of the software tool, the menu bar and the tool bar. The second window is in the center. It is the canvas of the model. The user could define models in this window. The third window is at the bottom. It is an information window. When there is a warning, an error or an announcement, the messages are shown in this window.

On the left hand side, it is the forth window, which is named as "Explorer". It contains to sub windows: "Palette" (on the left hand of Figure V-5) and "File" (on the left hand of Figure V-6). The "Palette" provides the modeling symbols of the modeling elements. The user could drag the modeling symbols to the canvas to define the models. The “Fill” is an explorer of the files on the server side. The files are presented as file tree.

On the right hand side, it is the fifth window. It is named “Add more knowledge”. This window contains two sub windows: “Properties” (on the right hand of Figure V-5) and “Ontology” (on the right hand of Figure V-6). The “Properties” shows the detailed information of modeling elements. The user could add, change or delete the properties of the modeling elements. The “Ontology” loads the collaborative ontology from .owl file, which comes from Protégé (Figure V-7).

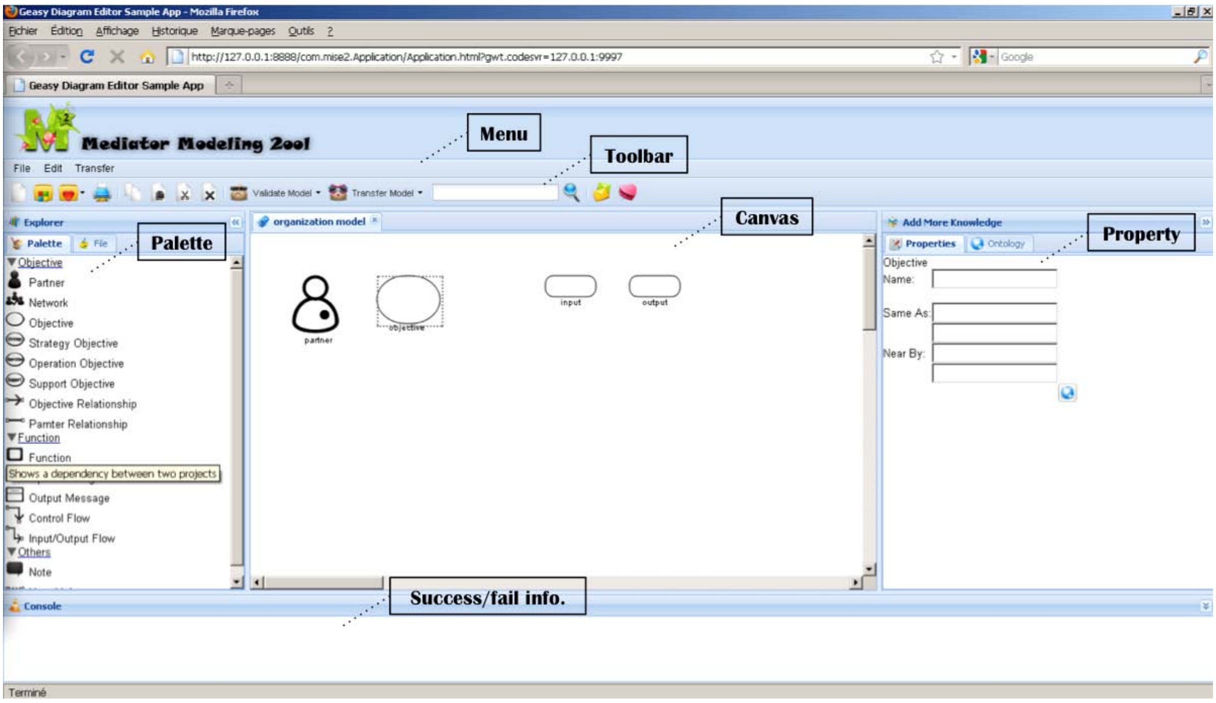
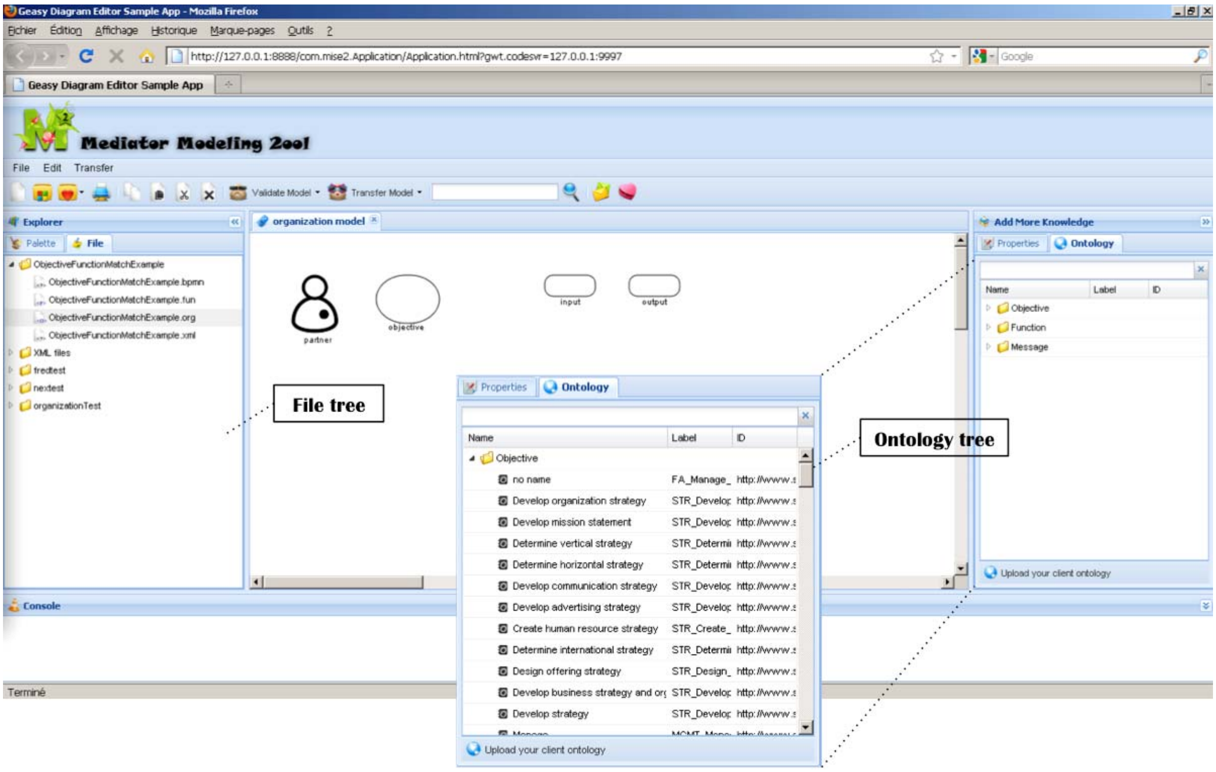


FIGURE V - 5 MAIN INTERFACE (PALETTE AND PROPERTY)

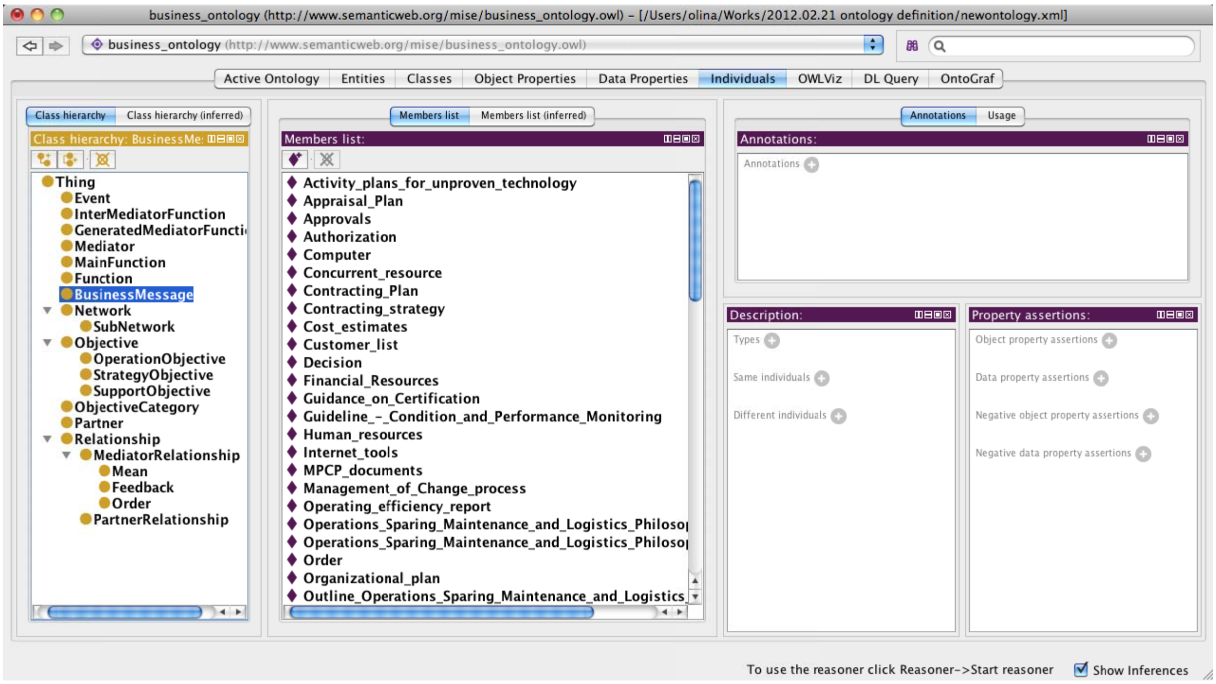


Name	Label	ID
Objective		
no name	FA_Manage_	http://www.s
Develop organization strategy	STR_Develop	http://www.s
Develop mission statement	STR_Develop	http://www.s
Determine vertical strategy	STR_Determi	http://www.s
Determine horizontal strategy	STR_Determi	http://www.s
Develop communication strategy	STR_Develop	http://www.s
Develop advertising strategy	STR_Develop	http://www.s
Create human resource strategy	STR_Create_	http://www.s
Determine international strategy	STR_Determi	http://www.s
Design offering strategy	STR_Design_	http://www.s
Develop business strategy and or	STR_Develop	http://www.s
Develop strategy	STR_Develop	http://www.s

FIGURE V - 6 MAIN INTERFACE (FILE AND ONTOLOGY)

This collaborative ontology has been defined thanks to Protégé. A program has been written to extract the instances of MIT process handbook and insert into the collaborative ontology. As shown in Figure V-7, the first window on the left hand side is the definitions of concepts. The second window on the left hand side is the definitions of individuals (instances). The "BusinessMessage" is selected in the first window, so the individuals in the second window are the ones of the "BusinessMessage".

But in the ontology tree of Mediator modeling 2ool, there are only "Function", "Objective" and "BusinessMessage", because the objectives of the collaborative network model, functions and input/output messages of function model has the property same as/near by, which has to been selected from the collaborative ontology.


business_ontology (http://www.semanticweb.org/mise/business_ontology.owl) – [/Users/olina/Works/2012.02.21 ontology definition/newontology.xml]

business_ontology (http://www.semanticweb.org/mise/business_ontology.owl)

To use the reasoner click Reasoner->Start reasoner

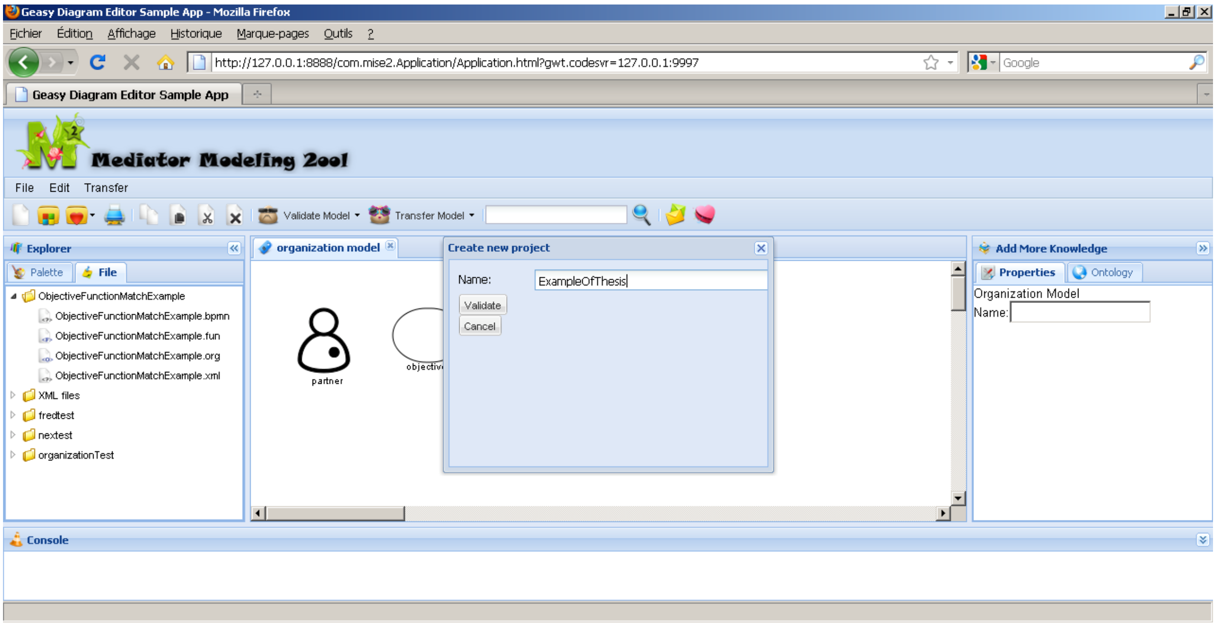
FIGURE V - 7 COLLABORATIVE ONTOLOGY IN PROTÉGÉ

III. Example

In this section, an example is developed to present the usage of the Mediator modeling 2ool. The example follows the design flows, which has introduced in previous section.

Step 1: Create new project

As shown in Figure V-8, the user clicks on the first button ( ) of the toolbar (from the left hand side). The window of "create new project" pops up. The user could type in the name (ExampleOfThesis) of the new project and then validate it.


Geasy Diagram Editor Sample App - Mozilla Firefox
Fichier Edition Affichage Historique Marque-pages Outils ?
http://127.0.0.1:8888/com.mise2.Application/Application.html?gwt.codesvr=127.0.0.1:9997
Geasy Diagram Editor Sample App
Mediator Modeling Tool
File Edit Transfer
Validate Model
Transfer Model
Explorer
Palette
File
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmn
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest
nextest
organizationTest
organization model
partner
objective
Create new project
Validate
Cancel
Add More Knowledge
Properties
Ontology
Organization Model
Console

Name: ExampleOfThesis

Name:

FIGURE V - 8 CREATE NEW PROJECT

The folder “ExampleOfThesis” is added into the file system on the server and presented in the file tree as a node (Figure V-9). In this folder, there are two files, which are created automatically. The file “ExampleOfThesis.org” is an empty fill. The user could open this fill in the canvas and define the collaborative network model in the file. The file “ExampleOfThesis.fun” is also an empty fill. The user could open the file in the canvas and define the function model in the file.

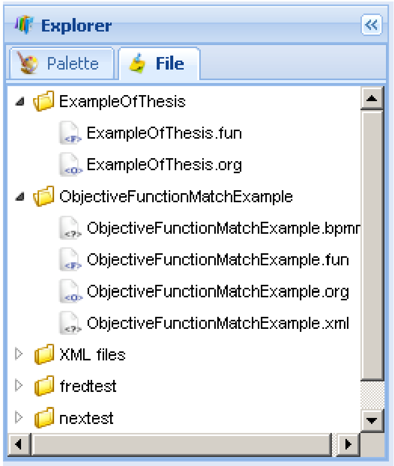
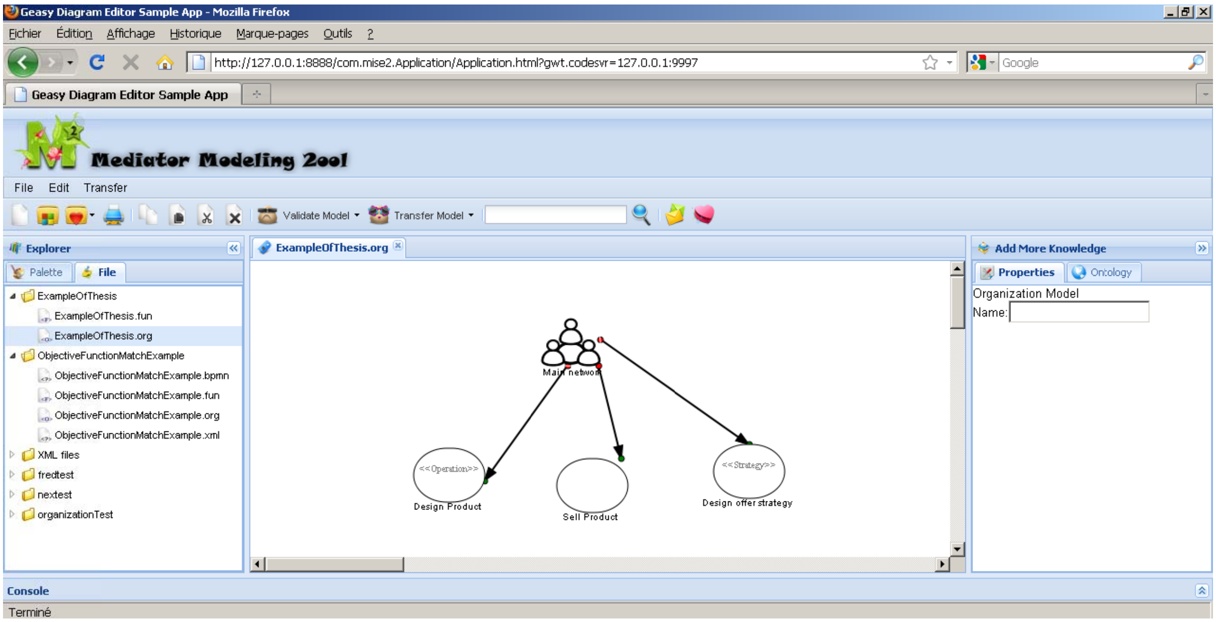

Explorer
Palette
File
ExampleOfThesis
ExampleOfThesis.fun
ExampleOfThesis.org
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmr
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest
nextest

FIGURE V - 9 ADDED NEW FOLDER AND FILES

Step 2: Define the collaborative network model

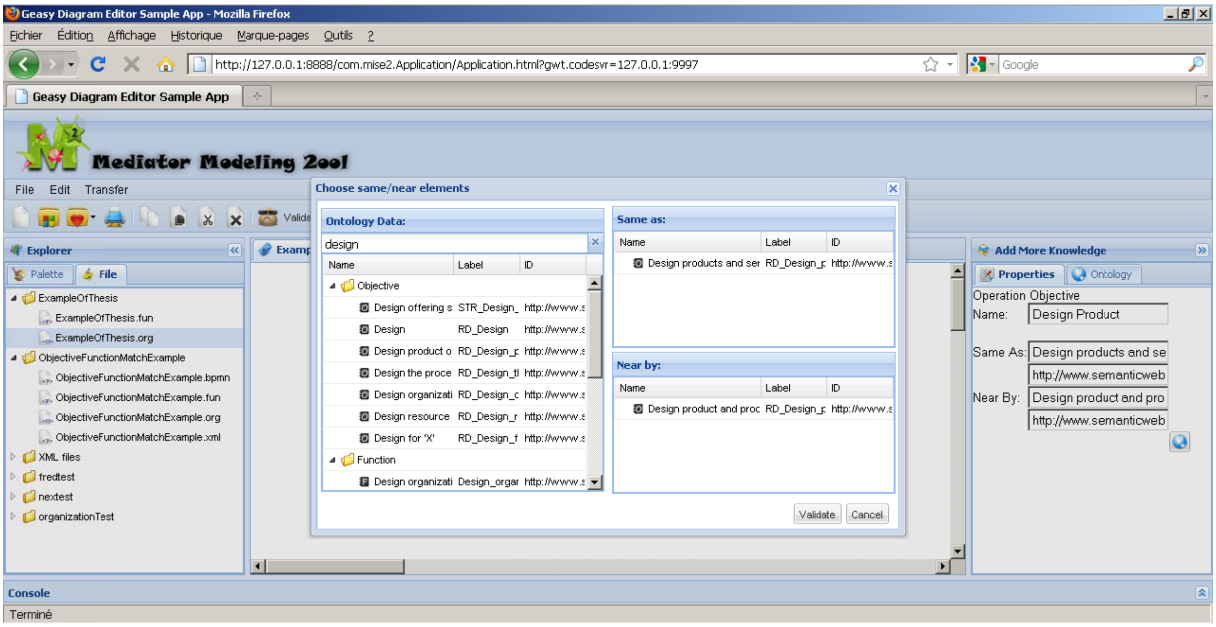
The main collaborative network model is represented in Figure V-10. The main collaborative network has three collaborative objectives: operation objective “design product”, strategy objective “design offer strategy” and general objective “sell product”.


Geasy Diagram Editor Sample App - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils ?
http://127.0.0.1:8888/com.mise2.Application/Application.html?gwt.codesvr=127.0.0.1:9997
Google
Geasy Diagram Editor Sample App
Mediator Modeling Tool
File Edit Transfer
Validate Model
Transfer Model
Explorer
Palette
File
ExampleOfThesis
ExampleOfThesis.fun
ExampleOfThesis.org
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmn
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest
nextest
organizationTest
ExampleOfThesis.org
Main network
<<Operation>>
Design Product
Sell Product
<<Strategy>>
Design offer strategy
Add More Knowledge
Properties
Ontology
Console
Terminé

Organization Model
Name:

FIGURE V - 10 LEVEL 0 COLLABORATIVE NETWORK MODEL

For each objective, in the property window, the same as/near by instances have to be selected. As shown in Figure V-11, the user could click on the small "world" button (🌐) in the property window. The window "Choose same/near elements" is opened. The user could drag the same as/near by instances from ontology window to the other two windows and then validate.


Geasy Diagram Editor Sample App - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils ?
http://127.0.0.1:8888/com.mise2.Application/Application.html?gwt.codesvr=127.0.0.1:9997
Google
Geasy Diagram Editor Sample App
Mediator Modeling Tool
File Edit Transfer
Valida
Explorer
Palette
File
ExampleOfThesis
ExampleOfThesis.fun
ExampleOfThesis.org
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmn
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest
nextest
organizationTest
Examp
Choose same/near elements
Validate
Cancel
Add More Knowledge
Properties
Ontology
Console
Terminé

Ontology Data:

design

Name	Label	ID
Objective		
Design offering s	STR_Design_	http://www.s
Design	RD_Design	http://www.s
Design product o	RD_Design_p	http://www.s
Design the proce	RD_Design_tl	http://www.s
Design organizati	RD_Design_c	http://www.s
Design resource	RD_Design_r	http://www.s
Design for 'X'	RD_Design_f	http://www.s
Function		
Design organizati	Design_orgar	http://www.s

Same as:

Name	Label	ID
Design products and ser	RD_Design_p	http://www.s

Near by:

Name	Label	ID
Design product and proc	RD_Design_p	http://www.s

Operation Objective
Name: Design Product
Same As: Design products and se
http://www.semanticweb
Near By: Design product and pro
http://www.semanticweb

FIGURE V - 11 DEFINE SAME AS/NEAR BY INSTANCE FOR "DESIGN PRODUCT"

To define the sub collaborative network for each objective, the user could pass over the symbol of objective. A small menu with two buttons pops up (Figure V-12). The user clicks on the first button on the top (). The sub collaborative model is created and opened in a new canvas. The user could define the sub model in the new window.

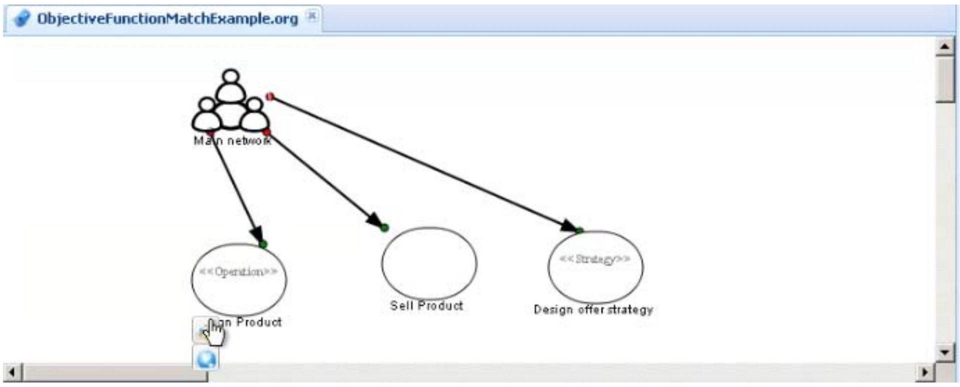


FIGURE V - 12 DEFINE SUB MODEL

The sub model of “design product” is represented in Figure V-13. In this sub network, there are two partners: co-designer and manufacturer. The co-designer has one operation objective: “Design plan and prototype”. The manufacturer has one operation objective: “Evaluate design”.

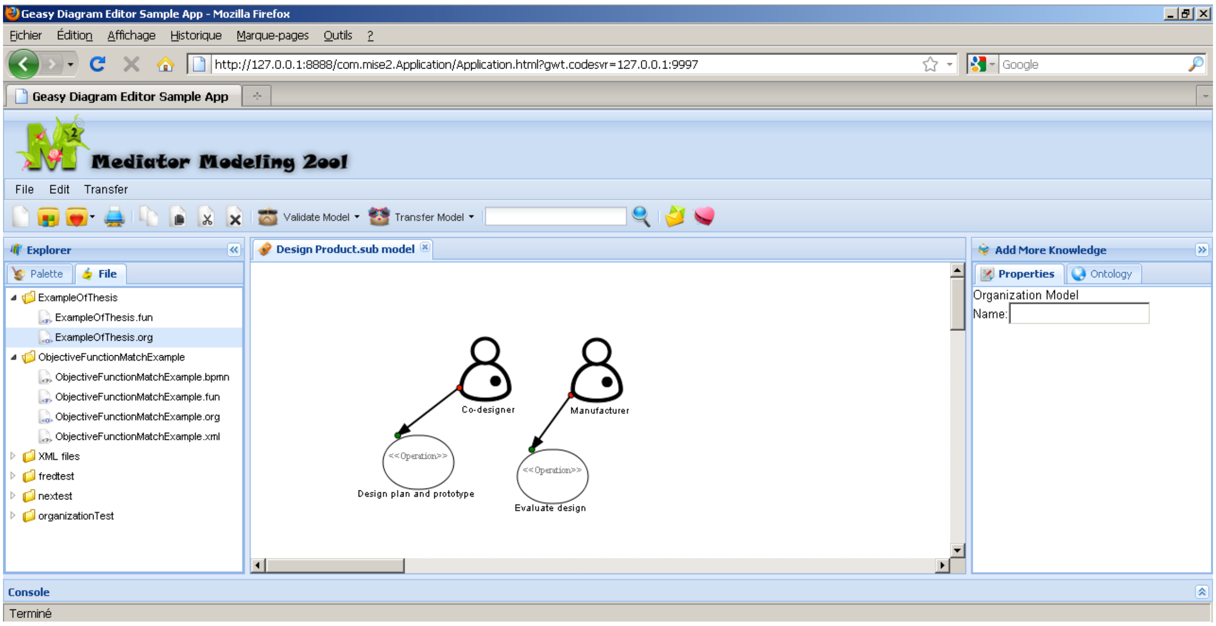
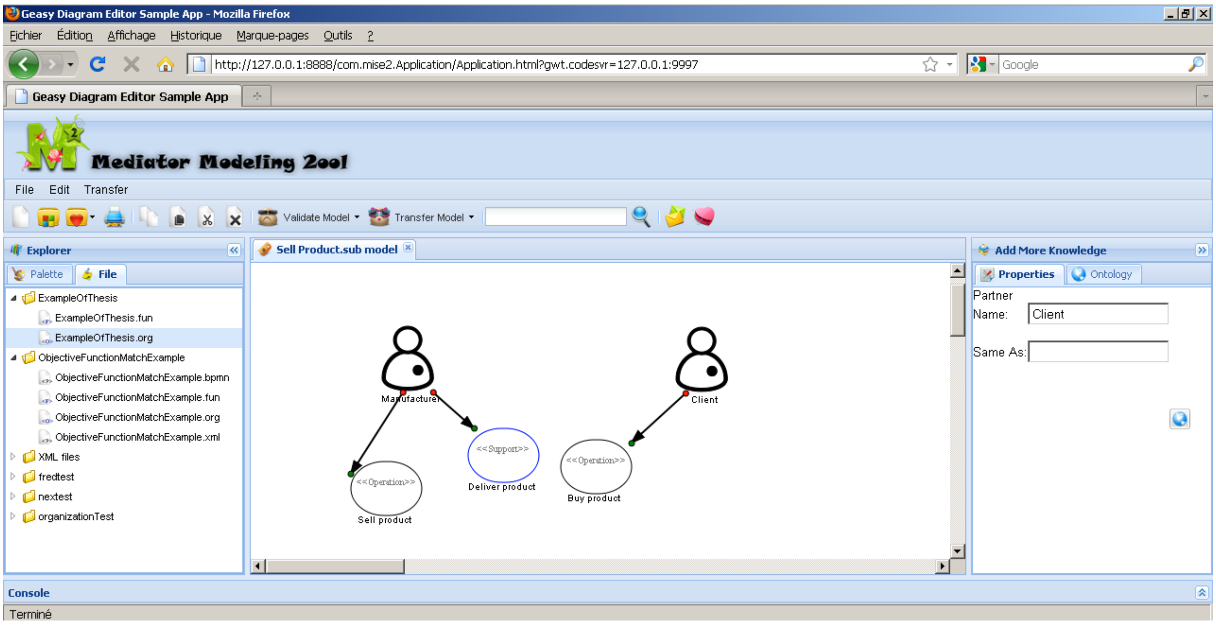

Geasy Diagram Editor Sample App - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils ?
http://127.0.0.1:8888/com.mise2.Application/Application.html?gwt.codesvr=127.0.0.1:9997
Google
Geasy Diagram Editor Sample App
Mediator Modeling Tool
File Edit Transfer
Validate Model
Transfer Model
Explorer
Palette
File
ExampleOfThesis
ExampleOfThesis.fun
ExampleOfThesis.org
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmn
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest
nextest
organizationTest
Design Product.sub model
Co-designer
Manufacturer
<<Operation>>
Design plan and prototype
<<Operation>>
Evaluate design
Add More Knowledge
Properties
Ontology
Organization Model
Name:
Console
Terminé

FIGURE V - 13 SUB NETWORK OF “DESIGN PRODUCT”

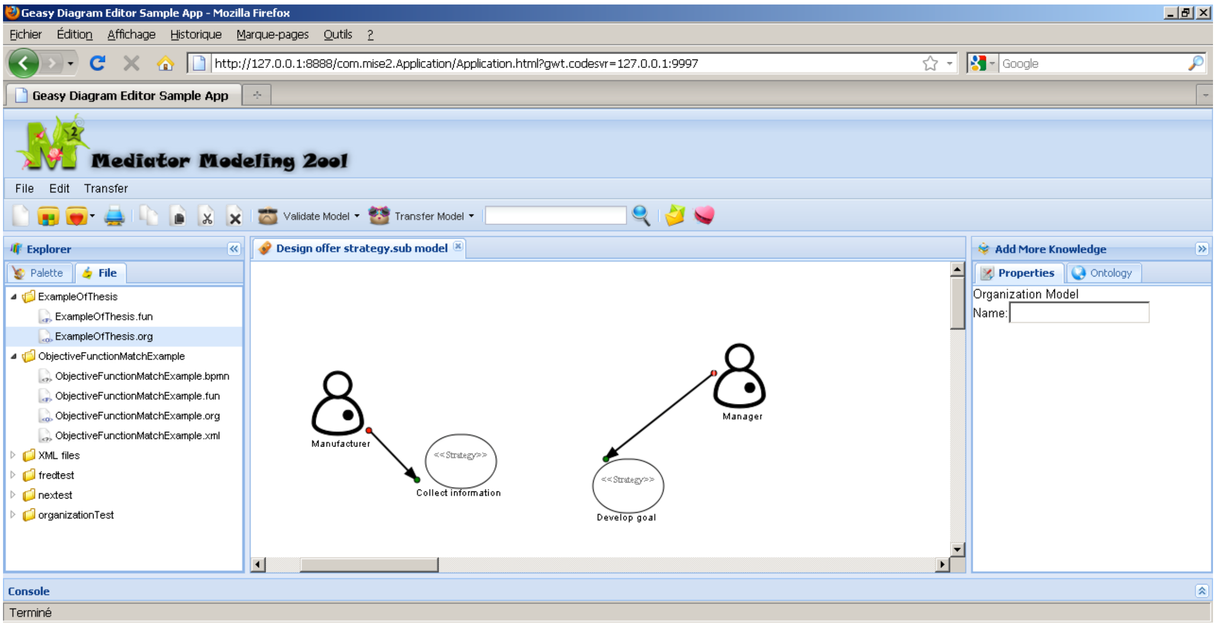
The sub model of “sell product” is shown in Figure V-14. Two partners make up the sub model: manufacturer and client. The manufacturer has two objectives: operation objective “Sell product” and support objective “Deliver product”. The client has one operation objective: “Buy product”.


Geasy Diagram Editor Sample App - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils ?
http://127.0.0.1:8888/com.mise2.Application/Application.html?gwt.codesvr=127.0.0.1:9997
Google
Geasy Diagram Editor Sample App
Mediator Modeling Tool
File Edit Transfer
Validate Model Transfer Model
Explorer
Palette File
ExampleOfThesis
ExampleOfThesis.fun
ExampleOfThesis.org
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmn
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest
nextest
organizationTest
Sell Product.sub model
Manufacturer
Client
<<Support>>
Deliver product
<<Operation>>
Buy product
<<Operation>>
Sell product
Add More Knowledge
Properties Ontology
Console
Terminé

Partner
Name: Client
Same As:

FIGURE V - 14 SUB NETWORK OF "SELL PRODUCT"

The sub model of "Design offer strategy" is shown in the Figure V-15. The sub model has two partners: manufacturer and manager. The manufacturer has one strategy objective "Collect information". The manager has one strategy objective "Develop goal".

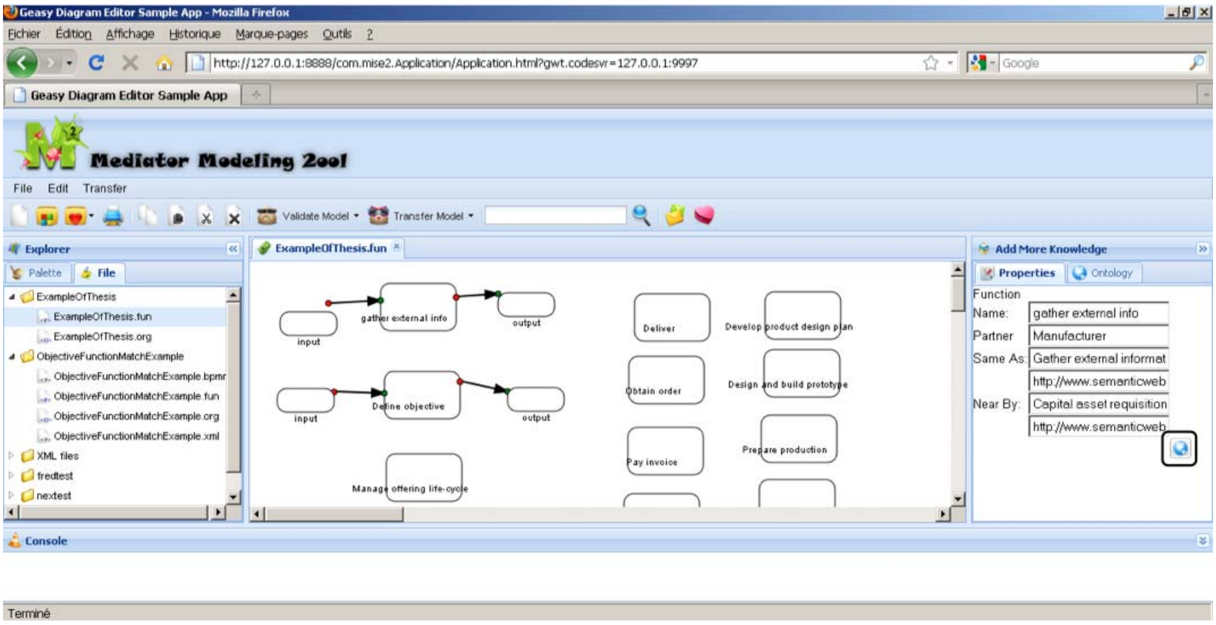

Geasy Diagram Editor Sample App - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils ?
http://127.0.0.1:8888/com.mise2.Application/Application.html?gwt.codesvr=127.0.0.1:9997
Google
Geasy Diagram Editor Sample App
Mediator Modeling Tool
File Edit Transfer
Validate Model Transfer Model
Explorer
Palette File
ExampleOfThesis
ExampleOfThesis.fun
ExampleOfThesis.org
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmn
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest
nextest
organizationTest
Design offer strategy.sub model
Manufacturer
Manager
<<Strategy>>
Collect information
<<Strategy>>
Develop goal
Add More Knowledge
Properties Ontology
Console
Terminé

Organization Model
Name:

FIGURE V - 15 SUB NETWORK OF "DESIGN OFFER STRATEGY"

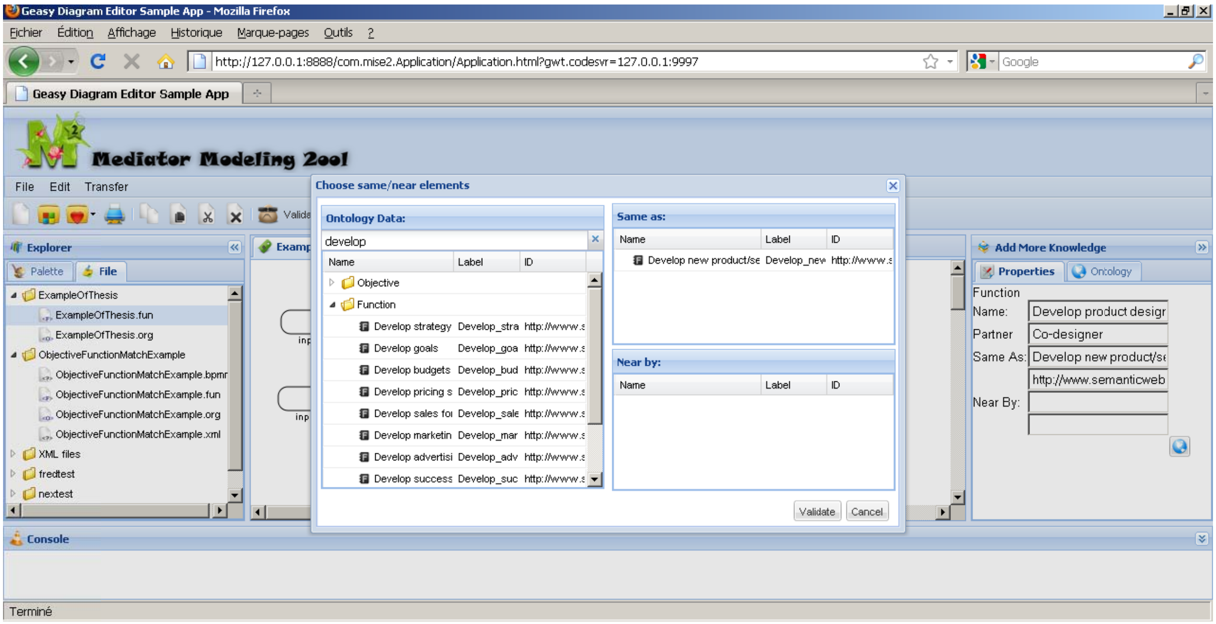
Step 3: Define the function model

As shown in Figure V-16, the user could open the ExampleOfThesis.fun to define the function model. All the functions of partners are listed in the model. For each function, the user needs to define the property: name, partner and same as/near by. To define the same as/near by, the user could click on the button in the property window. The window of "Choose same/near elements" pops up (Figure V-17). The user could drag the instances of ontology to the same as/near by windows.


Geasy Diagram Editor Sample App - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils ?
http://127.0.0.1:8888/com.mise2.Application/Application.html?gwt.codesvr=127.0.0.1:9997
Google
Geasy Diagram Editor Sample App
Mediator Modeling Tool
File Edit Transfer
Validate Model
Transfer Model
Explorer
Palette
File
ExampleOfThesis
ExampleOfThesis.fun
ExampleOfThesis.org
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmr
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest
nextest
ExampleOfThesis.fun
input
gather external info
output
input
Define objective
output
Manage offering life-cycle
Deliver
Develop product design plan
Obtain order
Design and build prototype
Pay invoice
Prepare production
Add More Knowledge
Properties
Ontology
Console
Terminé

Function
Name: gather external info
Partner: Manufacturer
Same As: Gather external informat
http://www.semanticweb
Near By: Capital asset requisition
http://www.semanticweb

FIGURE V - 16 DEFINE FUNCTION MODEL


Geasy Diagram Editor Sample App - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils ?
http://127.0.0.1:8888/com.mise2.Application/Application.html?gwt.codesvr=127.0.0.1:9997
Google
Geasy Diagram Editor Sample App
Mediator Modeling Tool
File Edit Transfer
Explorer
Palette
File
ExampleOfThesis
ExampleOfThesis.fun
ExampleOfThesis.org
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmr
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest
nextest
Add More Knowledge
Properties
Ontology
Validate
Cancel
Console
Terminé

Choose same/near elements

Ontology Data:

develop

Name	Label	ID
Objective		
Function		
Develop strategy	Develop_stra	http://www.s
Develop goals	Develop_goa	http://www.s
Develop budgets	Develop_bud	http://www.s
Develop pricing s	Develop_pric	http://www.s
Develop sales foi	Develop_sale	http://www.s
Develop marketin	Develop_mar	http://www.s
Develop advertisi	Develop_adv	http://www.s
Develop success	Develop_suc	http://www.s

Same as:

Name	Label	ID
Develop new product/se	Develop_new	http://www.s

Near by:

Name	Label	ID

Function
Name: Develop product desigr
Partner: Co-designer
Same As: Develop new product/s
http://www.semanticweb
Near By:

FIGURE V - 17 DEFINE SAME AS/NEAR BY INSTANCES FOR FUNCTION

Step 4: Select business services

On the toolbar, there is a group of button, which is "Transfer model" (Figure V-18). The user could select the first one "match objectives and functions" (🍲). The server side starts to find functions, which can achieve the objectives. The results are written in the "ExampleOfThesis.xml" file. As shown in Figure V-19, in the "ExampleOfThesis" folder, there is a new file: "ExampleOfThesis.xml".

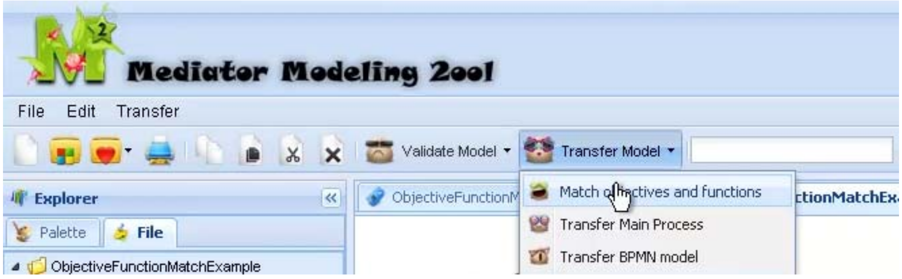


FIGURE V - 18 "TRANSFER MODEL" MENU

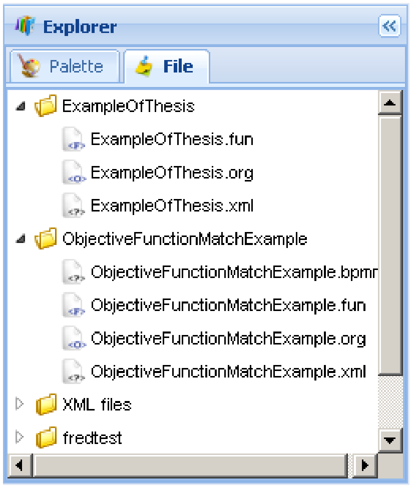
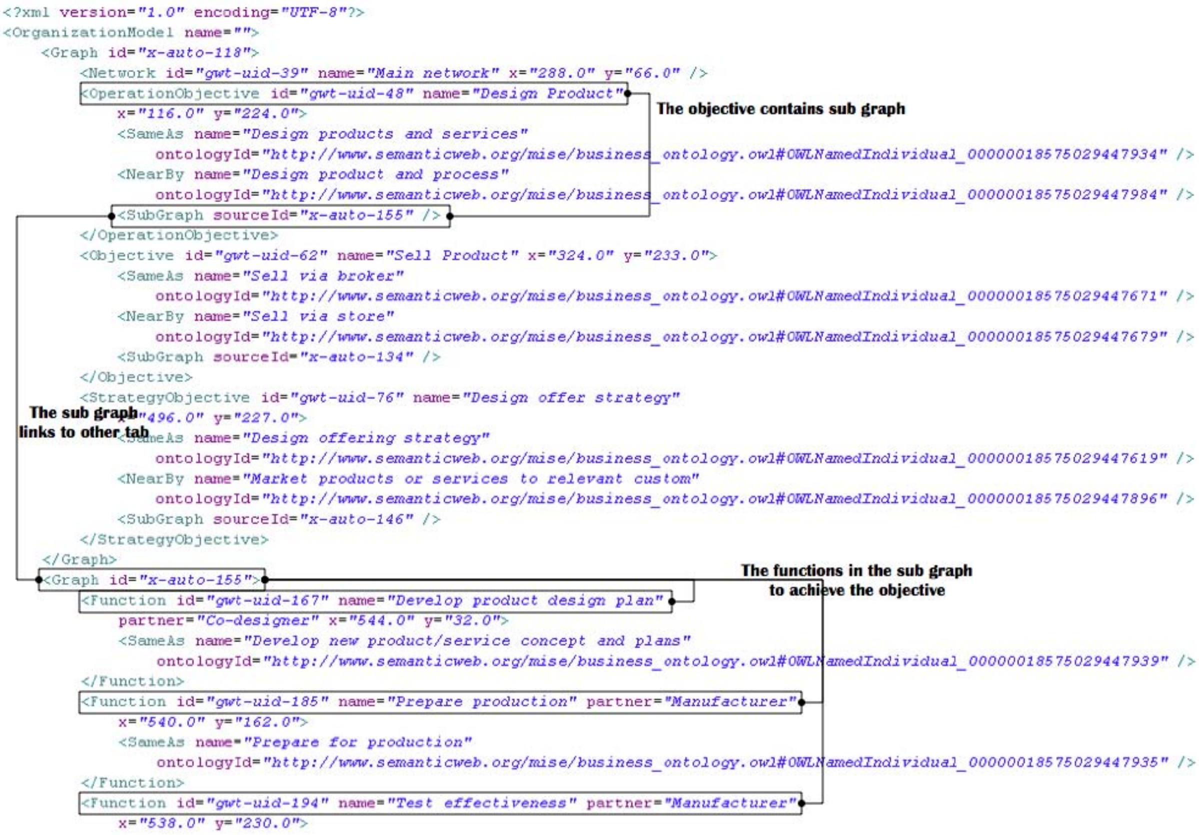

Explorer
Palette
File
ExampleOfThesis
ExampleOfThesis.fun
ExampleOfThesis.org
ExampleOfThesis.xml
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmr
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files
fredtest

FIGURE V - 19 .XML FILE

The user could open the .xml file (Figure V-20). There is the "OperationObjective" tab, which contains "SubGraph". The "SubGraph" is linked to another "Graph" tab, which contains the partner functions. These functions are automatically selected to achieve the operation objective. Further more, the .xml file can be opened in the canvas of Mediator modeling 2ool. The user can check the result in a graphic view.



```
<?xml version="1.0" encoding="UTF-8"?>
<OrganizationModel name="">
    <Graph id="x-auto-118">
        <Network id="gwt-uid-39" name="Main network" x="288.0" y="66.0" />
        <OperationObjective id="gwt-uid-48" name="Design Product"
            x="116.0" y="224.0">
            <SameAs name="Design products and services"
                ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_000001857502944 7934" />
            <NearBy name="Design product and process"
                ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447984" />
            <SubGraph sourceId="x-auto-155" />
        </OperationObjective>
        <Objective id="gwt-uid-62" name="Sell Product" x="324.0" y="233.0">
            <SameAs name="Sell via broker"
                ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447671" />
            <NearBy name="Sell via store"
                ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447679" />
            <SubGraph sourceId="x-auto-134" />
        </Objective>
        <StrategyObjective id="gwt-uid-76" name="Design offer strategy"
            x="496.0" y="227.0">
            <SameAs name="Design offering strategy"
                ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447619" />
            <NearBy name="Market products or services to relevant custom"
                ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447896" />
            <SubGraph sourceId="x-auto-146" />
        </StrategyObjective>
    </Graph>
    <Graph id="x-auto-155">
        <Function id="gwt-uid-167" name="Develop product design plan"
            partner="Co-designer" x="544.0" y="32.0">
            <SameAs name="Develop new product/service concept and plans"
                ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447939" />
        </Function>
        <Function id="gwt-uid-185" name="Prepare production" partner="Manufacturer"
            x="540.0" y="162.0">
            <SameAs name="Prepare for production"
                ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447935" />
        </Function>
        <Function id="gwt-uid-194" name="Test effectiveness" partner="Manufacturer"
            x="538.0" y="230.0">
```

The objective contains sub graph

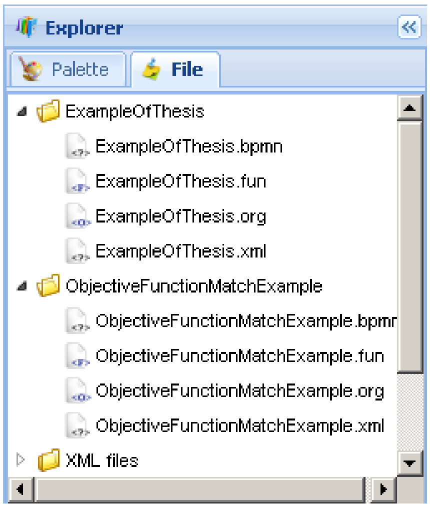
The sub graph links to other tab

The functions in the sub graph to achieve the objective

FIGURE V - 20 XML FILE OF THE SELECTION RESULT

Step 5: Deduce collaborative process cartography

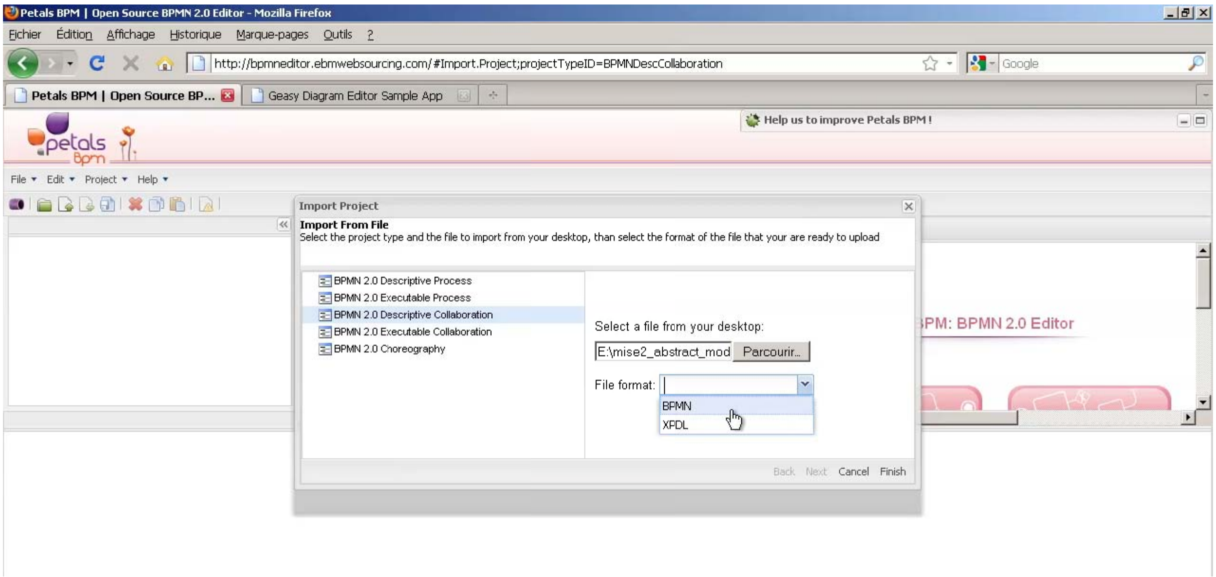
On the toolbar, there is a group of button, which is "Transfer model" (Figure V-18). The user could select the second one "deduce main process" ( ). The server side starts to deduce the collaborative process cartography. The results are written in the "ExampleOfThesis.bpmn" file. As shown in Figure V-21, in the "ExampleOfThesis" folder, there is a new file: "ExampleOfThesis.bpmn".



ExampleOfThesis
ExampleOfThesis.bpmn
ExampleOfThesis.fun
ExampleOfThesis.org
ExampleOfThesis.xml
ObjectiveFunctionMatchExample
ObjectiveFunctionMatchExample.bpmr
ObjectiveFunctionMatchExample.fun
ObjectiveFunctionMatchExample.org
ObjectiveFunctionMatchExample.xml
XML files

FIGURE V - 21 .BPMN FILE OF COLLABROATIVE PROCESS CARTOGRAPHY

The user could open Petals BPM in the web browser and import the deduced "ExampleOfThesis.bpmn" file as BPMN 2.0 descriptive collaboration in BPMN format (Figure V-22). The model of collaborative process cartography is shown in Figure V-23.



Petals BPM | Open Source BPMN 2.0 Editor - Mozilla Firefox

http://bpmneditor.ebmwebsourcng.com/#Import.Project;projectTypeID=BPMNDescCollaboration

Import Project

Import From File

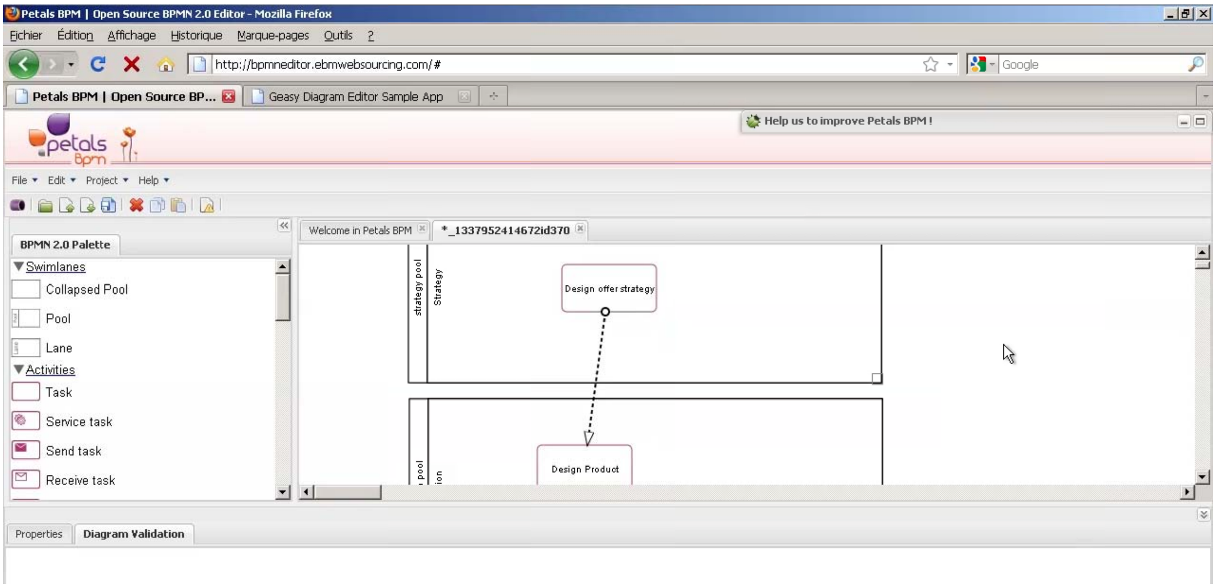
Select the project type and the file to import from your desktop, than select the format of the file that your are ready to upload

- BPMN 2.0 Descriptive Process
- BPMN 2.0 Executable Process
- BPMN 2.0 Descriptive Collaboration
- BPMN 2.0 Executable Collaboration
- BPMN 2.0 Choreography

Select a file from your desktop: E:\mise2_abstract_mod

File format: BPMN / XFDL

FIGURE V - 22 IMPORT .BPMN TO PETALS BPM



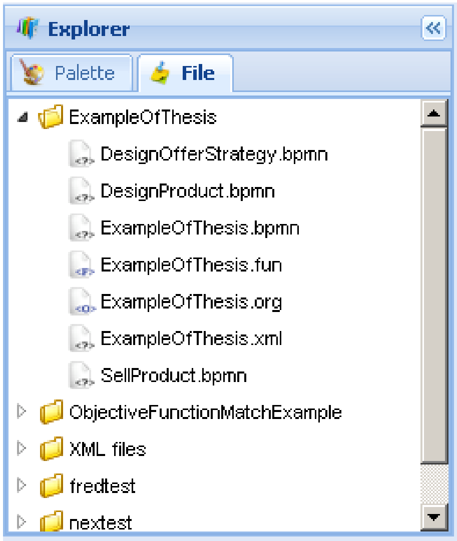
Petals BPM | Open Source BPMN 2.0 Editor - Mozilla Firefox

http://bpmneditor.ebmwebsourcng.com/#

FIGURE V - 23 THE COLLABROATIVE PROCESS CARTOGRAPHY IN PETALS BPM

Step 6: Deduce collaborative processes

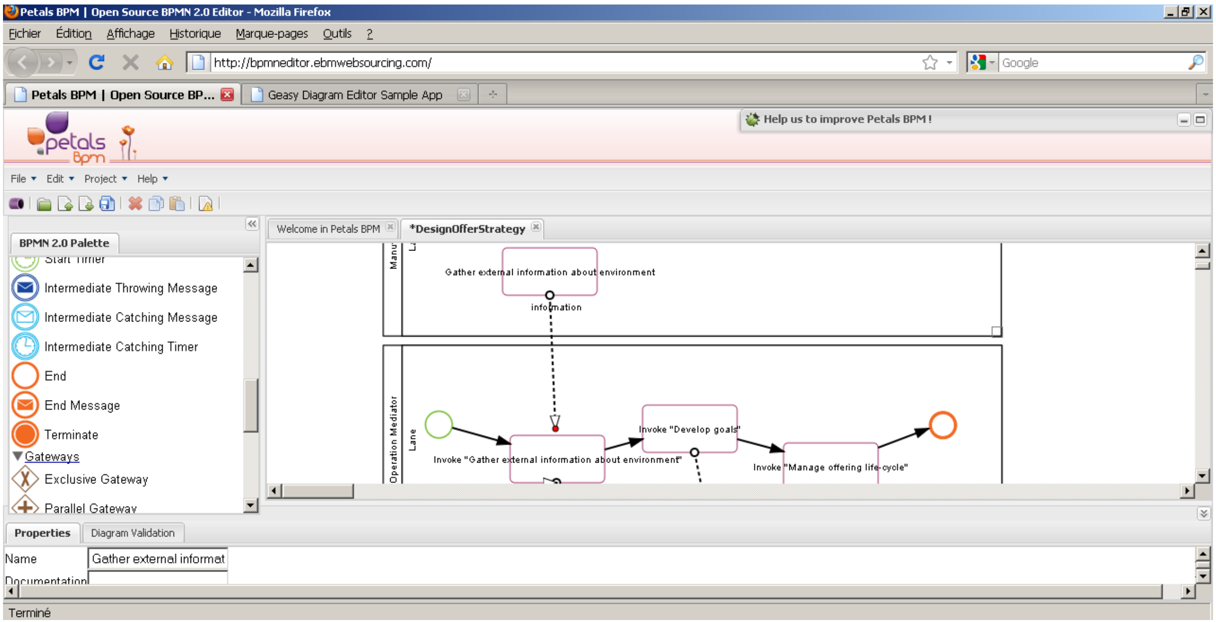
On the toolbar, there is a group of button, which is "Transfer model" (Figure V-18). The user could select the first one "transfer BPMN model" ( ). The server side starts to transfer collaborative processes for the main tasks in the collaborative process cartography. The results are written in the "DesignOfferStrategy.bpmn", "DesignProduct.bpmn" and "SellProduct.bpmn" files. As shown in Figure V-24, in the "ExampleOfThesis" folder, there are new files: "DesignOfferStrategy.bpmn", "DesignProduct.bpmn" and "SellProduct.bpmn".



- ExampleOfThesis
  - DesignOfferStrategy.bpmn
  - DesignProduct.bpmn
  - ExampleOfThesis.bpmn
  - ExampleOfThesis.fun
  - ExampleOfThesis.org
  - ExampleOfThesis.xml
  - SellProduct.bpmn
- ObjectiveFunctionMatchExample
- XML files
- fredtest
- nextest

FIGURE V - 24 .BPMN FILES OF COLLABROATIVE PROCESSES

The user could open Petals BPM in the web browser and import the deduced "DesignOfferStrategy.bpmn" file as BPMN 2.0 descriptive collaboration in BPMN format as in Figure V-22. The model of collaborative process of "DesignOfferStrategy"is shown in Figure V-25. The sequences are deduced by mapping input/output messages.


Petals BPM | Open Source BPMN 2.0 Editor - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils ?
http://bpmneditor.ebmwebsourcing.com/
Google
Petals BPM | Open Source BP...
Geasy Diagram Editor Sample App
Help us to improve Petals BPM !
petals bpm
File Edit Project Help
BPMN 2.0 Palette
Start Timer
Intermediate Throwing Message
Intermediate Catching Message
Intermediate Catching Timer
End
End Message
Terminate
Gateways
Exclusive Gateway
Parallel Gateway
Welcome in Petals BPM
*DesignOfferStrategy
Manu
Gather external information about environment
information
Operation Mediator
Lane
Invoke "Gather external information about environment"
Invoke "Develop goals"
Invoke "Manage offering life-cycle"
Properties
Diagram Validation
Terminé

Name: Gather external informat
Documentation:

FIGURE V - 25 COLLABROATIVE PROCESS OF "DESIGN OFFER STRATEGY"

IV. Conclusion

The whole MISE 2.0 abstract level work is based on the collaborative ontology. The model transformation rules are defined according to the ontology. The selection of business service has to choose instances of the collaborative ontology. But in the collaborative ontology, the MIT process handbook is the only one ontology, which has been included by the collaborative ontology. This leads to that the number of the instances limits the efficiency of the deduction of collaborative process. To find more instances in different domains, the ontology definition tool is developed by Tiexin Wang in his master internship in our lab from January 2012 to August 2012. The main theory of the ontology definition tool is represented in Figure V-26.

The theory is to select the instances from other domain ontologies and insert them to the collaborative ontology through the single one-to-one concepts mapping among metamodels. For example, between crisis metamodel and collaborative ontology, the mapping can be defined, as the concept A of crisis metamodel is same with the concept B of the collaborative ontology. So the instances of concept A can be inserted to the collaborative ontology as the instances of concept B. Furthermore, the concept B may have relations with other concepts. The new instances, which come form the concept A, have potentially to be linked with existing new instances.

The main interface of ontology definition tool is shown in Figure V-27. The user could upload new metamodel or ontology in .uml or .owl format as the ontology, which needs instances. The user could directly create instances in the tool or download an empty Excel form, which is structured according to the collaborative metamodel. By fulfilling and uploading the Excel form, the user may complete the collaborative ontology. The user could also upload another ontology, define the mapping rules between the two ontologies and insert instances automatically.

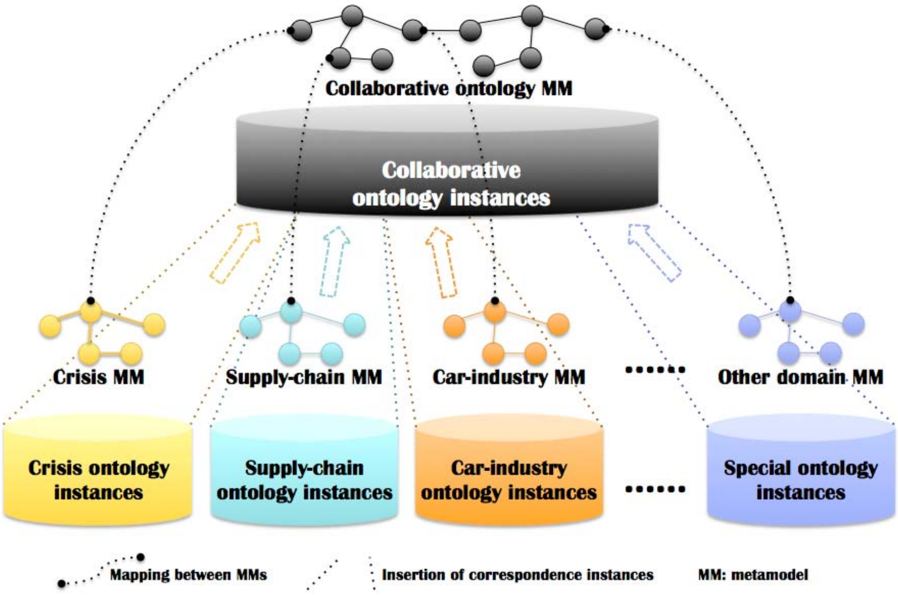
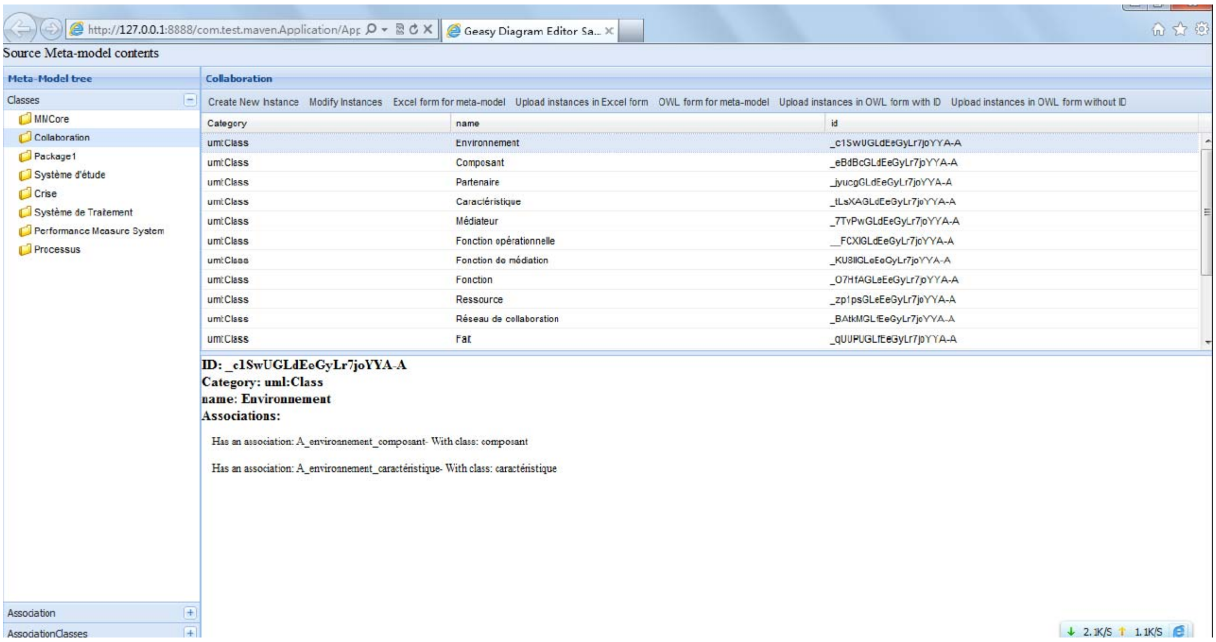

Collaborative ontology MM
Collaborative ontology instances
Crisis MM
Supply-chain MM
Car-industry MM
······
Other domain MM
Crisis ontology instances
Supply-chain ontology instances
Car-industry ontology instances
······
Special ontology instances
Mapping between MMs
Insertion of correspondence instances
MM: metamodel

FIGURE V - 26 THEORY OF ONTOLOGY DEFINISTION TOOL

The window on the left side of Figure V-27 is the metamodel tree of source metamodel (the core metamodel of MISE 3.0). The tree includes three menus: "Classes", "Association" and "Association Classes". In the menu of "Classes", there are the packages in the metamodel. On the right hand side of Figure V-27 ("Collaboration window"), it is the list of classes in the selected package of "Classes" menu. The window below shows the information of the selected class. On the top of "Collaboration window", there is a menu bar. The user can create new instance, modify instance, export Excel form for metamodel, upload Excel form, transfer metamodel to .owl format, and upload instances from .owl with ID and without ID.



http://127.0.0.1:8888/com.test.maven.Application/App
Geasy Diagram Editor Sa...
Source Meta-model contents
Meta-Model tree
Classes
MNCore
Collaboration
Package1
Système d'étude
Crise
Système de Traitement
Performance Measure System
Processus
Association
AssociationClasses
Collaboration
Create New Instance
Modify Instances
Excel form for meta-model
Upload instances in Excel form
OWL form for meta-model
Upload instances in OWL form with ID
Upload instances in OWL form without ID
2.1K/S
1.1K/S

Category	name	id
uml:Class	Environnement	_c1SwUGLdEeGyLr7joYYA-A
uml:Class	Composant	_eBdBcGLdEeGyLr7joYYA-A
uml:Class	Partenaire	_jyucgGLdEeGyLr7joYYA-A
uml:Class	Caractéristique	_tLsXAGLdEeGyLr7joYYA-A
uml:Class	Médiateur	_7TvPwGLdEeGyLr7joYYA-A
uml:Class	Fonction opérationnelle	__FCXIGLdEeGyLr7joYYA-A
uml:Class	Fonction de médiation	_KU8llGLeEeGyLr7joYYA-A
uml:Class	Fonction	_O7HfAGLeEeGyLr7joYYA-A
uml:Class	Ressource	_zp1psGLeEeGyLr7joYYA-A
uml:Class	Réseau de collaboration	_BAtkMGLfEeGyLr7joYYA-A
uml:Class	Fait	_qUUPUGLfEeGyLr7joYYA-A

ID: _c1SwUGLdEeGyLr7joYYA-A
Category: uml:Class
name: Environnement
Associations:

Has an association: A_environnement_composant- With class: composant

Has an association: A_environnement_caractéristique- With class: caractéristique

FIGURE V - 27 MAIN INTERFACE OF ONTOLOGY DEFINITION TOOL

General Conclusion

In collaborative situation, all the partners come with collaborative objectives and their own objectives to achieve and business services to share. They expect to combine their own business services with suitable ones from other partners to work towards their common objectives. In addition, collaborative business process is a combination of business functions, which is inter-linked and filled with sequences and orders. With these needs, the selection of business service, which are objective-oriented and the creation of business process are absolute essentials in collaboration world. Considering self-updating and re-building of collaborative business process, we shall design an automatic way to deal with service selection and process creation in design level. Further more from the implementation viewpoint, first the supporting software tool deals with a collaborative situation, all the partners may use the software in the same time or individually. Secondly, in order to interact with other software tools (which are developed in our lab), the software should be able to deploy on an ESB (Enterprise Service Bus). These lead that the software tool involved in the methodology should be a web service. SaaSs (Software as a Service) seems to be a good solution.

The research work of this thesis describes the design and development of MISE 2.0 abstract level. The research work improves the CIM level work of MISE 1.0. Based on the dynamically deduced collaborative process of MISE 1.0 (which has also been improved), the work of this thesis defines the collaborative process cartography, which classifies the collaborative process into strategy, operation and support level. The design of MISE 2.0 abstract level can be summarized as four main phases:

- Phase 1: Knowledge Gathering. The knowledge in this phase covers the target collaborative situation. In the work of Dr. Vatcharaphun Rajsiri, the initial knowledge is structured according to *collaborative network*, *partners* and *common goal*. In the work of Dr. Sébastien Truptil, the *shared functions* of partners are added to the initial knowledge. In the work of this thesis, the above two results are combined together and improved. The collaborative network model and function model represent and define the initial collaborative situation. The collaborative network model does not only collect the *collaborative network*, *partners* and *partner relations* but also *sub collaborative network*, and *collaborative objectives*. The function model represents the information concerning *shared partner functions* and *input/output messages*.
- Phase 2: Knowledge Transferring. In this phase, the collaborative ontology and transformation rules are defined to transfer the collaboration concepts to the mediation concepts in the collaborative ontology. The knowledge in this phase covers the mediation concepts and instances in the collaborative ontology. There are five groups of transformation rules: create Mediator, create Mediator Relationship, create Generated Mediator Function, link Generated Mediator Function to Mediator, and Create Inter Mediator Function. With the transformation rules, the mediation concepts are deduced, but there is not enough knowledge for the extraction of collaborative process, so the next phase comes.
- Phase 3: Knowledge Completing. The knowledge of this phase presents the matching between objective and functions. In this phase, one methodology is developed: business service selection to choose functions to achieve objectives by linking the functions and objectives to the instances of the collaborative ontology by using *same as* and *near by* relations.
- Phase 4: Knowledge Extracting. The knowledge covers the *collaborative process* extraction and *sequence/gateway* deduction. In this phase, the deduction rules are defined to extract the collaborative process cartography and collaborative processes. To complete the *sequence* and the *gateway*, the method of sequence deduction is developed.

To support the models, the collaborative ontology, the transformation rules and the methodologies, the Mediator modeling 2ool is designed and implemented. Software as a Service (SaaS) is increasingly being used for this purpose. It allows users to utilize an application in a Web Client as a rich application. No complex client-side installation is required. For the implementation of the modeling tool of the abstract level of MISE 2.0, SaaS is a suitable and popular choice. GWT (Google Web Toolkit), XML, JDOM and Eclipse have been chosen as developing tools. The tool mainly implements the following functions: i) define the collaborative network model and the function model, ii) import the instances of the collaborative ontology and help to choose *same as* and *near by* instances for the defined objectives, functions and input/output messages, and iii) transfer the defined models to the collaborative process cartography by implementing the transformation rules of the collaborative ontology.

Because the collaborative ontology is the most important part of the design, and it needs large number of instances (from various domains such as crisis management, supply-chain, etc.), a collaborative definition tool is developed by Tiexin Wang. This ontology definition tool can import ontologies in OWL RDF format or metamodel in UML format. It can extract the instances of the ontology and then insert them into our ontology thanks to one-to-one mapping of concepts.

The abstract level work of MISE 2.0 has been reviewed in the previous paragraphs. Now, the advantages and the dis-advantages of the research work are discussed. The strong points of the research work of MISE 2.0 abstract level are:

- The collaborative network model defines objectives and collaborative/sub networks by the decomposition of the whole network. This can be used easily to verify that the group of partners complete main tasks or achieve main collaborative objectives. The organizations of the collaborative networks are clearer and more efficient.
- The function model defines the shared function and the same as/near by instances of the collaborative ontology. This helps the dynamic selection of functions for each collaborative objective and also enlarges the instances of the collaborative ontology by using same as/near by relations with existing instances. Furthermore, the user only provides the information concerning function, input/output message and the same as/near by relations. The function model is easy to handle. It decreases the workload of the user.
- The deduction of gateway and sequence is improved. In the work of Dr. Vatcharaphun Rajsiri, she deduced the sequence and gateway by linking the output message of one function to the same input message of another function. The method of Dr. Vatcharaphun Rajsiri leads that there are loops in the collaborative process or there are too many sequences. In this thesis, the objective-based method is developed. It avoids the problem mentioned above.
- The collaborative process cartography is deduced. Comparing with one single and complex collaborative process, in the work of this thesis, the collaborative process is presented as a main process and several sub processes. The tasks of the main process are classified to strategy, operation and support. This kind of collaborative process is more understandable. The workers, operators or mangers from different levels of the collaboration and departments of organizations can find the part of the collaborative process, which they are concerned.

However, any system has its weak points. These are summarized as follows:

- The same as/near by relations are manually made. Because the collaborative ontology does not yet contain a large number of instances. It limits the semantic function of the collaborative ontology. In the following works, we should seek more instances for the collaborative ontology, improve the semantic selection and try to make the same as/near by relations automatical.
- With regard to the knowledge-gathering phase, users provide models. This phase is manually completed. However, some current research works are dealing with this specific point. The Event-Driven Architecture is able to provide a technical infrastructure that allows devices, sensors and other services to publish their messages (as events). These events may then be used to feed the situational modeling editor. This would finally be a way to link the Internet of things (devices) with the Internet of knowledge (ontology) to drive the Internet of services (web services).
- As regards the gateways in BPMN, we can only finish the transformation of exclusive, inclusive and parallel gateways. These are enough for the following BPEL transformation in the concrete level. But we still have not covered all the gateways defined in BPMN. We have to admit that the collaborative process cartography is enough for process modeling but not good enough for covering all kinds of knowledge. As regards events, we only use start, end, start message and end message events. Thus, all the BPMN-defined events are not covered.

In MISE 2.0 (the global picture is shown in Figure 1), the collaborative process cartography is the output of abstract level and the input of the concrete level. The abstract level work covers: knowledge gathering, process cartography and the transformation from the knowledge gathering phase to the process cartography phase (the work of this thesis). In the concrete level, the collaborative process is transferred to the BPEL based technical process: the business activities of the collaborative process are replaced by the web services (in a one-to-one or many-to-many approach). The web services are selected automatically thanks to syntactic/semantic reconciliation. Nicolas Boissel-Dallier does this part of work in his PhD work (2009-2012). The PhD work covers: the MIS deployment and the transformation from the process cartography to the MIS deployment. Going through the design-time of abstract level and concrete level, an ESB based MIS can be developed. Regarding the agility management, if the technical, logical or business errors occur in the run-time of the MIS, the MIS must know how to fix or avoid these errors. The PhD work of Anne-Marie Barthe is proposed (2010-2013). The work includes the detection of errors and the adaption of knowledge gathering, process cartography and MIS deployment. Until now, all the work is focus on the functional requirements. For the non-functional requirement (e.g. security, time limits, weather limits, etc.), the PhD work of Sarah Zribi started in 2010. She adds the non-functional aspects to each shared function of partners. The work enriches the information of function. It improves quality of the business to technical transformation by refining the selection of web services. The MISE 2.0 project gathers collaborative knowledge, transfers it to collaborative process cartography, adds non-functional requirement to the business functions, transfers the collaborative process cartography to the technical collaborative process, deploys the technical process on the ESB to build the run-time MIS and finally defines the agility management (detection of errors and adaptation of solutions) for the run-time MIS.

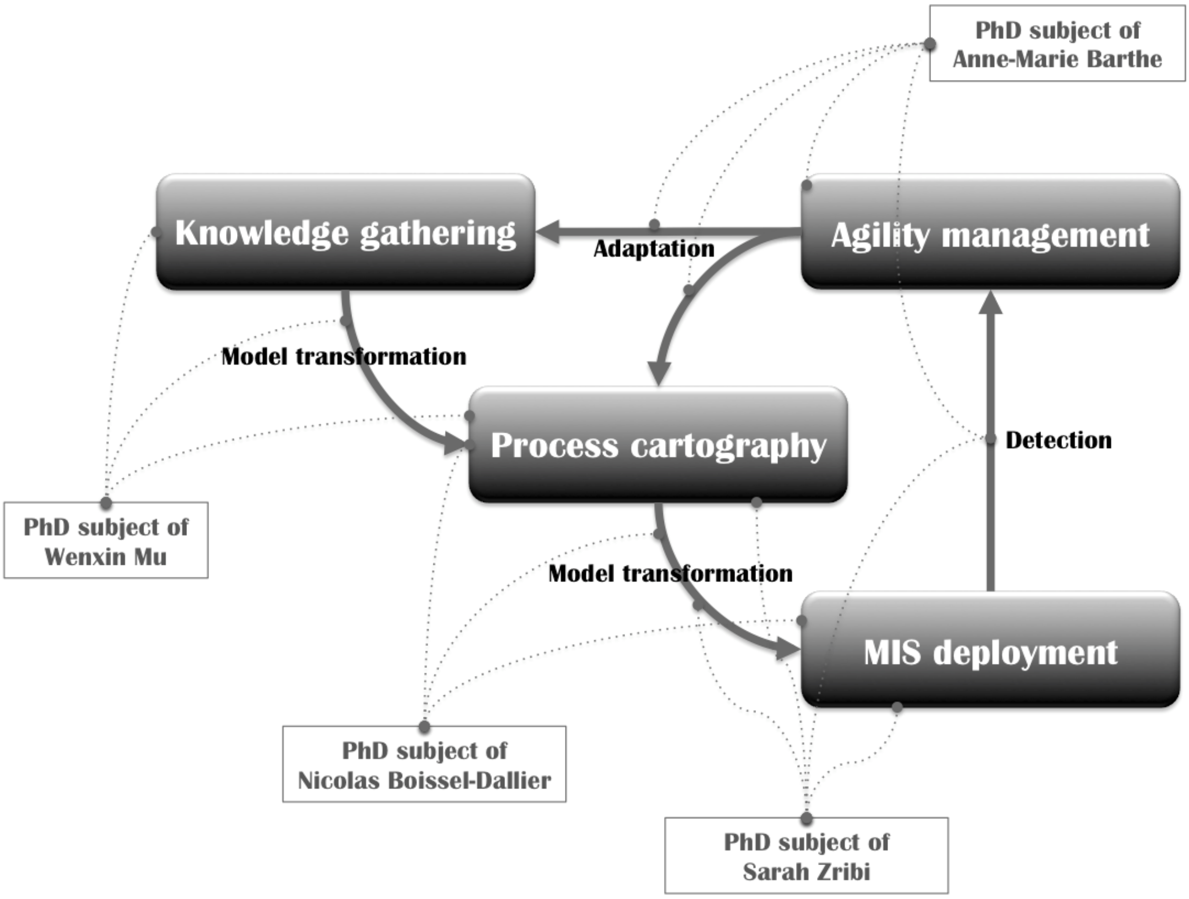


FIGURE 1 GLOBAL VIEW OF MISE 2.0

The development of MISE 3.0 has been launched in our lab in 2011. The whole MISE project (MISE 1.0, MISE 2.0 and MISE 3.0) is summarized in Figure 2. The MISE 3.0 project covers the PhD subjects of Guillaume Macé-Ramete (2011-2014), Aurèlie Montarnal (2012-2015) and Loïc Bidoux (2012-2015). The MISE 3.0 improves the work of MISE 2.0 from following points:

- Characterization continuing: real time detection of events. In the whole design-time and run-time, the events can be detected and handle at the first time.

- Performance indicators deducing: indicators measure the performances of business function and web services. For each collaborative function or web service, the indicator is added to measure: for example, the launching time, the responding time, the correction of the output message, etc.
- Workflow monitoring: from both functional view and non-functional view, the user could observe which step is the workflow being executed now.
- Cloud deploying: the whole MISE tools and the deployed MIS is integrated in the cloud platform. The MISE tools can be uploaded to the cloud in a PaaS (Platform as a Service) as SaaS.
- Detection by performance monitoring: with indicators of performance, the detection of agility management can be improved. The indicators can directly launch the detection by identifying the feedback of performance.
- Decision-making transiting: for each step of model transformation, the decision-making mechanism is added. This module suggests the best solution for the choice of functions, processes, web services and so on.

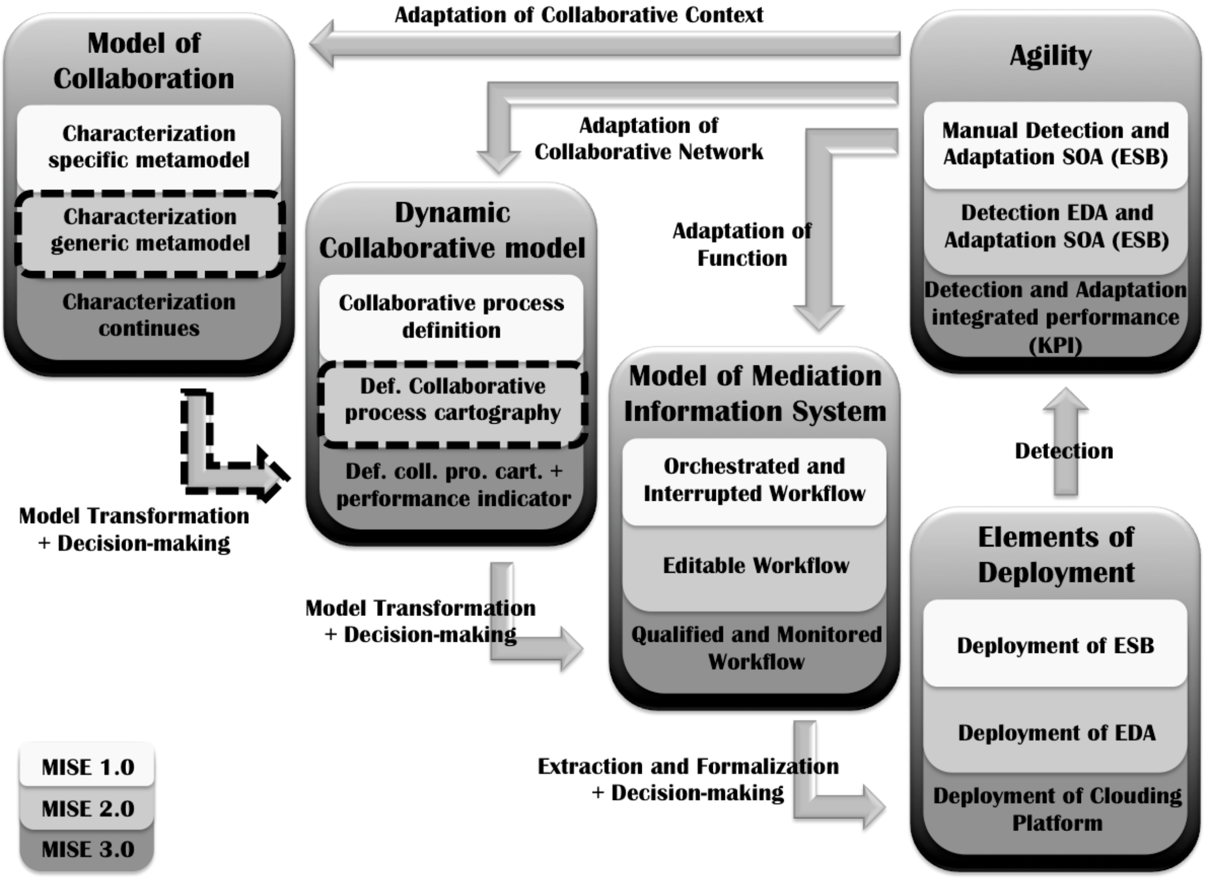

Adaptation of Collaborative Context
Model of Collaboration
Characterization specific metamodel
Characterization generic metamodel
Characterization continues
Agility
Manual Detection and Adaptation SOA (ESB)
Detection EDA and Adaptation SOA (ESB)
Detection and Adaptation integrated performance (KPI)
Adaptation of Collaborative Network
Adaptation of Function
Dynamic Collaborative model
Collaborative process definition
Def. Collaborative process cartography
Def. coll. pro. cart. + performance indicator
Model of Mediation Information System
Orchestrated and Interrupted Workflow
Editable Workflow
Qualified and Monitored Workflow
Detection
Elements of Deployment
Deployment of ESB
Deployment of EDA
Deployment of Clouding Platform
Model Transformation + Decision-making
Model Transformation + Decision-making
Extraction and Formalization + Decision-making
MISE 1.0
MISE 2.0
MISE 3.0

FIGURE 2 GLOBAL PICTURE OF MISE PROJECT

The research works of this thesis are the modules in the dash line boxes in Figure 2. They deal with the characterization of generic metamodel, the definition of the collaborative process cartography and the model transformation between them.

Acronyms

ADM	Architecture Development Mode
AIF	ATHENA Interoperability Framework
ARIS	Architecture of Integrated Information Systems
ATL	Atlas Transformation Language
BPEL	Business Process Execution Language
BPMA	Business Process Model Abstraction
BPM	Business Process Management
BPMN	Business Process Model Notation
BST	Business Level of Dr. Sébastien Truptil
BVR	Business Level of Dr. Vatcharaphun RAJSIRI
C4ISR	Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance
CO	Collaborative Ontology
CO1	Collaborative Ontology of MISE 1.0
CO2	Collaborative Ontology of MISE 2.0
CIM	Computing Independent Model
CIMOSA	Computer Integrated Manufacturing Open System Architecture
CPO	Collaborative Process Ontology
DoDAF	Department of Defense Architecture Framework
DOGMA	Developing Ontology-Grounded Methods and Applications
EA	Enterprise Architecture
EF	Enterprise Interoperability Framework
EIA	Enterprise Integration Architecture
EIF	Enterprise Interoperability Framework
EPC	Event Process Chain
ESA	Enterprise Software Application
ESB	Enterprise Service Bus
FEA	Federal Enterprise Architecture

FEAF	Federal Enterprise Architecture Framework
F-logic	Frame Logic
GERA	Generic Enterprise Reference Architecture
GERAM	Generic Enterprise Reference Architecture and Methodology
GIM	GRAI Integrated Methodology
GMF	Graphical Modeling Framework
GWT	Google Web Toolkit
ICT	Information and Communication Technology
IEM	Integrated Enterprise Modeling
IDEAS	Interoperability Development for Enterprise Application and Software
IDEF	Integration Definition for Function Modeling
InterOp NoE	Interoperability Research for Networked Enterprises Applications and Software - Network of Excellence
IOCM	Inter-Organizational Collaborative Model
IS	Information System
ISO	International Organization for Standardization
IsyCri	Interopérabilité des Systèmes en Situation de Crise
KIF	Knowledge Interchange Format
KM	Knowledge Machine
MIS	Mediation Information System
MDA	Model Driven Achitecture
MDE	Model Driven Engineering
MDSD	Model Driven Software Development
MDI	Model Driven Interoperability
MIS	Mediation Information System
MISE	Mediation Information System Engineering
MISE 1.0	Mediation Information System Engineering Version 1.0
MISE 2.0	Mediation Information System Engineering Version 2.0
MOD	Ministry of Defense
MODAF	British Ministry of Defense Architecture Framework
MOF	Meta-Object Facility

ODP	Open Distributed Processing
OIL	Ontology Inference Layer
OKBC	Open Knowledge Base Connectivity
OMCK	Organization Model based on Collaborative Knowledge
OMG	Object Management Group
OMOA	Organization Model in three levels of Abstraction
OS	Organization structure
OWL	Web Ontology Language
PIM	Platform Independent Model
PIM4SOA	Platform Independent Model for Server Oriented Architecture
PM	Product Manufacture
PSM	Platform Specific Model
RCC	Relations of Collaborative life Cycle
RCE	Relations of Collaborative Elements
RCL	Relations of Collaborative Levels
RDF	Resource Description Framework
RM	Role model
SaaS	Software as a Service
SADT	Structured Analysis and Design Technique
SOA	Service Oriented Architecture
TAFIM	Technical Architecture Framework for Information Management
TAFIM TRM	Technical Architecture Framework for Information Management Technical Reference Model
TOGAF	Open Group Architecture Framework
UJ	Figure I-16
UML	Unified Modeling Language
UPDM	Unified Profile for DoDAF/MODAF
W3C	The World Wide Web Consortium
WSDL	Web Service Description Language
XML	Extensible Markup Language
XPDL	XML Process Definition Language

XSLT	Extensible Stylesheet Language Transformations

References

(Van Der Aalst, 2003) Van Der Aalst, W.M.P. (2003). Patterns and xpdl: A critical evaluation of the xml process definition language. Queensland University of Technology, Tech. Rep. FIT-TR-2003-06.

(Van Der Aalst and Ter Hofstede, 2000) Van Der Aalst, W.M.P., and Ter Hofstede, A.H.M. (2000). Verification of workflow task structures: A petri-net-baset approach. Information Systems *25*, 43–69.

(Van Der Aalst and Van Hee, 2004) Van Der Aalst, W.M.P., and Van Hee, K.M. (2004). Workflow management: models, methods, and systems (The MIT press).

(Abt et al., 2009) Abt, V., Vigier, F., and Schneider, M. (2009). Enterprise Business Modelling Languages Applied to Farm Enterprise: A Case Study for IDEF0, GRAI Grid, and AMS Languages. Advances in Modeling Agricultural Systems 167–191.

(AG, 2009) AG, I.D.S.S. (2009). Method ARIS 7.1 (PDF).

(Benaben et al., 2010) Benaben, F., Mu, W., and Truptil, S. (2010). Information Systems design for emerging ecosystems. DEST-IEEE.

(Benaben et al., 2008) Benaben, F., Touzi, J., Rajsiri, V., and Lorré, J.P. (2008). Mediation Information System Design in a collaborative SOA context through a MDD Approach. Proceedings of MDISIS'08 1–17.

(Benaben et al., 2006) Benaben, F., Touzi, J., Rajsiri, V., and Pingaud, H. (2006). Collaborative Information System Design. In: AIM 2006 Information Systems and Collaboration: State of the Art and Perspectives. GI-Edition, Lecture Notes in Informatics 281–296.

(Benjamin et al., 1994) Benjamin, P.C., Menzel, C., Mayer, R., Fillion, F., Futrell, M., deWitte, P., and Lingineni, M. (1994). Idef5 method report. Knowledge Based Systems, Inc.

(Bernstein et al., 1995) Bernstein, A., Dellarocas, C., Malone, T.W., and Quimby, J. (1995). Software tools for a process handbook. Data Engineering *51*, 41.

(Berre et al., 2007) Berre, A., Elvesæter, B., Figay, N., Guglielmina, C., Johnsen, S., Karlsen, D., Knothe, T., and Lippe, S. (2007). The ATHENA Interoperability Framework. In Enterprise Interoperability II, pp. 569–580.

(Bézivin, 2004) Bézivin, J. (2004). In Search of a Basic Principle for Model Driven Engineering. UP GRADE 21.

(Bézivin, 2005) Bézivin, J. (2005). On the unification power of models. Software and Systems Modeling *4*, 171–188.

(Biggs, 2005) Biggs, B. (2005). Ministry of Defence Architectural Framework (MODAF). IEE Digest *2005*, 43–82.

(Bobrik et al., 2007) Bobrik, R., Reichert, M., and Bauer, T. (2007). View-based process visualization. In Proceedings of the 5th International Conference on Business Process Management, pp. 88–95.

(Börger, 2011) Börger, E. (2011). Approaches to modeling business processes: a critical analysis of BPMN, workflow patterns and YAWL. Software & Systems Modeling.

(Bourey et al., 2007) Bourey, J.-P., Grangel, R., and Doumeingts, G. (2007). Report on Model Driven Interoperability. Deliverable DTG 2.

(Bouslimi et al., 2009) Bouslimi, I., Ghédira, K., and Hanachi, C. (2009). An agent-based organizational model for cooperative information gathering. Advanced Internet Based Systems and Applications 180–189.

(Brabänder and Daris, 2007) Brabänder, E., and Davis, R. (2007). ARIS design platform–getting started with BPM (London: Springer-Verlag London Limited).

(C4ISR, 1997) C4ISR (1997). C4ISR Architecture Framework Version 2.0. AWG - US department of defence.

(Camarinha-Matos and Afsarmanesh, 2008) Camarinha-Matos, L.M., and Afsarmanesh, H. (2008). On reference models for collaborative networked organizations. International Journal of Production Research *46*, 2453.

(Chen et al., 2008) Chen, D., Doumeingts, G., and Vernadat, F. (2008). Architectures for enterprise integration and interoperability: Past, present and future. Computers in Industry *59*, 647–659.

(Chen et al., 1997) Chen, D., Vallespir, B., and Doumeingts, G. (1997). GRAI integrated methodology and its mapping onto generic enterprise reference architecture and methodology. Computers in Industry *33*, 387–394.

(Clark et al., 2004) Clark, T., Evans, A., Sammut, P., and Willans, J. (2004). Applied Metamodelling: A Foundation for Language Driven Development Version 0.1.

(Council, 1999) Council, C.I.O. (1999). Federal Enterprise Architecture Framework (FEAF)–Version 1.1 (September).

(Dorn et al., 2010) Dorn, C., Burkhart, T., Werth, D., and Dustdar, S. (2010). Self-adjusting Recommendations for People-Driven Ad-Hoc Processes. In Business Process Management, R. Hull, J. Mendling, and S. Tai, eds. (Springer Berlin / Heidelberg), pp. 327–342.

(Doumeingts et al., 2006) Doumeingts, G., Vallespir, B., and Chen, D. (2006). GRAI GridDecisional modelling. Handbook on Architectures of Information Systems 321–346.

(Dr et al., 1996) Dr, P., Papazoglou, M., Wiederhold, G., Wiederhold, G., Genesereth, M., and Genesereth, M. (1996). The Conceptual Basis for Mediation Services. IEEE EXPERT *12*, 38–47.

(DTIC, 1996) DTIC (1996). Technical Architecture Framework for Information Management. Volumes 1-8. Version 3.0 (Computer Diskette).

(Force, 1999) Force, I.I.. (1999). GERAM: Generalised Enterprise Reference Architecture and Methodology. IFIP-IFAC Task Force on Architectures for Enterprise Integration, Tech. Rep.

(Fox and Gruninger, 1998) Fox, M., and Gruninger, M. (1998). Enterprise Modelling. AI Magazine 19, 109.

(Godoy, 2005) Godoy, C.P. (2005). Knowledge-Based Reasoning over the Web. Citeseer.

(Gruber, 1993) Gruber, T.R. (1993). A translation approach to portable ontology specifications. Knowledge Acquisition *5*, 199–220.

(Guarino and Pierdaniele, 1995) Guarino, N., and Giaretta, P. (1995). Ontologies and Knowledge Bases - Towards a Terminological Clarification. In Towards Very Large Knowledge Bases, (IOS Press, Amsterdam, The Netherlands), pp. 32, 25.

(Ter Hofstede et al., 2003) Ter Hofstede, A., Van Der Aalst, W., and Weske, M. (2003). Business Process Management: A Survey. In Business Process Management, M. Weske, ed. (Springer Berlin / Heidelberg), pp. 1019–1019.

(IEEE, 1990) IEEE (1990). IEEE: Standard Computer Dictionary - A Compilation of IEEE Standard Computer Glossaries.

(IDEAS, 2003) IDEAS (2003). A gap Analysis - Required activities in Research, Technology and standardisation to close the RTS Gap - Roadmaps and Recommendations on RTS activites.

(Idee, 2009) Idee, N. (2009). ARIS Solution Business driven SOA.

(IDEF0, 1993) IDEF0 (1993). Announcing the Standard for INTEGRATION DEFINITION FOR FUNCTION MODELING (IDEF0).

(ISO 9000, 2005) ISO 9000 (2005). NF EN ISO 9000 Systèmes de management de la qualité - Principes essentiels et vocabulaire.

(ISO 9000 X50-130, 2005) ISO 9000 X50-130 (2005). NF EN ISO 9000 X50-130 Systèmes de management de la qualité - Principes essentiels et vocabulaire.

(Jiang et al., 2011) Jiang, J., Dignum, V., Tan, Y.H., and Overbeek, S. (2011). A context-aware inter-organizational collaboration model applied to international trade. Electronic Government 308–319.

(Jong and Dietz, 2010) Jong, J., and Dietz, J.L.G. (2010). Understanding the realization of organizations. Advances in Enterprise Engineering IV 31–49.

(Jouault and Kurtev, 2006) Jouault, F., and Kurtev, I. (2006). Transforming Models with ATL. In Satellite Events at the MoDELS 2005 Conference, J.-M. Bruel, ed. (Springer Berlin / Heidelberg), pp. 128–138.

(Katzy et al., 2005) Katzy, B., Zhang, C., and Löh, H. (2005). Reference models for virtual organisations. Virtual Organizations 45–58.

(Kelley, 1975) Kelley, J.L. (1975). General topology (Springer Verlag).

(Ko, 2009) Ko, R.K.L. (2009). A computer scientist's introductory guide to business process management (BPM). Crossroads *15*, 4:11–4:18.

(Komoto et al., 2008) Komoto, H., D'amelio, V., Echavarria, E., Tomiyama, T., and others (2008). A review of function modeling: Approaches and applications. Artificial Intelligence for Engineering Design, Analysis and Manufacturing *22*, 147–169.

(Kosanke, 1995) Kosanke, K. (1995). CIMOSA -- Overview and status. Computers in Industry *27*, 101–109.

(Kosanke et al., 1999) Kosanke, K., Vernadat, F., and Zelm, M. (1999). CIMOSA: enterprise engineering and integration. Computers in Industry *40*, 83–97.

(Konstantas et al., 2005) Konstantas, D., Bourrières, J.-P., Léonard, M., and Boudjlida, N. (2005). Interoperability of Enterprise Software and Applications (Springer-Verlag).

(Krafzig et al., 2005) Krafzig, D., Banke, K., and Slama, D. (2005). Enterprise SOA: service-oriented architecture best practices (Prentice Hall PTR).

(Lankhorst, 2009) Lankhorst, M. (2009). Enterprise Architecture at Work: Modelling, Communication and Analysis.

(Lee et al., 2001) Lee, T.B., Hendler, J., Lassila, O., and others (2001). The semantic web. Scientific American *284*, 34–43.

(Li and Iijima, 2007a) Li, B., and Iijima, J. (2007a). A situation calculus based approach to dynamic management of e-business services. Journal of Systems Science and Systems Engineering *16*, 336–355.

(Li and Iijima, 2007b) Li, B., and Iijima, J. (2007b). Architecture on a hybrid business process design and verification system. In Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference On, pp. 6199–6204.

(Li and Iijima, 2007c) Li, B., and Iijima, J. (2007c). Bridging the gap between XPDL and situation calculus: A hybrid approach for business process verification. In Proceedings of the 5th International Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems (MSVVEIS),.

(Li et al., 2009) Li, Q., Chen, Y.-L., Li, Q., and Chen, Y.-L. (2009). IDEF0 Function Modeling. In Modeling and Analysis of Enterprise and Information Systems, (Springer Berlin Heidelberg), pp. 98–122.

(Lightsey, 2001) Lightsey, B. (2001). Systems engineering fundamentals (DTIC Document).

(Lohmann and Wolf, 2010) Lohmann, N., and Wolf, K. (2010). How to implement a theory of correctness in the area of business processes and services. Business Process Management 61–77.

(MacKenzie et al., 2006) MacKenzie, C.M., and others (2006). Reference model for service oriented architecture. Public Review Draft 2.

(Malone et al., 2003) Malone, T.W., Crowston, K., and Herman, G.A. (2003). Organizing business knowledge: the MIT process handbook (the MIT Press).

(Marley, 2008) Marley, S. (2008). Architectural Framework.

(McGuinness et al., 2004) McGuinness, D.L., Van Harmelen, F., and others (2004). OWL web ontology language overview. W3C Recommendation *10*, 2004–03.

(Mellor, 2004) Mellor, S.J. (2004). Principles of model-driven architecture (Addison-Wesley).

(Miller et al., 2003) Miller, J., Mukerji, J., and others (2003). MDA Guide Version 1.0. 1. Object Management Group *234*, 51.

(Milojicic et al., 2002) Milojicic, D.S., Kalogeraki, V., Lukose, R., Nagaraja, K., Pruyne, J., Richard, B., Rollins, S., and Xu, Z. (2002). Peer-to-peer computing (Citeseer).

(Missikoff et al., 2002) Missikoff, M., Navigli, R., and Velardi, P. (2002). Integrated approach to web ontology learning and engineering. Computer *35*, 60–63.

(Morley, 2006) Morley, C. (2006). UML pur l'analyse d'un système d'information. Le cahier des charges du maitre d'ouvrage (France: Dunod).

(De Moor et al., 2006) De Moor, A., De Leenheer, P., and Meersman, R. (2006). DOGMA-MESS: A meaning evolution support system for interorganizational ontology engineering. Conceptual Structures: Inspiration and Application 189–202.

(Navigli, 2004) Navigli, R. (2004). Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites. Computational Linguistics 30, 151–179.

(Neches et al., 1991) Neches, R., Fikes, R.E., Finin, T., Gruber, T., Patil, R., Swartout, W.R., and others (1991). Enabling technology for knowledge sharing. AI Magazine *12*, 36.

(Neiger et al., 2009) Neiger, D., Churilov, L., and Flitman, A. (2009). Business Objectives Modelling. In Value-Focused Business Process Engineering : a Systems Approach, (Boston, MA: Springer US), pp. 1–26.

(Nicolle et al., 2005) Nicolle, C., Simon, J.C., and Yetongnon, K. (2005). Interoperability of information systems. Encyclopedia of Information Science and Technology (III), M.Khosrow-Pour, Ed4, *1651*-1650.

(Noran, 2003) Noran, O.S. (2003). An analysis of the Zachman Framework for Enterprise Architecture from the GERAM. Annual Reviews in Control 27, 163–183.

(OMG, 2011) OMG (2011). Business Process Model and Notation (BPMN) Version 2.0.

(Patig et al., 2010) Patig, S., Casanova-Brito, V., and Vögeli, B. (2010). IT Requirements of Business Process Management in Practice – An Empirical Study. In Business Process Management, R. Hull, J. Mendling, and S. Tai, eds. (Springer Berlin / Heidelberg), pp. 13–28.

(Pépiot et al., 2007) Pépiot, G., Cheikhrouhou, N., Furbringer, J.-M., and Glardon, R. (2007). UECML: Unified Enterprise Competence Modelling Language. Computers in Industry *58*, 130–142.

(Pingaud, 2009) Pingaud, H. (2009). Rationalité du développement de l'interopérabilité dans les organisations. Management Des Technologies Organisationnelles, Presses De l'Ecole Des Mines De Paris, France, Pp19–30.

(Polyvyanyy et al., 2010) Polyvyanyy, A., García-Bañuelos, L., and Dumas, M. (2010). Structuring Acyclic Process Models. In Business Process Management, R. Hull, J. Mendling, and S. Tai, eds. (Springer Berlin / Heidelberg), pp. 276–293.

(Polyvyanyy et al., 2008) Polyvyanyy, A., Smirnov, S., and Weske, M. (2008). Process model abstraction: A slider approach. In Enterprise Distributed Object Computing Conference, 2008. EDOC'08. 12th International IEEE, pp. 325–331.

(Prasanna et al., 2009) Prasanna, R., Nisdaitul, F.M.B., Mohd, N.H.H.J., and Nurmaisara, Z. (2009). FRAMEWORK OF CONTENT MANAGEMENT SYSTEM FOR UITM E-LEARNING SYSTEM.

(PUBS, 1993) PUBS, F. (1993). Integration definition for function modelling (IDEF0). Federal Information Processing Standards Publication *183*.

(Rajsiri, 2010) Rajsiri, V. (2010). Knowledge-based system for collaborative process specification. Thèse en Système Industriels. INPT-ENSTIMAC.

(Rajsiri et al., 2010) Rajsiri, V., Lorré, J.-P., Bénaben, F., and Pingaud, H. (2010). Knowledge-based system for collaborative process specification. Computers in Industry *61*, 161–175.

(Rupietta, 1994) Rupietta, W. (1994). Organization models for cooperative office applications. In Database and Expert Systems Applications, pp. 114–124.

(Scheer and Schneider, 2006) Scheer, A.-W., and Schneider, K. (2006). ARIS — Architecture of Integrated Information Systems. In Handbook on Architectures of Information Systems, P. Bernus, K. Mertins, and G. Schmidt, eds. (Springer Berlin Heidelberg), pp. 605–623.

(Scheer, 2000) Scheer, A.W. (2000). ARIS-business process modeling (Springer Verlag).

(Smirnov et al., 2012) Smirnov, S., Reijers, H., Weske, M., and Nugteren, T. (2012). Business process model abstraction: a definition, catalog, and survey. Distributed and Parallel Databases *30*, 63–99.

(Smullyan, 1995) Smullyan, R.M. (1995). First-order logic (Dover Publications).

(Stahl and Völter, 2006) Stahl, T. (Tom), and Völter, M. (2006). Model-driven software development (John Wiley & Sons New York).

(Stein et al., 2008) Stein, S., Lauer, J., and Ivanov, K. (2008). ARIS Method Extension for Business-Driven SOA. WIRTSCHAFTSINFORMATIK *50*, 436–444.

(Touzi, 2007) Touzi, J. (2007). Aide à la conception de Système d'Information Collaboratif support de l'interoperabilité des entreprise. Thèse en Système Industriels. INPT-ENSTIMAC.

(Touzi et al., 2009) Touzi, J., Benaben, F., Pingaud, H., and Lorré, J.P. (2009). A model-driven approach for collaborative service-oriented architecture design. International Journal of Production Economics *121*, 5–20.

(Truptil, 2011) Truptil, S. (2011). Etude de l'approche de l'interopérabilité par médiation dans le cadre d'une dynamique de collaboration appliquée à la gestion de crise. Thèse en Système Industriels. INPT-ENSTIMAC.

(Truptil et al., 2008) Truptil, S., Benaben, F., Couget, P., Lauras, M., Chapurlat, V., and Pingaud, H. (2008). Interoperability of information systems in crisis management: Crisis modeling and metamodeling. ENTERPRISE INTEROPERABILITY III: NEW CHALLENGES AND INDUSTRIAL 583–594.

(Truptil et al., 2010) Truptil, S., Benaben, F., Lauras, M., and Pingaud, H. (2010). Mediation Information System Engineering for Interoperability support in crisis management. In I-ESA'10, (Coventry).

(Umheh et al., 2007) Umheh, N., Miller, A., and Dagli, C. (2007). TOGAF vs. DoDAF: Architecting Frameworks for Net-Centric Systems. pp. 19–23.

(Vamus and Panaich, 2009) Vamus, J., and Panaich, N. (2009). TOGAF 9 Survey Results Presentation.

(Velardi et al., 2001) Velardi, P., Fabriani, P., and Missikoff, M. (2001). Using text processing techniques to automatically enrich a domain ontology. In Proceedings of the International Conference on Formal Ontology in Information Systems-Volume 2001, pp. 270–284.

(Vernadat, 2007) Vernadat, F. (2007). Interoperable enterprise systems: Principles, concepts, and methods. Annual Reviews in Control *31*, 137–145.

(Vernadat, 2002) Vernadat, F. (2002). Enterprise modeling and integration (EMI): Current status and research perspectives. Annual Reviews in Control *26*, 15–25.

(White, 2005) White, S. (2005). Using BPMN to model a BPEL process. BPTrends *3*, 1–18.

(Wiederhold, 1992) Wiederhold, G. (1992). Mediators in the architecture of future information systems. Computer *25*, 38–49.

(Williams and Li, 1999) Williams, T.J., and Li, H. (1999). PERA AND GERAM–ENTERPRISE REFERENCE ARCHITECTURES IN ENTERPRISE INTEGRATION. p. 3.

(Zachman, 1987) Zachman, J.A. (1987). A framework for information systems architecture. IBM Systems Journal *26*, 276–292.

(Zaidat, 2005) Zaidat, A. (2005). Spécification d'un cadre d'ingénierie pour les réseaux d'organizations.

(Zelm et al., 1995) Zelm, M., Vernadat, F.B., and Kosanke, K. (1995). The CIMOSA business modelling process. Computers in Industry *27*, 123–142.

Annex A: Collaborative Metamodel

I. Collaborative Metamodel 154

I.1. Classes in Organizational View 154

I.2. Classes in Functional View 154

I.3. Classes in Informational View 155

I.4. Classes in Process View 156

I.5. Relations among Views 157

II. Transformation Rules in Collaborative Metamodel 160

I. Collaborative Metamodel

The collaborative metamodel of MISE 2.0 is shown in Figure 5. In the metamodel, there are four packages (organizational view, functional view, informational view and process view).

Each package manages one model. The organizational view (Figure 1) mainly stores the information concerning the collaborative network model (e.g. collaborative network, partners and objectives). The functional view (Figure 2) mainly manages activities, tasks or functions provided by partners and mediator. The informational view (Figure 3) is defined to confirm modeling elements in the IDEF1-based informational model. The process view (Figure 4) is used to present collaborative process model knowledge. We also define associations among packages. These packages tell us which functions are used to implement an objective, which messages are transferred among different functions, which mediator activities constitute collaborative process, and so on.

I.1. Classes in Organizational View

First of all, classes in the organizational view (Figure 1) of the collaborative metamodel are introduced as follows:

- The class *collaborative network* defines the organizations' network in the collaborative situation. One *collaborative network* can have several objectives. A sub collaborative network can implement one objective, which is defined in a higher-level collaborative network.
- The class *partner* defines a partner who is involved in a collaboration situation.
- The class *partner relationship* is linked to the association partner relationship. This class is used to store and define the partner relationship value. The *partner relationship* can be *strategy partner relationship*, *operation partner relationship* or *support partner relationship*.
- The class *objective* defines the objectives of the partners. The *objective* can be *strategy objective*, *operation objective* or *support objective*.

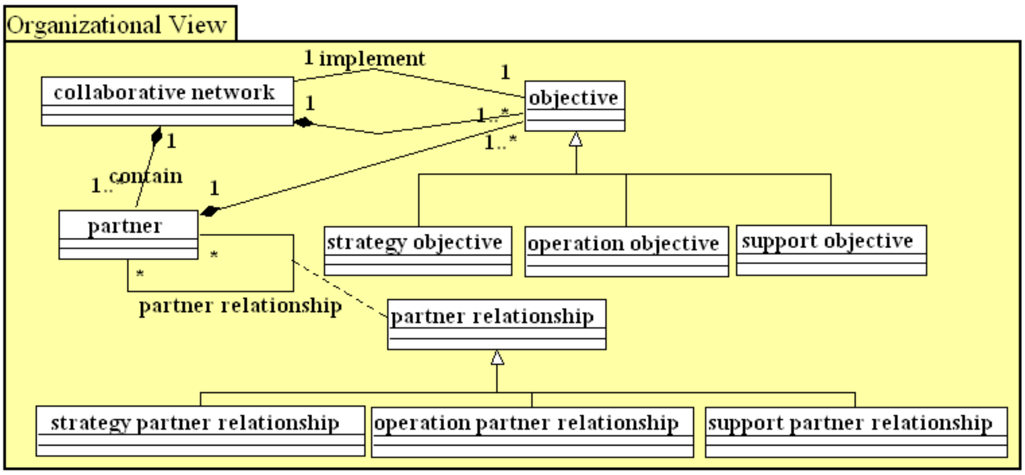


FIGURE 1 METAMODEL-ORGANIZATIONAL VIEW

I.2. Classes in Functional View

In the functional view (Figure 2), the partner activities are functions provided by the partners. The collaboration activities are functions provided by a mediator. The collaboration activities are deduced from the partner activities by transformation rules (section II).

Partner activity:

- The class *strategy activity* is used to provide strategy or decision service.
- The class *operation activity* is used to provide operational service.
- The class *support activity* is used to provide support activity.

Collaboration activity:

- The class *invoking activity* is used to receive a message from a partner, send a message to a partner or send and receive a message. These three activities are class *receiving activity*, class *calling activity* and class *receiving and calling activity*.
- The class *added value activity* does not send or receive any message. The activity is a service provided by the mediator (e.g. providing a required function that partners do not). The class *translating activity* is used to pass a message or change the format of a message.

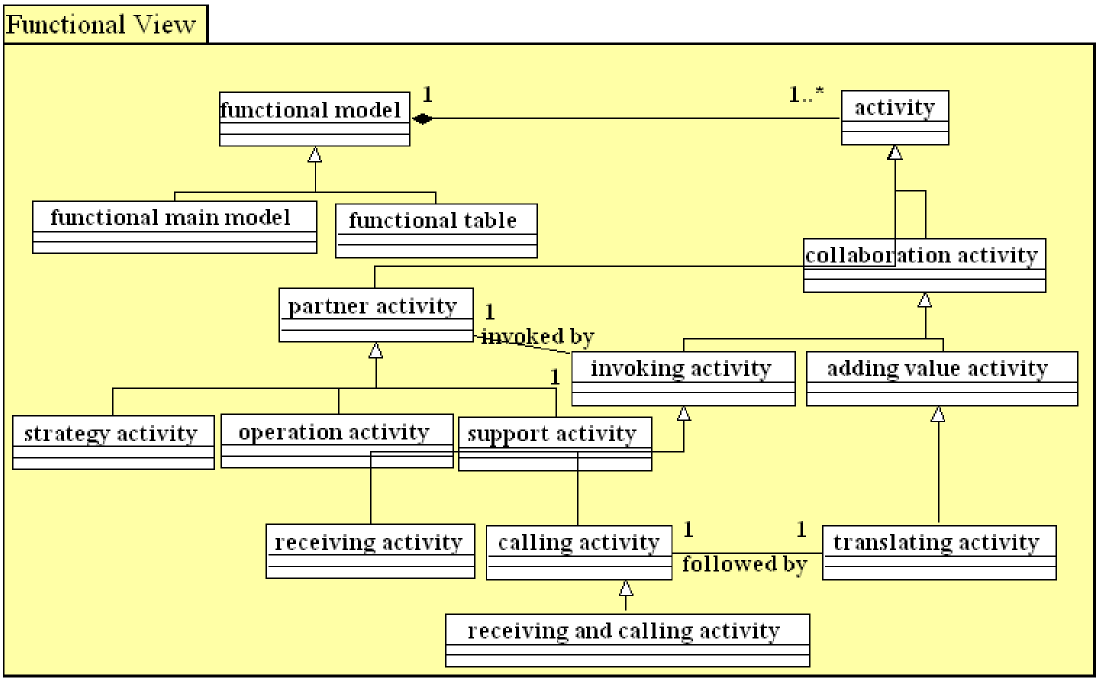


FIGURE 2 METAMODEL-FUNCTIONAL VIEW

I.3. Classes in Informational View

The exchanged business messages and process communication messages are managed in the informational view (Figure 3). The exchanged business message helps communication among partners in one type of collaborative process (strategy, operational or support) while the process communication message helps communication among different types of collaborative processes (strategy, operation and support).

The class *message* has got one or more *message relationship* associations with other messages. The *message relationship* refers to the *message relationship association.*

The class *exchange business message* contains:

- The class *strategy message*, which is one kind of *exchange business message*. The *strategy message* is transferred from one *strategy activity* to another *strategy activity*.
- The class *operation message*, which is one kind of *exchange business message*. The *operation message* is transferred from one *operation activity* to another *operation activity*.

- The class *support message*, which is one kind of *exchange business message*. The *support message* is transferred from one *support activity* to another *support activity*.

The class *process communication message* contains:

- The class *objective message*, which can be transferred from strategy process to operation process or from strategy process to support process.
- The class *feedback message*, which can be transferred from operation process to support process or from operation process to strategy process.
- The class *mean message*, which can be transferred from support process to strategy process or from support process to operation process.

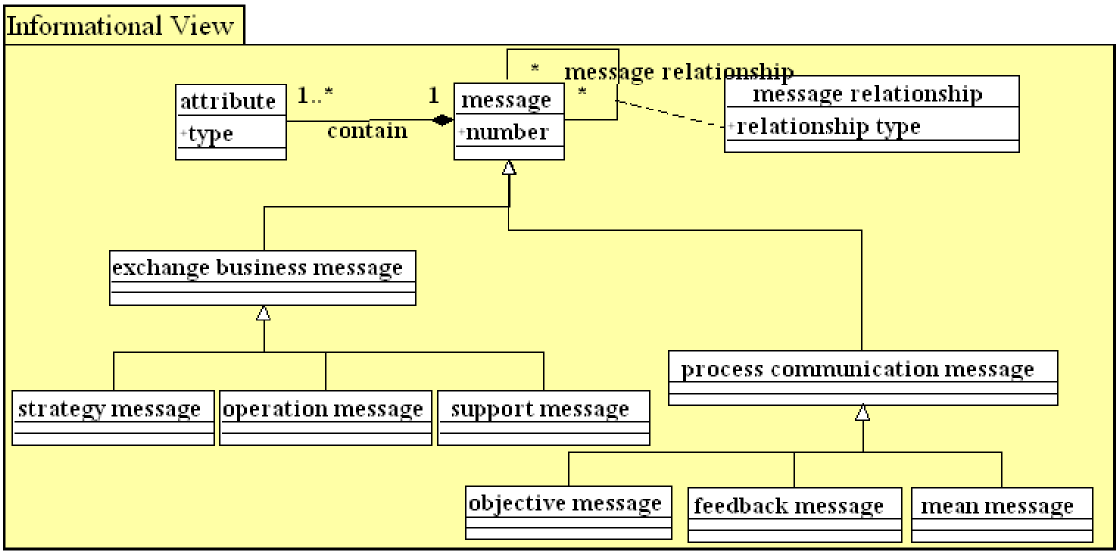


FIGURE 3 METAMODEL-INFORMATIONAL VIEW

I.4. Classes in Process View

As shown in Figure 4, the collaborative process contains three parts: strategy process, operation process and support process. Each type of process contains activities. Inside each process, the activities are organized through sequence flow. The class sequence flow links two activities. The sequence flow can also be linked to event or gateway. Outside each process, the message flow is used to communicate.

The class *collaborative process*:

- The class *strategy process* defines a strategy part of a collaborative process. One *strategy process* contains one or more collaborative strategy activities.
- The class *operation process* defines an operational part of the collaborative process. One *operation process* contains one or more collaborative operation activities.
- The class *support process* defines a support part of the collaborative process. One *support process* contains one or more collaborative support activities.

The class *process communication message flow*:

- The class *objective message flow* sends an objective information message from *strategy process* to *operation process* or from *strategy process* to *support process*.
- The class *feedback message flow* sends a feedback information message from *operation process* to *support process* or from *operation process* to *strategy process*.
- The class *mean message flow* sends a mean message from *support process* to *strategy process* or from *support process* to *operation process*.

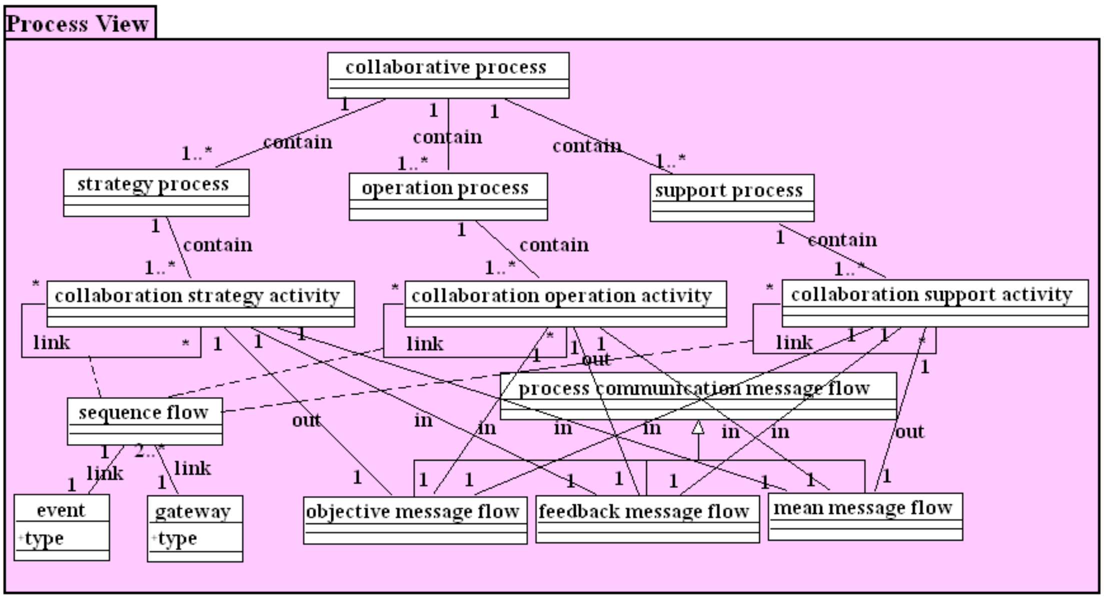


FIGURE 4 METAMODEL-PROCESS VIEW

I.5. Relations among Views

Associations between the Organizational view and the Functional view (Figure 5):

- The association *implement* from *collaborative network* to *functional model*: with this association, functional main model and functional table can be initialized.
- The association *implement* from *objective* to *activity*: one partner activity achieves a goal, which is described by the objective.

Associations between Functional View and Informational View (Figure 5) These Links are defined to give input message and output message to each function:

- The association *in* from *message* to *receiving activity*: one *receiving activity* only receives input message without output message.
- The association *out* from *message* to *calling activity*: one *calling activity* only sends one output message without input message.
- The association *in* and *out* from *message* to *receiving and calling activity*: one *receiving and calling activity* has to send and receive messages.
- The association *in* and *out* from *strategy message* to *strategy activity*: one *strategy activity* may have one input strategy message, one output strategy message or both.
- The association *in* and *out* from *operation message* to *operation activity*: one *operation activity* may have one input operation message, one output operation message or both.
- The association *in* and *out* from *support message* to *support activity*: one *support activity* may have one input support message, one output support message or both.
- The association *in* and *out* is from *process communication message* to *partner activity*. These links help identify messages, which are transferred between different types of activities (for example, between *strategy activity* and *operation activity*).

Associations between the Functional View and the Process View are used to identify supporting partner and mediator activities in a collaborative process:

- The association *represent* from *collaboration activity* to *collaboration strategy/operation/support activity*: one *collaboration strategy/operation/support activity* can refer to one *collaboration activity*. The *collaboration strategy/operation/support activity* defines the size, position and symbol of *collaboration activity* in the process model.

- The association *contain* from *collaboration activity* to *collaboration process*: one *collaborative activity* can have one sub-collaborative process.

Association *transferred by* from the information view to the process view:

- One *objective message* is transferred by one *objective message flow*.
- One *feedback message* is transferred by one *feedback message flow*.
- One *mean message* is transferred by one *mean message flow*.

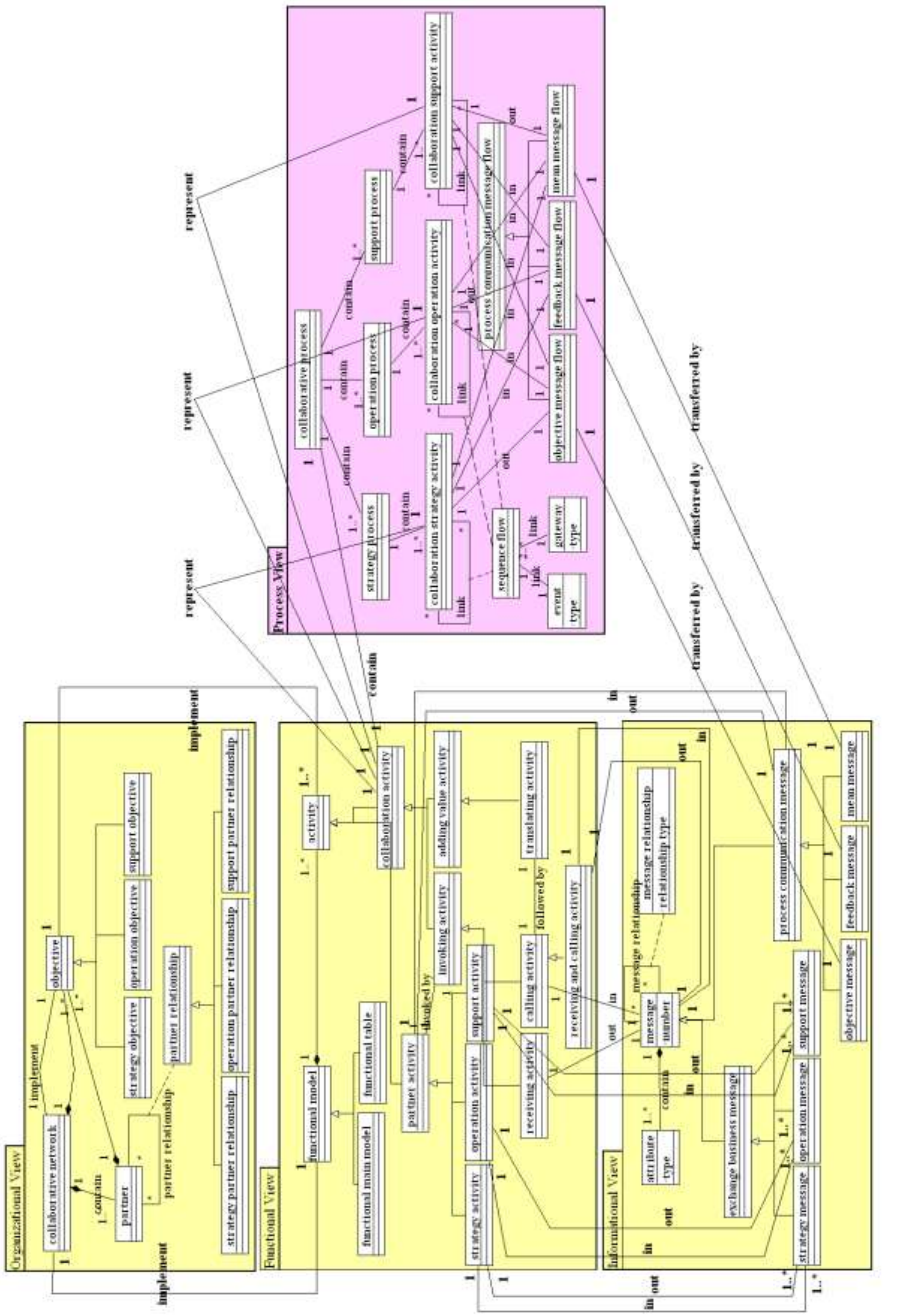


FIGURE 5 RELATIONS AMONG VIEWS

II. Transformation Rules in Collaborative Metamodel

To define initial rules formally, the rules have been defined with first-order logic. Because of the specialization of model transformation, first-order logic still needs to be expanded. The expanded rules are listed as follows:

- Class: X is collaborative network → collaborative network(X)
- Association: Y is association implement which is between collaborative network X1 and objective X2 → implement(Y) (collaborative network(X1), objective(X2))
- If-then-else: if (X) → then (Y), else if (X1) → then (Y1), else → then (Y2)
- A set of variables: from X1, X2, X3 to Xn→ X1 … Xn

The transformation rules are defined in six groups. As shown in Figure 6, the classes in white present the knowledge gathered by the collaborative network model, IDEF0-based functional model and IDEF1 model. The gray and black classes need to be deduced by the transformation rules.

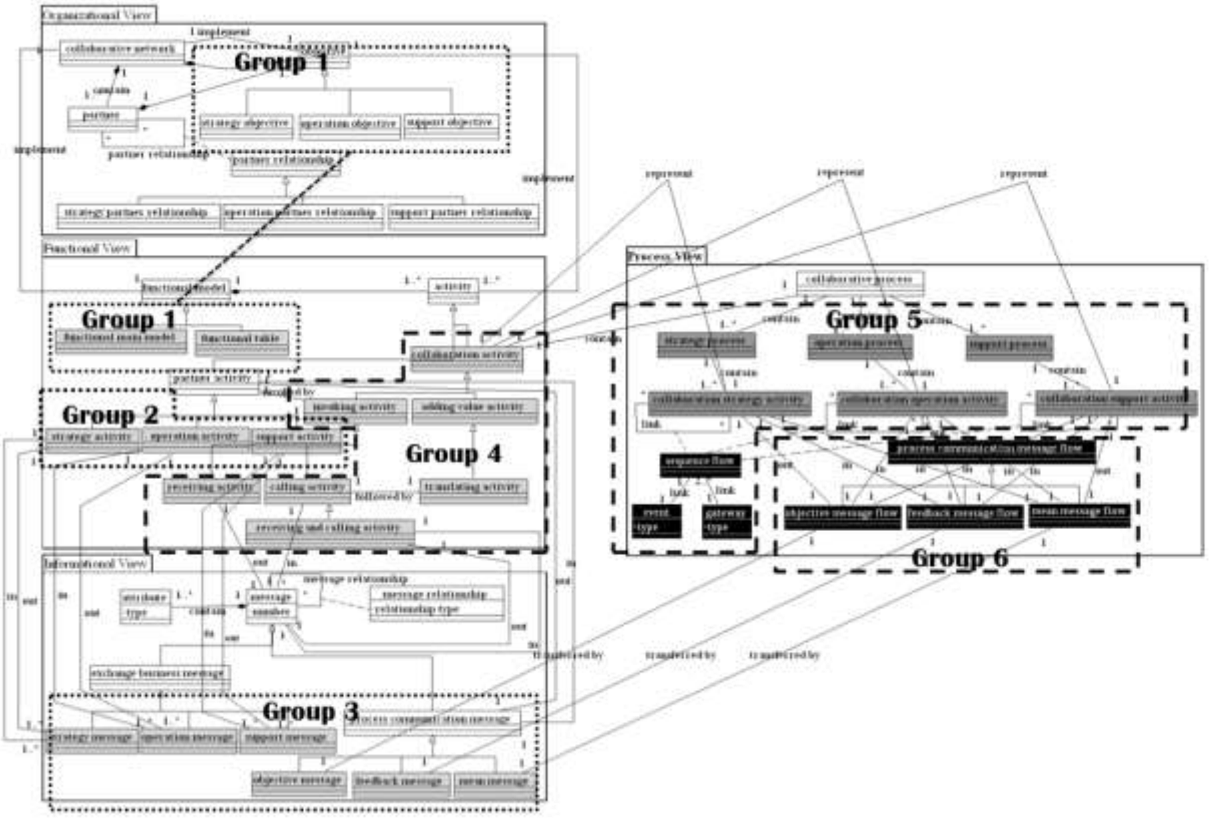


FIGURE 6 TRANSFORMATION GROUPS

Group 1: collaborative network →functional model. Group 1 Transformation Rules are used to initialize the functional model (equation no.1 and no.2). For any collaborative network with a sub-network, one functional main model is initialized. For any collaborative network without a sub-network, one functional table is initialized.

TABLE 1 TRANSFORMATION RULES GROUP 1

Group 1: collaborative network → functional model

∃ collaborative network (x) (∃ implement (collaborative network (x), objective (x_0))) ∧ ∄ contain (collaborative network (x), partner (x_1))	**(1)**
→∃ functional main model (y) ∧ ∃ implement (collaborative network (x), functional main model (y))	

∃ collaborative network (x) (∃ implement (collaborative network (x), objective (x_0))) ∧ ∃ contain (collaborative network (x), partner (x_1))	**(2)**
→∃ implement (collaborative network (x), functional main model (y))	

Group 2: partner activity →strategy/operation/support activity (equation no.3, no.4 and no.5). This group of transformation rules helps the classification of partner activities. If one partner activity links to a strategy objective, then the partner activity is a strategy activity. If one partner activity links to an operation objective, then the partner activity is an operation activity. If one partner activity links to a support objective, then the partner activity is a support activity.

TABLE 2 TRANSFORMATION RULES GROUP 2

Group 2: partner activity → strategy/operation/support activity	
∃ implement (strategy objective (x), partner activity (x_0))	**(3)**
→∃ implement (strategy objective (x), strategy activity (x_0))	

∃ implement (operation objective (x), partner activity (x_0))	**(4)**
→∃ implement (operation objective (x), operation activity (x_0))	

∃ implement (support objective (x), partner activity (x_0))	**(5)**
→∃ implement (support objective (x), support activity (x_0))	

Group 3: exchanged business message→strategy/operation/support message and process communication message→objective/feedback/mean message (equation no.6 to no.11). This group of transformation rules is defined to classify exchanged business messages and process communication messages. If one exchanged business message is an input or output message for a strategy activity, then the exchange business message is a strategy message. If one exchanged business message is an input or output message for an operation activity, then the exchanged business message is an operation message. If one exchanged business message is an input or output message for a support activity, then the exchanged business message is a support message. If a process communication message is an output message of a strategy activity and an input message of an operation or support activity, then the process communication message is an objective message. If a process communication message is an output message of an operation activity and an input message of an objective activity or support activity, then the process communication message is a feedback message. If a process communication message is an output message of a support activity and an input message of a strategy activity or a support activity, then the process communication message is a mean message.

Table 3 Transformation rules group 3

Group 3: partner activity → strategy/operation/support activity & partner activity → strategy/operation/support activity	
∃ exchanged business message (x) (∃ in (m)(strategy activity (x_1), exchanged business message (x)))	**(6)**
→∃ strategy message (x)	

∃ exchanged business message (x) (∃ in (m)(operation activity (x_1), exchanged business message (x)))	**(7)**
→∃ operation message (x)	

∃ exchanged business message (x) (∃ in (m)(support activity (x_1), exchanged business message (x)))	**(8)**
→∃ support message (x)	

∃ process communication message (x) (∃ out (m_1)(strategy activity (x_1), process communication message (x))) ∧ ∃ in (m_2)(operation activity (x_2), process communication message (x)) ∨ ∃ in (m_3)(support activity (x_3), process communication message (x))	**(9)**
→∃ objective message (x)	

∃ process communication message (x) (∃ out (m_1)(operation activity (x_1), process communication message (x))) ∧ ∃ in (m_2)(strategy activity (x_2), process communication message (x)) ∨ ∃ in (m_3)(support activity (x_3), process communication message (x))	**(10)**
→∃ feedback message (x)	

∃ process communication message (x) (∃ out (m_1)(support activity (x_1), process communication message (x))) ∧ ∃ in (m_2)(strategy activity (x_2), process communication message (x)) ∨ ∃ in (m_3)(operation activity (x_3), process communication message (x))	**(11)**
→∃ mean message (x)	

Group 4: partner activity→collaboration activity (equation no.12 to no.14). Transformation Rules of Group 4 are used to create collaboration activities in the functional view. If a partner activity has got one input message, then a calling activity and an association out to the message are created. An association invoked by from a partner activity to a calling activity is created. If a partner activity has one output message, then a receiving activity and an association in to the message are created. An association invoked by from a partner activity to a receiving activity is created. If a partner activity has both input message and output message, then a receiving

and calling activity and an association in/out to the messages are created. An association invoked by from a partner activity to a calling and receiving activity is created.

TABLE 4 TRANSFORMATION RULES GROUP 4

Group 4: partner activity → collaboration activity	
∃ partner activity (x)(∃ in(x_1)(partner activity (x), message (x_2)) ∧ ∃ out(x_3)(partner activity (x), message (x_4)))	**(12)**
→∃ receiving and calling activity (y)(∃ in(y_1)(receiving and calling activity (y), message (x_2)) ∧ ∃ out (y_2)(receiving and calling activity (y), message (x_4))) ∧ ∃ invoked by (z)(partner activity (x), receiving and calling activity (y))	

∃ partner activity (x)(∃ in(x_1)(partner activity (x), message (x_2)) ∧ ∄ out(x_3)(partner activity (x), message (x_4)))	**(13)**
→∃ calling activity (y)(∃ in(y_1)(calling activity (y), message (x_2)) ∧ ∄ out (y_2)(calling activity (y), message (x_4))) ∧ ∃ invoked by (z)(partner activity (x), calling activity (y))	

∃ partner activity (x)(∄ in(x_1)(partner activity (x), message (x_2)) ∧ ∃ out(x_3)(partner activity (x), message (x_4)))	**(14)**
→∃ receiving activity (y)(∄ in(y_1)(receiving activity (y), message (x_2)) ∧ ∃ out (y_2)(receiving activity (y), message (x_4))) ∧ ∃ invoked by (z)(partner activity (x), receiving activity (y))	

Group 5: collaborative activity →collaborative strategy/operation/support activity and sequence flow (equation no.15 to no.17). The Transformation Rules of Group 5 are used to create sequence flows in process view. This group of transformation rules is implemented by a breadth-first traversal algorithm graph [30]. A functional model can be analyzed as a graph. The function boxes can be seen as nodes. The input/output messages can be seen as arrows in a graph. The algorithm is summarized as follows: (i) If one pre-node has one post-node, then create a sequence flow between the pre-node and post-node (pre-node and post-node are partner activities, but they are linked to collaborative strategy/operation/support activity by a collaborative activity through an association represented by and an association invoked by, so the sequence flow is created between collaborative strategy/operation/ support activities). (ii) If one pre-node has several post-nodes, parallel gateway and sequence flows are created among the pre-node and post-nodes. (iii) If one post-node has several pre-nodes, sequence flows and parallel gateway are created among the pre-nodes and post-nodes. The first logic equation for this group is also defined. Here, we use strategy activity as an example to present the first logic equation.

TABLE 5 TRANSFORMATION RULES GROUP 5

Group 5: collaborative activity → collaborative strategy/operation/support activity	
∃ strategy activity (x_1), strategy activity (x_2) ∧ ∃ out (y1)(partner activity (x_1), message (x_3)) ∧ ∃ in (y2)(partner activity (x_2), message(x_4))	**(15)**
→∃ collaborative strategy activity (z_1) ∧	

∃ collaborative strategy activity (z_2) ∧ ∃ sequence flow (s)(collaborative strategy activity (z_1), collaborative strategy activity (z_2))	

∃ strategy activity (x_0), strategy activity (x_1)…strategy activity (x_n) ∧ ∃ out (y_0)(partner activity (x_0), message (m)) ∧ ∃ in (y_1)(partner activity (x_1), message (m))… ∧ ∃ in(y_n)(partner activity (x_n), message (m))	**(16)**
→∃ collaborative strategy activity (z_0) ∧ ∃ collaborative strategy activity (z_1)… ∧ ∃ collaborative strategy activity (z_n) ∧ ∃ gateway (g) ∧ ∃ sequence flow (s_0)(collaborative strategy activity (z_0), gateway (g)) ∧ ∃ sequence flow (s_1)(collaborative strategy activity (z_1), gateway (g))… ∧ ∃ sequence flow (s_n)(collaborative strategy activity (z_n), gateway (g))	

∃ strategy activity (x_0), strategy activity (x_1)…strategy activity (x_n) ∧ ∃ out (y_1)(partner activity (x_1), message (m))… ∧ ∃ out (y_n)(partner activity (x_n), message (m) ∧ ∃ in(y_0)(partner activity (x_0), message (m))	**(17)**
→∃ collaborative strategy activity (z_0) ∧ ∃ collaborative strategy activity (z_1)… ∧ ∃ collaborative strategy activity (z_n) ∧ ∃ gateway (g) ∧ ∃ sequence flow (s_0)(collaborative strategy activity (z_1), gateway (g))… ∧ ∃ sequence flow (s_1)(collaborative strategy activity (z_n), gateway (g)) ∧ ∃ sequence flow (s_n)(collaborative strategy activity (z_0), gateway (g))	

Group 6: collaborative activity →collaborative strategy/operation/support activity and sequence flow (equation no.18 to no.20). The Transformation Rules of Group 6 are used to create message flows in the process view. If there is a process communication message, which is in/out to partner activity, then process communication message flow, which is in/out to collaborative strategy/operation/support activity, is created.

TABLE 6 TRANSFORMATION RULES GROUP 6

Group 6: collaborative activity → collaborative strategy/operation/support activity and sequence flow	
(∃ objective message (x)(∃ out (m_1)(strategy activity (x_1), objective message (x)) ∧ ∃ in (m_2)(operation activity (x_2), objective message (x)) ∨ ∃ in (m_3)(support activity (x_3), objective message (x)))) ∧ ∃ x_1 (invoked by (x_1, invoking activity (X_1))) ∧ ∃ x_2 (invoked by (x_2, invoking activity (X_2))) ∧ ∃ x_3 (invoked by (x_3, invoking activity (X_3))) ∧ ∃ X_1 (represent (X_1, collaborative strategy activity (y_1))) ∧ ∃ X_2 (represent (X_2, collaborative operation activity (y_2))) ∧ ∃ X_3 (represent (X_3, collaborative support activity (y_3))) ∧	**(15)**

→∃ objective message flow (y) ∧ ∃ out (y_1, y) ∧ ∃ (in (y_2, y) ∨ in (y_3, y))	

(∃ feedback message (x)(∃ out (m_1)(operation activity (x_1), feedback message (x)) ∧ ∃ in (m_2)(strategy activity (x_2), feedback message (x)) ∨ ∃ in (m_3)(support activity (x_3), feedback message (x)))) ∧ ∃ x_1 (invoked by (x_1, invoking activity (X_1))) ∧ ∃ x_2 (invoked by (x_2, invoking activity (X_2))) ∧ ∃ x_3 (invoked by (x_3, invoking activity (X_3))) ∧ ∃ X_1 (represent (X_1, collaborative operation activity (y_1))) ∧ ∃ X_2 (represent (X_2, collaborative strategy activity (y_2))) ∧ ∃ X_3 (represent (X_3, collaborative support activity (y_3))) ∧	**(16)**
→∃ feedback message flow (y) ∧ ∃ out (y_1, y) ∧ ∃ (in (y_2, y) ∨ in (y_3, y))	

(∃ mean message (x)(∃ out (m_1)(support activity (x_1), mean message (x)) ∧ ∃ in (m_2)(operation activity (x_2), mean message (x)) ∨ ∃ in (m_3)(strategy activity (x_3), mean message (x)))) ∧ ∃ x_1 (invoked by (x_1, invoking activity (X_1))) ∧ ∃ x_2 (invoked by (x_2, invoking activity (X_2))) ∧ ∃ x_3 (invoked by (x_3, invoking activity (X_3))) ∧ ∃ X_1 (represent (X_1, collaborative support activity (y_1))) ∧ ∃ X_2 (represent (X_2, collaborative operation activity (y_2))) ∧ ∃ X_3 (represent (X_3, collaborative strategy activity (y_3))) ∧	**(17)**
→∃ mean message flow (y) ∧ ∃ out (y_1, y) ∧ ∃ (in (y_2, y) ∨ in (y_3, y))	

Annex B: Tool Design and Implementation

I. SaaS and GWT 168

I.1. GWT 169

I.2. GEasyTools 171

II. Detailed Design of Mediator Modeling 2ool 172

III. XML Files of Models 183

III.1. XML File of Collaborative Network Model....183

III.2. XML File of Function Model 185

III.3. XML File of Collaborative Process Cartography 187

IV. Codes of Ontology Tree 188

IV.1. OntologyTreePanel.java....188

IV.2. OntologyTreeGridDragPanel.java 191

V. References....194

I. SaaS and GWT

The term Software-as-a-Service (SaaS) entered the mainstream computing vocabulary a few years into this millennium. Initially, the term was used for various forms of service oriented computing (Gold et al., 2004), but is currently used for software that is provisioned over the internet and used usually with a web browser. The same naming convention is currently used also for other parts of the computing stack, e.g. Infrastructure-as-a-Service (IaaS) and Platform-as-a-Service (PaaS) (Schaffer, 2009). According to (Sääksjärvi et al., 2005), *"Software as a Service is time and location independent online access to a remotely managed server application, that permits concurrent utilization of the same application installation by a large number of independent users (customers), offers attractive payment logic compared to the customer value received, and makes a continuous flow of new and innovative software possible."* (Campbell-Kelly, 2009) defines that *"SaaS is different from traditional software licensing, which involves the buyer's purchasing a perpetual use license from the software publisher and then making additional investments for hardware, installation, and maintenance. In contrast, in the SaaS model, a user buy a subscription to the software and the software publisher (seller) runs and maintains the software on his own hardware. Users with current subscriptions can obtain access to the software using the Internet."* (Sun et al., 2010) says that *"Software as a Service (SaaS) is a software delivery model, which provides customers access to business functionality remotely (usually over the internet) as a service. The customer does not specially purchase a software license. The cost of the infrastructure, the right to use the software, and all hosting, maintenance and support services are all bundled into a single monthly or per-use charging."*

(Mäkilä et al., 2010) summarizes five distinct characteristics are typically associated with SaaS:

- Product is used through a web browser.
- Product is not tailor made for each customer.
- The product does not include software that needs to be installed at the customer's location.
- The product does not require special integration and installation work.
- The pricing of the product is based on actual usage of the software.

(Mäkilä et al., 2010) made a subjective judgment whether the analyzed SaaS product i) was pure SaaS, ii) had high SaaS characteristics or iii) was not SaaS service at all. In addition, the researchers took notes about the nature of the found SaaS product. Table V-1 cross-tabulate the categories of the firms meeting different numbers of SaaS criteria with the business model classification developed with cluster analysis of the revenue share data.

TABLE V - 1 CROSS-TABULATIO OF BUSINESS MODELS AND SAAS CRITERIA (MÄKILÄ ET AL., 2010)

	Number of SaaS criteria filled				
Business Model	Little or no SaaS characteristics	Web based solutions	High SaaS characteristics	Pure SaaS	**Total**
Software product	26.9%	35.5%	17.2%	10.0%	29.5%
Deployment project	11.9%	14.9%	0.0%	0.0%	11.5%
Development service	34.3%	21.5%	20.7%	20.0%	25.1%
ASP and SaaS	4.5%	11.6%	37.9%	70.0%	15.4%
Not software	11.9%	9.9%	17.2%	0.0%	11.0%
Content and ads	0.0%	2.5%	3.4%	0.0%	2.8%
Software consulting	6.0%	2.5%	3.4%	0.0%	3.5%
Hardware	4.5%	1.7%	0.0%	0.0%	2.2%
Total	100.0%	100.0%	100.0%	100.0%	100.0%

(Kang et al., 2010) discusses several cases of current large SaaS vendors, which have their own characteristics of functionalities of SaaS service, and derive the essential common functions to build successful SaaS service.

Amazon provides SaaS service in terms of Amazon Web Services. It mainly focus on providing computing resources to users rather than a separated web-based application so that it gives customers various business application on their service infrastructure that is based on cloud computing paradigm. The goal of Amazon's SaaS service can be summarized by these terms: Cost-effective, Dependable, Flexible, and Comprehensive. In order to achieve the goals of Amazon Web Services, they settled various service types from business infrastructure to Web search and on-demand workforce. A distinct characteristic of Amazon Web Service is to give the opportunity to ISVs (Independent Software Provider), where the main targeted user of other vendors is end-users themselves who use the applications on the SaaS framework.

The SaaS service of Salesforce.com can be summarized as Force.com platform. It is a multi-tenant on-demand business platform, which consists of service component and process. The biggest difference of the strategic directions of salesforce.com is to be a solution provider to enterprise with multi-tenant support platform. The whole service process of Force.com service aimed to achieving the next level of current SaaS, which is called PaaS. It means that PaaS should be able to provide the tools for developing on-demand applications easily on the Web-based infrastructure as well as using and distributing the solutions.

The main target users of Microsoft are customers who have used Microsoft's package software such as Windows and Microsoft Office. They try to add the service strategy based on the web to existing software in comparison with the other vendors who provide their service through network by using Web browser. This strategy is called Software + Service. The strategic directions of Microsoft are categorized into four parts: Unified Experience, Server and Cloud, Tightly Coupled System, and Multiple Business Model. By adapting these strategies, Microsoft tries to get flexibility and availability on service process from building service with software-to-software distribution.

Google provide SaaS service as the set of Google application named GoogleApps. It provides communicate and connect service through Web browser, and they are inter-linked by collaboration process of Google Application such as Google Docs and Google Sites. In order to use their infrastructure and ability to search on the Web, Google tries to organize user's service via Web application development environment named Google Apps Engine. Most of Google SaaS service is supposed in the form of distributed APIs to guarantee effectiveness, flexibility, and easiness of application usage.

TABLE V - 2 SUMMARIZATION OF SAAS SERVICE VENDORS (KANG ET AL., 2010)

Vendor	Service Description	Business Model	Origin	Strategy
Amazon	Computing Resource Providing	Amazon Web Service	Web Service	Service Infrastructure
Salesforce	Web-based CRM	Force.com	Web service/CRM	Platform as a Service
Microsoft	Personal/Office Tools	Microsoft Office Live	Package Software	Software+Service
Google	Web Office Tools	Google Apps	Web-based Service	Service on the Web

I.1. GWT

Google Web Toolkit (GWT)[1] (Dwyer, 2008; Gupta, 2008a) is a development toolkit for building and optimizing complex browser-based applications. Its goal is to enable productive development of high-performance web applications without the developer having to be an expert in browser quirks,

[1] Google web toolkit: https://developers.google.com/web-toolkit/?hl=zh-CN

XMLHttpRequest, and JavaScript. It's open source, completely free, and used by thousands of developers around the world. Using GWT, developers can develop and debug Ajax applications (Vohra, 2008) in the Java language using the Java development tools of their choice. When the application is deployed, the GWT cross-compiler translates the Java application to standalone JavaScript files that are optionally obfuscated and deeply optimized. GWT applications can be run in two modes. First, development mode (formerly Hosted mode): The application is run as Java bytecode within the Java Virtual Machine (JVM). This mode is typically used for development, supporting hot swapping of code and debugging. Second, production mode (formerly Web mode): The application is run as pure JavaScript and HTML, compiled from the Java source. This mode is typically used for deployment.

According to (Smeets et al., 2008b), GWT can roughly be divided into four main parts (Gupta, 2008b), although the last one especially has many separate subdivisions:

- GWT Java-to-JavaScript Compiler: translates the Java programming language to the JavaScript programming language. This is the heart of GWT, and its most impressive part. The compiler will make sure that all code that you write is eventually translated into JavaScript.
- GWT Development Mode: allows the developer to run and execute GWT applications in development mode (the app runs as Java in the JVM without compiling to JavaScript). Prior to 2.0, GWT hosted mode provided a special-purpose "hosted browser" to debug your GWT code. In 2.0, the web page being debugged is viewed within a regular browser. Development mode is supported through the use of a native-code plugin called the Google Web Toolkit Developer Plugin for many popular browsers.
- JRE emulation library, JavaScript implementations of the commonly used classes in the Java standard class library (such as most of the java.lang package classes and a subset of the java.util package classes). GWT needs to compile the code you write in Java into JavaScript. In order for this to work, GWT has to provide an emulation of the core Java constructs and classes so they can be translated to code that works in JavaScript.
- GWT Web UI class library (Smeets et al., 2008a), a set of custom interfaces and classes for creating widgets. This part of GWT consists of many subparts. This makes up almost the entire code base provided by GWT, including the actual UI components, RPC support, history management, and much more.

GWT version 1.0 RC 1 (build 1.0.20) was released on May 16, 2006. The most recent stable version is GWT version 2.4. The last version is GWT version 2.5 RC. In version 2.5 RC, the following new features are added:

- Super Dev Mode[1] (experimental) is an experimental replacement for Development Mode.
- Elemental (experimental) is an experimental new library for fast, lightweight, and "to the metal" web programming in GWT. It's intended for developers who are comfortable working with the browser API's that JavaScript programmers use.
- New compiler optimizations[2] can optionally use the Closure compiler to provide additional JavaScript optimizations. The Closure compiler has a collection of Javascript optimizations that can benefit code size, including a graph-coloring-based variable allocator, comprehensive JavaScript function and variable inlining, cross-module code motion, statement fusing, name shadowing and many more.
- Updated ARIA support, they added a new accessibility ARIA library that has a full coverage of the W3C ARIA standard. This makes it easier to correctly set ARIA roles, states, and properties on DOM elements. For more details have a look at the updated GWT accessibility documentation.
- UIBinder Enhancements, GWT 2.5 adds extensions to UiBinder that allow it to support Cell rendering and event handling. In particular, this design enables UiBinder to generate a UiRenderer implementation to assist with rendering SafeHtml, and dispatching events to methods specified by @UiHandler tags.

Several open-source plugins are available for making GWT development easier with other IDEs. E.g., GWT4NB for NetBeans, Cypal Studio for GWT, Eclipse and JDeveloper etc. The Google Plugin for Eclipse handles most GWT related tasks in the IDE, including creating projects, invoking the GWT compiler, creating

[1] Supper DevModel: https://developers.google.com/web-toolkit/articles/superdevmode

[2] Elemental: https://developers.google.com/web-toolkit/articles/elemental

GWT launch configurations, validations, syntax highlighting, etc. GWT Maven plugin supports GWT compiler execution, generation of GWT internationalization, running tests, running or debugging DevMode from Maven Dev Mode, integration with Eclipse, etc. There is also GWT-based framework (Slender, 2009) that allows user to utilize the comprehensive widget library for the application UI (e.g., SmartGWT[1], GXT[2], etc.). In the development of Mediator Modeling 2ool, the GWT maven plugin (GWT SDK 2.0.4) and GXT are used.

I.2. GEasyTools

GEasyTools[3] is a set of libraries that aims to build rich interactive applications more easily with GWT. It is a GMF (Plante, 2006) like tool. It provides generative components and runtime infrastructures for developing graphical editors based on GWT. For example, the palette of modeling elements, the canvas and panel to draw the graphs of modeling elements, the property views of elements, the validation of models and the import and export of models.

GEasyTools actually contains the following main libraries (the details of libraries are summarized in Table V-3): i) GeasyUI deals with user interactions (drag & drop, resize handling, selection handling), ii) GeasySVG: the cross-browser SVG[4]/VML[5] library, iii) GeasyGraph deals with graph problematic: path-finding etc., iv) GeasyModelManager handles models on client side for undo/redo, methods observers and UIBinding and v) GeasyDiagramEditor: the project for editing diagrams based on OMG Diagram Definition standard.

TABLE V - 3 LIBRARIES OF GEASYTOOL

Library Name	V.	P.	Functional Package Name	Class Descriptions
Diagram-common	1.0	2	Diagramcommon.layout	Font, dimension, point and bounds
Diagram-interchange	1.0	2	Interchange.impl	Edge, diagram, label, node and sharp
Geasy-diagram-editor	1.0	14	Impl.events.loader	Editor view load events concern state change, element loading, etc.
			Impl.event.validation	Editor view validation events concern start, success and warning
			Impl.modeleditor	Editor view and editor view changed event
			Impl.validation	Notification of validation rule and Registration of validation rule
Geasy-graph	1.0	4	Geasygraph.impl	Graph, node and path finder
Geasy-svg	1.0	5	Geasysvg.core.impl	Canvas, circle, group, path, text, SVG document, SVG element and JSNI
			Geasysvg.ext.impl	Linear path and point
Geasy-ui	1.0	33	Geasyui.impl.connectable	Connector, connector point, magnet and events of connection
			Geasyui.impl.contextualmenu	Contextual menu and drag proxy
			Geasyui.impl.core	UIElement, UIPanel and events of remove, add and resize element
			Geasyui.impl.draggable	Draggable proxy, move/start/stop events and drop indicator
			Geasyui.impl.droppable	Drop handler, out and over events and drop refused event
			Geasyui.impl.palette	Palette and palette group

[1] SmartGWT: http://code.google.com/p/smartgwt/
[2] GXT: http://www.sencha.com/products/gxt
[3] GeasyTools: http://research.petalslink.org/display/geasytools/GEasyTools+Overview
[4] SVG: Scalable Vector Graphics
[5] VML: Vector Markup Language

			Geasyui.impl.resizable	Element resized handler and resize start/stop event
			Geasyui.impl.selectable	Element selected handler and select/unselect event
Geasy-widgets	1.0	22	Widgets.ext.impl.file	File, folder and events concern file selected and loaded
			Widgets.ext.impl.notification	Message type, notification bubble and configuration
Model-manager	1.0	4	Modelmanager.client	Model manager and undo/redo session
			Modelmanager.uibinder	UI binding manager

V.: version P.: package no.

II. Detailed Design of Mediator Modeling 2ool

TABLE 1 CLASSES IN ../CLIENT/

Package Name: ../client/	
Class	Class Description
Application.java	Entry point class of GWT
Main Operation	Description
Public void onModuleLoad (){}	Load all the interface classes: palette window, property window, canvas window and so on
Public EditorView getEditorViewPanel (){}	Create and return the instance of property window. It is invoked by onModuleLoad()
Class	Class Description
MyFrameLayout.java	Main interface class, it creates all the other interfaces classes
Main Operation	Description
Public MyFrameLayout (){}	Main operation of this class to load all interfaces
Public static MyFrameLayout getInstance (){}	Return the instance of MyFrameLayout
Protected void onWindowResize (int width, int height){}	Deal with the resized event of window
Public void setPalette (MyDependencyPalette palette){}	Set palette window
Public void setGraph (MyGraphView graph){}	Set canvas window
Public void setEditorView (EditorView editorView){}	Set property window
Public ContentPanel getSouthPanel (){}	Return the panel of console
Public TabPanel getCenterPanel (){}	Return the panel of canvas
Public WestPanel getWestPanel (){}	Return the panel of palette and file
Public EastPanel getEastPanel (){}	Return the panel of property and ontology
Public void setWestPanel (WestPanel westPanel){}	Set the palette and file window
Public void setEastPanel (EastPanel eastPanel){}	Set the property and ontology window

TABLE 2 CLASSES IN ../CLIENT/CUSTOMICONS

Package Name: ../client/customicons	
Class	Class Description
Resources.java	Creates ExampleIcons.class
Class	Class Description
ExampleIcons.java	Import icons in the folder to java class
Main Operation	Description
AbstractImagePrototype center_panel_tab_f (){}	Define icon of tab of function model
…	…
AbstractImagePrototype validate_organization (){}	Define icon of validation button of collaborative network model

TABLE 3 CLASSES IN ../CLIENT/UI/PANEL

Package Name: ../client/ui/panel	
Class	Class Description
NorthMenuBar.java	Define interface of menu bar
Main Operation	Description
Public NorthMenuBar (){}	Create menu items and mouse clicked event of items
Class	Class Description
NorthToolBar.java	Creates north tool bar
Main Operation	Description
Public NorthToolBar (){}	Create tool bar buttons and clicked event
Class	Class Description
WestPanel.java	Creates palette and file windows in the west panel
Main Operation	Description
Public WestPanel (){}	Main operation of the class, add palette and file windows
Public TabItem getTabFile (){}	Return file window
Public void setTabFile (TabItem tabFile){}	Set file window
Public TabItem getTabPalette (){}	Return palette window
Public void setTabPalette (TabItem tabPalette){}	Set palette window
Public FileTreePanel getFileTree (){}	Return file tree
Public void setFileTree (FileTreePanel fileTree){}	Set file tree
Class	Class Description
EastPanel.java	Creates east panel, which includes property and ontology window
Main Operation	Description
Public EastPanel (){}	Main operation of this class, creates property and ontology window
Public TabItem getTabProperty (){}	Return property window
Public void setTabProperty (TabItem tabProperty){}	Set property window
Public TabItem getTabOntology (){}	Return ontology window
Public void setTabOntology (TabItem tabOntology){}	Set ontology window
Public OntologyTreePanel getTreePanel (){}	Return ontology tree

Public void setTreePanel (OntologyTreePanel treePanel){}	Set ontology tree
Class	Class Description
CenterPanel.java	Creates canvas window
Main Operation	Description
Public CenterPanel (){}	Create center tab panel, add canvas
Class	Class Description
GraphTabItem.java	Creates tab item with canvas
Main Operation	Description
Public GraphTabItem (String name, String type, MyGraphView graph){}	Create tab item with graph
Public MyGraphView getGraph (){}	Return graph of the tab item
Public void setGraph (MyGraphView graph){}	Set graph of the tab item
Public MyGraphView getParentGraph (){}	Return the parent graph of the item
Public void setParentGraph (MyGraphView parentGraph){}	Set the parent graph of the item
Public FileTreeFileNode getFile (){}	Get the file, which is opened in the item
Public void setFile (FileTreeFileNode file){}	Set the file, which is opened in the item
Class	Class Description
FileNewWindow.java	Creates the interface for creating new project
Main Operation	Description
Public FileNewWindow (){}	Create the main interface
Public String getName (){}	Set the file name
Public void setName (String name){}	Get the file name

Table 4 Classes in ../client/ui/tree

Package Name: ../client/ui/tree	
Class	Class Description
FileTreeFileNode.java	Create file node in the file tree
Class	Class Description
FileTreeFolderNode.java	Create folder node in the file tree
Class	Class Description
FileTreePanel.java	Define file tree panel by using files on the server
Main Operation	Description
Public FileTreePanel (){}	Main operation of the class, invokes other operations
Public FileTreeFileNode getSelectedNode (){}	Return the selected node in the file tree
Public void setSelectedNode (FileTreeFileNode selectedNode){}	Set the selected node for the file tree
Public void refreshTreePanel (){}	Refresh or re-draw the file tree
Public void drawTreePanel (){}	Read fills on the server side and draw the basics of the tree
Class	Class Description
FileUploadWindow.java	Create the window to upload an new file
Main Operation	Description
Public FileUploadWindow (){}	Create the interface to upload a file
Public Window getWindow (){}	Return the window
Public void setWindow (Window window){}	Set the window
Public FileUploadField getFile (){}	Get uploaded file

Public void setFile (FileUploadField file){}	Set uploaded file
Class	Class Description
OntologyTreeNodeElement.java	Create ontology node
Class	Class Description
OntologyTreeNode.java	Create ontology root node
Class	Class Description
OntologyTreePanel.java	Creates ontology tree panel
Main Operation	Description
Public OntologyTreePanel (){}	Create tab item with graph
Public void setOntology (String ontologyXML){}	Return graph of the tab item
Public void createTreeModelData (){}	Set graph of the tab item
Public void createTreeGridPanel (){}	Return the parent graph of the item
Public ContentPanel getTreeGridPanel (){}	Set the parent graph of the item
Public void setTreeGridPanel (ContentPanel treeGridPanel){}	Get the file, which is opened in the item
Public OntologyTreeNode getRootNode (){}	Get the file, which is opened in the item
Public void setRootNode (OntologyTreeNode rootNode){}	Set the file, which is opened in the item
Class	Class Description
OntologyTreeGridDragPanel.java	Creates the interface for selecting same as/near by instance (in the ontology) for objective, function and message
Main Operation	Description
Public OntologyTreeGridDragPanel (){}	Main operation invokes others
Public void CreateTreeGragSourcePanel (){}	Create the source panel to represent instances of collaborative ontology
Public void CreateTreeGragTargetPanel (){}	Create same as and near by panel
Public void setTree (TreeGrid<ModelData> tree){}	Set the ontology tree
Public ColumnModel getColumnModel (){}	Return tree grid column
Public OntologyTreeNodeElement getSameasElement (){}	Return same as node
Public void setSameasElement (OntologyTreeNodeElement sameasElement){}	Set same as node by using the node itself
Public setSameasElement (String id){}	Set same as node by using the id of the node
Public OntologyTreeNodeElement getNearbyElement (){}	Return near by node
Public void setNearbyElement (OntologyTreeNodeElement nearbyElement){}	Set near by node by using the node itself
Public void setNearbyElement (String id){}	Set near by node by using the id of the node

TABLE 5 CLASSES IN ../CLIENT/ELEMENTS

Package Name: ../client/elements	
Class	Class Description
OrganizationNetwork.java	Create symbols and dragging, dropping and connecting ability of network element in collaborative network model
OrganizationNetworkSyntax.java	Add binding syntax for network element
OrganizationObjective.java	Create symbols and dragging, dropping and connecting ability of objective element in collaborative network model
OrganizationObjectiveSyntax.java	Add binding syntax for objective element
OrganizationOperationObjective.java	Create symbols and dragging, dropping and connecting ability of operation objective element in collaborative network model
OrganizationOperationObjectiveSyntax.java	Add binding syntax for operation objective element
OrganizationStrategyObjective.java	Create symbols and dragging, dropping and connecting ability of strategy objective element in collaborative network model
OrganizationStrategyObjectiveSyntax.java	Add binding syntax for strategy objective element
OrganizationSupportObjective.java	Create symbols and dragging, dropping and connecting ability of support objective element in collaborative network model
OrganizationSupportObjectiveSyntax.java	Add binding syntax for support objective element
OrganizationPartner.java	Create symbols and dragging, dropping and connecting ability of partner element in collaborative network model
OrganizationPartnerSyntax.java	Add binding syntax for partner element
FunctionFunction.java	Create symbols and dragging, dropping and connecting ability of function element in function model
FunctionFunctionSyntax.java	Add binding syntax for function element
InformationInputMessage.java	Create symbols and dragging, dropping and connecting ability of input message element in function model
InformationInputMessageSyntax.java	Add binding syntax for input message element
InformationOutputMessage.java	Create symbols and dragging, dropping and connecting ability of output message element in function model
InformationOutputMessageSyntax.java	Add binding syntax for output message element
OthersNote.java	Create symbols and dragging, dropping and connecting ability of note element
OthersNoteSyntax.java	Add binding syntax for note element
ICoreElement.java	Abstract class for modeling element
MyAbstractConnectableUIElement.java	Base class for all above classes

TABLE 6 CLASSES IN ../CLIENT/CONNECTOR

Package Name: ../client/connector	
Class	Class Description
MyAbstractConnectorElement.java	Define basic operations and variables of connector
MyConnectorEndElement.java	End point of connector
MyConnectorPointElement.java	Normal points of connector
MyConnectorStartElement.java	Start point of connector
MyMagnetElement.java	Magnet point on the connectable element
ObjectiveRelationship.java	Connector for objective relationship
ObjectiveRelationshipSyntax.java	Add binding syntax for objective relationship
PartnerRelationship.java	Connector for partner relationship
PartnerRelationshipSyntax.java	Add binding syntax for partner relationship
InOutFlow.java	Connector for in/out flow of input/output message
InOutFlowSyntax.java	Add binding syntax for in/out flow
NoteLink.java	Connector for note link
NoteLinkSyntax.java	Add binding syntax for note link

TABLE 7 CLASSES IN ../CLIENT/EDITORMODEL

Package Name: ../client/editormodel	
Class	Class Description
OrganizationEditorModel.java	Define property view of collaborative network model
OrganizationNetworkEditorModel.java	Define property view of collaborative network element
OrganizationOperationObjectiveEditorModel.java	Define property view of operation objective element
OrganizationPartnerEditorModel.java	Define property view of partner element
OrganizationStrategyObjectiveEditorModel.java	Define property view of strategy objective element
OrganizationSupportObjectiveEditorModel.java	Define property view of support objective element
PartnerRelationshipEditorModel.java	Define property view of partner relationship element
ObjectiveRelationshipEditorModel.java	Define property view of objective relationship element
FunctionFunctionEditorModel.java	Define property view of function element
InformationInputMessageEditorModel.java	Define property view of input message element
InformationOutputMessageEditorModel.java	Define property view of output message element

TABLE 8 CLASSES IN ../CLIENT/TEMPLATE

Package Name: ../client/template	
Class	Class Description
ModelTemplate.java	Define the interface of property view
Main Operation	Description
Public IDiagramView getDiagram (){}	Return the diagram, which linked to the template
Public void setDiagram (IDiagramView diagram){}	Set the diagram, which linked to the template
Public Widget getTemplate (){}	Return the drawn interface of property view

TABLE 9 CLASSES IN ../CLIENT/SUBMENU

Package Name: ../client/submenu	
Class	Class Description
ISubModelMenu.java	Interface of SubModelMenu.java
Class	Class Description
IHasSubModelMenu.java	Interface to implement for connectable element.java
Class	Class Description
SubModelMenu.java	Define sub menu interfaces and mouse event
Main Operation	Description
Public SubModelMenu (){}	Set menu size, menu icons and define mouse event
Public void init (){}	Implement mouse event and set timer
Class	Class Description
ContextualMenuHandler.java	Deal with events
Main Operation	Description
Public ContextualMenuHandler (){}	Create class
Public IUIPanel getUIPanel (){}	Return diagram
Public void onProxyDragStart (IProxyDragStartEvent event){}	Deal with drag start event
Public void onProxyDragStop (IProxyDragStopEvent event){}	Deal with drag stop event
Public void onProxyDragMove (IProxyDragMoveEvent event){}	Deal with drag move event
Public void onProxyAcceptedBeforeDrop (IProxyAcceptedBeforeDropEvent event){}	Deal with drop accepted event (before drop)
Public void onProxyRefusedBeforeDrop (IProxyRefusedBeforeDropEvent event){}	Deal with drop refused event (before drop)
Public void onProxyAcceptedAfterDrop (IProxyAcceptedAfterDropEvent event){}	Deal with drop accepted event (after drop)
Public void onProxyRefusedAfterDrop (IProxyRefusedAfterDropEvent event){}	Deal with drop refused event (after drop)

TABLE 10 CLASSES IN ../CLIENT/FILEOPERATION

Package Name: ../client/fileoperation	
Class	Class Description
ImportFunctionXMLFile.java	Import function model (.fun)
Main Operation	Description
Public void parseFunctionXMLFile (){}	Load XML file and parse
Public MyGraphView drawGraph (Element graphElement){}	Draw all the modeling elements
Public MyGraphView getMainGraph (){}	Return diagram
Public void setMainGraph (MyGraphView mainGraph){}	Set diagram
Class	Class Description
ImportOrganizationXMLFile.java	Import collaborative network model (.org)
Main Operation	Description
Public void parseOrganizationXMLFile (){}	Load XML file and parse

Public MyGraphView drawGraph (Element graphElement){}	Draw all the modeling elements
Public MyGraphView getMainGraph (){}	Return diagram
Public void setMainGraph (MyGraphView mainGraph){}	Set diagram
Class	Class Description
ImportMatchingResultXMLFile.java	Import business service selection results (.xml)
Main Operation	Description
Public void parseMatchingResultXMLFile (){}	Load XML file and parse
Public MyGraphView drawGraph (Element graphElement){}	Draw all the modeling elements
Public MyGraphView getMainGraph (){}	Return diagram
Public void setMainGraph (MyGraphView mainGraph){}	Set diagram
Class	Class Description
ExportFunctionXMLFile.java	Export function model (.fun)
Main Operation	Description
Public ExportFunctionXMLFile (MyGraphView graph, String type){}	Get diagram and create empty XML document
Public void createDomTree (MyGraphyView graph){}	Create XML tree for XML document
Protected Element createElement (IUIElement element, String type){}	Add modeling element to tree
Public Element createConnector (IUIElement element, String type){}	Add connector element to tree
Public setPointAttribute (IConnectorPoint pont, Element pointElement){}	Set points (x, y) for connector
Public Document getDocument (){}	Return XML document
Public void setDocument (){}	Set XML document
Class	Class Description
ExportOrganizationXMLFile.java	Export collaborative network model (.org)
Main Operation	Description
Public ExportOrganizationXMLFile (MyGraphView graph, String type){}	Get diagram and create empty XML document
Public void createDomTree (MyGraphyView graph){}	Create XML tree for XML document
Protected Element createElement (IUIElement element, String type){}	Add modeling element to tree
Public Element createConnector (IUIElement element, String type){}	Add connector element to tree
Public setPointAttribute (IConnectorPoint pont, Element pointElement){}	Set points (x, y) for connector
Public Document getDocument (){}	Return XML document
Public void setDocument (){}	Set XML document

TABLE 11 CLASSES IN ../CLIENT/PALETTE

Package Name: ../client/palette	
Class	Class Description
FunctionFunctionDragProxy.java	Drag function symbol from palette to canvas
InformationInputMessageDragProxy.java	Drag input message symbol from palette to canvas
InformationOutputMessageDragProxy.java	Drag output message symbol from palette to canvas
InOutFlowDragProxy.java	Drag in/out flow symbol from palette to canvas
NoteLinkDragProxy.java	Drag note link symbol from palette to canvas
ObjectiveRelationshipDragProxy.java	Drag objective relationship symbol from palette to canvas
OrganizationNetworkDragProxy.java	Drag network symbol from palette to canvas
OrganizationObjectiveDragProxy.java	Drag objective symbol from palette to canvas
OrganizationOperationObjectiveDragProxy.java	Drag operation objective symbol from palette to canvas
OrganizationStrategyObjectiveDragProxy.java	Drag strategy objective symbol from palette to canvas
OrganizationSupportObjectiveDragProxy.java	Drag support objective symbol from palette to canvas
OthersNoteDragProxy.java	Drag others note symbol from palette to canvas
PartnerRelationshipDragProxy.java	Drag partner relationship symbol from palette to canvas
MyDependencyPalette.java	Create interface for palette

TABLE 12 CLASSES IN ../CLIENT/VIEW

Package Name: ../client/view	
Class	Class Description
MyGraphView.java	Create canvas
Main Operation	Description
Public MyGraphView (int width, int height){}	Create canvas by size
Public IDiagram getDiagram (){}	Return canvas
Public IEditorModel getEditorModel (){}	Return property view
Public IDiagramElementGraphicFactory getElementFactory (){}	Return factory
Public HashSet<Class<? extends IDraggableElement>> getAcceptedTypes (){}	Set drop accepted elements
Public IPalette getPalette (){}	Return palette
Class	Class Description
MyGraphFactory.java	Create factory, which creates all kinds of modeling elements
Main Operation	Description
Public MyGraphFactory (GraphDiagramView graph){}	Create class by graph
Public MyGraphFactory (MyGraphView view){}	Create class by view
Public IUIElement getElement (IHasDragProxy draggableProxyData){}	Create all kinds of modeling elements
Public IDiagramElementView getElementByDiagramElementModel (IDiagramElement diagramElement){}	Return element view

TABLE 13 CLASSES IN ../CLIENT/SERVICE

Package Name: ../client/service	
Class	Class Description
FileService.java	Interface class for server
Main Operation	Description
Public String getOntology (String name);	Return ontology file
Public List<FileTreeFileNode> getFolderChildren (FileTreeFileNode folder);	Return files in folder
Public String saveFileOnSever (String file,String path,String name, boolean creation) throws IOException	Save file on server
Public String loadFileOnSever (String path);	Load file from server
Public String matchObjectiveFunction (String projectName);	Launch mapping service
Public String transferMainProcess (String projectName);	Launch transformation of collaborative process cartography
Public String transferCollabroativeProcess (String projectName);	Launch transformation of collaborative processes
Class	Class Description
FileServiceAsync.java	Interface for client side
Main Operation	Description
Public void getOntology (String name, AsyncCallback<String> asyncCallback);	Return ontology file
Public void getFolderChildren (FileTreeFileNode model,AsyncCallback<List<FileTreeFileNode>> children);	Return files in folder
Public void saveFileOnSever (String file,String path,String name, boolean creation, AsyncCallback<String> asyncCallback);	Save file on server
Public void loadFileOnSever (String path, AsyncCallback<String> asyncCallback);	Load file from server
Public void matchObjectiveFunction (String projectName, AsyncCallback<String> asyncCallback);	Launch mapping service
Public void transferMainProcess (String projectName, AsyncCallback<String> asyncCallback);	Launch transformation of collaborative process cartography
Public void transferCollabroativeProcess (String projectName, AsyncCallback<String> asyncCallback);	Launch transformation of collaborative processes

TABLE 14 CLASSES IN ../SERVER

Package Name: ../server	
Class	Class Description
FileServiceImpl.java	Interface class for server
Main Operation	Description
Public FileServiceImpl (){}	Return ontology file
Public String getOntology (String name) {}	Return ontology file
Public List<FileTreeFileNode> getFolderChildren (FileTreeFileNode folder) {}	Return files in folder
Public String saveFileOnSever (String file,String	Save file on server

path,String name, boolean creation) throws IOException{}	
Public String loadFileOnSever (String path) {}	Load file from server
Public String matchObjectiveFunction (String projectName) {}	Launch mapping service
Public String transferMainProcess (String projectName) {}	Launch transformation of collaborative process cartography
Public String transferCollabroativeProcess (String projectName) {}	Launch transformation of collaborative processes
Class	Class Description
XMLFile.java	Parse XML file and add basic operations
Main Operation	Description
Public XMLFile (String path){}	Get file and parse
Public Node getNodeByID (String tag,String name, String value){}	Find node by id
Public List<Node> getNodesByAttribute (String tag, String name, String value){}	Find node by attribute
Public void removeChidren (Node graph){}	Remove children nodes
Class	Class Description
ObjectiveFunctionMatcher.java	Selected functions to achieve objectives
Main Operation	Description
Public ObjectiveFunctionMatcher (String projectName){}	Create class for probject
Public void CleanObjectiveModel (){}	Clean useless nodes in collaborative network model
Public void MoveFunction (String tagname){}	Find functions and add to objective
Public void OutputNewFunctionModel (){}	Write file on server (.xml)
Public void addFunctionToSameObjective (Node function, String value){}	Add function for same as objective
Public void addFunctionToNearObjective (Node function, String value){}	Add function for near by objective
Class	Class Description
MainCollaborativeProcess.java	Transfer collaborative process cartography
Main Operation	Description
Public MainCollaborativeProcess (String projectName){}	Create the class for the project
Public void AddMessageFlow (){}	Add message flow to BPMN
Public void SearchMainFunction (){}	Find main functions in collaborative network model
Public void addStrategyFunction (Element graph){}	Add strategy task to strategy pool
Public void addOperationFunction (Element graph){}	Add operation task to operation pool
Public void addSupportFunction (Element graph){}	Add support task to support pool
Public void addGeneralFunction (Element graph){}	Add general task to general pool
Public void OutputMainProcessModel (){}	Write BPMN file (.bpmn)
Class	Class Description
SubCollaborativeProcess.java	Transfer collaborative processes
Main Operation	Description
Public SubCollaborativeProcess (String projectName){}	Create the class for the project
Public void AddMessageFlow (Element graph){}	Add message flows
Public void AddPartnerFunction (Element graph){}	Add partner functions to partner pool

Public void AddMediatorFunction (Element graph){}	Add mediator functions to mediator pool
Public void AddMediatorEvent (Element graph){}	Add mediator event to mediator pool
Public void AddMediatorSequence (Element graph){}	Add sequences
Public void AddMediatorGateway (Element graph){}	Add gateways
Public void OutputSubProcessModel (){}	Write BPMN file (.bpmn)

III. XML Files of Models

III.1. XML File of Collaborative Network Model

```
<CollaborativeNetworkModel name="">
          <Graph id="x-auto-118">
                    <Network name="Main network" id="gwt-uid-39" x="288.0" y="66.0" />
                    <OperationObjective name="Design Product" id="gwt-uid-48"
                              x="116.0" y="224.0">
                              <SameAs name="Design products and services"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447934" />
                              <NearBy name="Design product and process"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447984" />
                              <SubGraph sourceId="x-auto-155" />
                    </OperationObjective>
                    <Objective name="Sell Product" id="gwt-uid-62" x="324.0" y="233.0">
                              <SameAs name="Sell via broker"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447671" />
                              <NearBy name="Sell via store"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447679" />
                              <SubGraph sourceId="x-auto-134" />
                    </Objective>
                    <StrategyObjective name="Design offer strategy" id="gwt-uid-76"
                              x="496.0" y="227.0">
                              <SameAs name="Design offering strategy"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447619" />
                              <NearBy name="Market products or services to relevant custom"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447896" />
                              <SubGraph sourceId="x-auto-146" />
                    </StrategyObjective>
                    <ObjectiveRelationship name="my link element"
                              id="gwt-uid-59">
                              <Target sourceId="gwt-uid-48" />
                              <Source sourceId="gwt-uid-39" />
                              <Points>
                                        <EndPoint id="gwt-uid-61" x="116.0" y="264.0" previous="gwt-uid-60" />
                                        <StartPoint id="gwt-uid-60" x="288.0" y="107.0" next="gwt-uid-61" />
                                        <Point id="gwt-uid-60" x="288.0" y="107.0" next="gwt-uid-61" />
                              </Points>
                    </ObjectiveRelationship>
                    <ObjectiveRelationship name="my link element"
                              id="gwt-uid-73">
                              <Target sourceId="gwt-uid-62" />
                              <Source sourceId="gwt-uid-39" />
                              <Points>
                                        <EndPoint id="gwt-uid-75" x="324.0" y="259.0" previous="gwt-uid-74" />
                                        <StartPoint id="gwt-uid-74" x="338.0" y="116.0" next="gwt-uid-75" />
                                        <Point id="gwt-uid-74" x="338.0" y="116.0" next="gwt-uid-75" />
                              </Points>
                    </ObjectiveRelationship>
                    <ObjectiveRelationship name="my link element"
                              id="gwt-uid-87">
                              <Target sourceId="gwt-uid-76" />
                              <Source sourceId="gwt-uid-39" />
                              <Points>
                                        <StartPoint id="gwt-uid-88" x="344.0" y="57.0" next="gwt-uid-89" />
                                        <Point id="gwt-uid-88" x="344.0" y="57.0" next="gwt-uid-89" />
                                        <EndPoint id="gwt-uid-89" x="514.0" y="227.0" previous="gwt-uid-88" />
                              </Points>
                    </ObjectiveRelationship>
          </Graph>
```

```xml
<Graph id="x-auto-155">
        <PartnerRelationship name="my link element" id="gwt-uid-343">
                <Target sourceId="gwt-uid-334" />
                <Source sourceId="gwt-uid-325" />
                <Points>
                        <StartPoint id="gwt-uid-344" x="281.0" y="140.0" next="gwt-uid-345" />
                        <Point id="gwt-uid-344" x="281.0" y="140.0" next="gwt-uid-345" />
                        <EndPoint id="gwt-uid-345" x="455.0" y="174.0" previous="gwt-uid-344" />
                </Points>
        </PartnerRelationship>
        <Partner name="Manufacturer" id="gwt-uid-334" x="455.0" y="112.0" />
        <ObjectiveRelationship name="my link element"
                id="gwt-uid-357">
                <Target sourceId="gwt-uid-346" />
                <Source sourceId="gwt-uid-325" />
                <Points>
                        <StartPoint id="gwt-uid-358" x="221.0" y="87.0" next="gwt-uid-359" />
                        <Point id="gwt-uid-358" x="221.0" y="87.0" next="gwt-uid-359" />
                        <EndPoint id="gwt-uid-359" x="193.0" y="249.0" previous="gwt-uid-358" />
                </Points>
        </ObjectiveRelationship>
        <OperationObjective name="Design plan and prototype"
                id="gwt-uid-346" x="129.0" y="249.0" />
        <ObjectiveRelationship name="my link element"
                id="gwt-uid-371">
                <Target sourceId="gwt-uid-360" />
                <Source sourceId="gwt-uid-334" />
                <Points>
                        <EndPoint id="gwt-uid-373" x="511.0" y="221.0" previous="gwt-uid-372" />
                        <StartPoint id="gwt-uid-372" x="506.0" y="181.0" next="gwt-uid-373" />
                        <Point id="gwt-uid-372" x="506.0" y="181.0" next="gwt-uid-373" />
                </Points>
        </ObjectiveRelationship>
        <Partner name="Co-designer" id="gwt-uid-325" x="221.0" y="79.0" />
        <OperationObjective name="Evaluate design" id="gwt-uid-360"
                x="520.0" y="230.0" />
</Graph>
<Graph id="x-auto-134">
        <Partner name="Manufacturer" id="gwt-uid-118" x="139.0" y="98.0" />
        <Partner name="Client" id="gwt-uid-127" x="349.0" y="98.0" />
        <SupportObjective name="Deliver product" id="gwt-uid-150"
                x="230.0" y="206.0" />
        <OperationObjective name="Buy product" id="gwt-uid-164"
                x="417.0" y="225.0" />
        <ObjectiveRelationship name="my link element"
                id="gwt-uid-147">
                <Target sourceId="gwt-uid-136" />
                <Source sourceId="gwt-uid-118" />
                <Points>
                        <StartPoint id="gwt-uid-148" x="130.0" y="89.0" next="gwt-uid-149" />
                        <Point id="gwt-uid-148" x="130.0" y="89.0" next="gwt-uid-149" />
                        <EndPoint id="gwt-uid-149" x="97.0" y="232.0" previous="gwt-uid-148" />
                </Points>
        </ObjectiveRelationship>
        <ObjectiveRelationship name="my link element"
                id="gwt-uid-161">
                <Target sourceId="gwt-uid-150" />
                <Source sourceId="gwt-uid-118" />
                <Points>
                        <EndPoint id="gwt-uid-163" x="239.0" y="206.0" previous="gwt-uid-162" />
                        <StartPoint id="gwt-uid-162" x="194.0" y="168.0" next="gwt-uid-163" />
                        <Point id="gwt-uid-162" x="194.0" y="168.0" next="gwt-uid-163" />
                </Points>
        </ObjectiveRelationship>
        <ObjectiveRelationship name="my link element"
                id="gwt-uid-175">
                <Target sourceId="gwt-uid-164" />
                <Source sourceId="gwt-uid-127" />
                <Points>
                        <EndPoint id="gwt-uid-177" x="446.0" y="225.0" previous="gwt-uid-176" />
                        <StartPoint id="gwt-uid-176" x="353.0" y="168.0" next="gwt-uid-177" />
                        <Point id="gwt-uid-176" x="353.0" y="168.0" next="gwt-uid-177" />
                </Points>
        </ObjectiveRelationship>
        <OperationObjective name="Sell product" id="gwt-uid-136"
                x="106.0" y="241.0" />
        <PartnerRelationship name="my link element" id="gwt-uid-319">
                <Target sourceId="gwt-uid-127" />
                <Source sourceId="gwt-uid-118" />
                <Points>
                        <EndPoint id="gwt-uid-321" x="349.0" y="121.0" previous="gwt-uid-320" />
                        <StartPoint id="gwt-uid-320" x="199.0" y="156.0" next="gwt-uid-321" />
                        <Point id="gwt-uid-320" x="199.0" y="156.0" next="gwt-uid-321" />
                </Points>
        </PartnerRelationship>
</Graph>
<Graph id="x-auto-146">
```

```
                <StrategyObjective name="Collect information" id="gwt-uid-202"
                        x="143.0" y="254.0" />
                <Partner name="Manager" id="gwt-uid-190" x="441.0" y="142.0" />
                <StrategyObjective name="Develop goal" id="gwt-uid-216"
                        x="511.0" y="259.0" />
                <ObjectiveRelationship name="my link element"
                        id="gwt-uid-213">
                        <Target sourceId="gwt-uid-202" />
                        <Source sourceId="gwt-uid-181" />
                        <Points>
                                <EndPoint id="gwt-uid-215" x="153.0" y="254.0" previous="gwt-uid-214" />
                                <StartPoint id="gwt-uid-214" x="202.0" y="211.0" next="gwt-uid-215" />
                                <Point id="gwt-uid-214" x="202.0" y="211.0" next="gwt-uid-215" />
                        </Points>
                </ObjectiveRelationship>
                <ObjectiveRelationship name="my link element"
                        id="gwt-uid-227">
                        <Target sourceId="gwt-uid-216" />
                        <Source sourceId="gwt-uid-190" />
                        <Points>
                                <StartPoint id="gwt-uid-228" x="441.0" y="149.0" next="gwt-uid-229" />
                                <Point id="gwt-uid-228" x="441.0" y="149.0" next="gwt-uid-229" />
                                <EndPoint id="gwt-uid-229" x="511.0" y="265.0" previous="gwt-uid-228" />
                        </Points>
                </ObjectiveRelationship>
                <PartnerRelationship name="my link element" id="gwt-uid-322">
                        <Target sourceId="gwt-uid-190" />
                        <Source sourceId="gwt-uid-181" />
                        <Points>
                                <StartPoint id="gwt-uid-323" x="262.0" y="217.0" next="gwt-uid-324" />
                                <Point id="gwt-uid-323" x="262.0" y="217.0" next="gwt-uid-324" />
                                <EndPoint id="gwt-uid-324" x="441.0" y="201.0" previous="gwt-uid-323" />
                        </Points>
                </PartnerRelationship>
                <Partner name="Manufacturer" id="gwt-uid-181" x="202.0" y="158.0" />
        </Graph>
</CollaborativeNetworkModel>
```

III.2. XML File of Function Model

```
<FunctionModel name="">
        <Graph id="x-auto-60">
                <InOutFlow name="my link element" id="gwt-uid-94">
                        <Target sourceId="gwt-uid-76" />
                        <Source sourceId="gwt-uid-85" />
                        <Points>
                                <EndPoint id="gwt-uid-96" x="138.0" y="39.0" previous="gwt-uid-95" />
                                <StartPoint id="gwt-uid-95" x="78.0" y="41.0" next="gwt-uid-96" />
                                <Point id="gwt-uid-95" x="78.0" y="41.0" next="gwt-uid-96" />
                        </Points>
                </InOutFlow>
                <InOutFlow name="my link element" id="gwt-uid-106">
                        <Target sourceId="gwt-uid-97" />
                        <Source sourceId="gwt-uid-76" />
                        <Points>
                                <StartPoint id="gwt-uid-107" x="218.0" y="45.0" next="gwt-uid-108" />
                                <Point id="gwt-uid-107" x="218.0" y="45.0" next="gwt-uid-108" />
                                <EndPoint id="gwt-uid-108" x="262.0" y="40.0" previous="gwt-uid-107" />
                        </Points>
                </InOutFlow>
                <Function name="gather external info" id="gwt-uid-76" x="138.0"
                        y="23.0" partner="Manufacturer">
                        <SameAs name="Gather external information about environment"

        ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_00000018575029447621" />
                        <NearBy name="Capital asset requisition"

        ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_00000006291279261749" />
                </Function>
                <InOutFlow name="my link element" id="gwt-uid-89">
                        <Target sourceId="gwt-uid-71" />
                        <Source sourceId="gwt-uid-80" />
                        <Points>
                                <EndPoint id="gwt-uid-91" x="142.0" y="129.0" previous="gwt-uid-90" />
                                <StartPoint id="gwt-uid-90" x="88.0" y="141.0" next="gwt-uid-91" />
                                <Point id="gwt-uid-90" x="88.0" y="141.0" next="gwt-uid-91" />
                        </Points>
                </InOutFlow>
                <OutputMessage name="output" id="gwt-uid-92" x="284.0"
                        y="137.0" />
                <InOutFlow name="my link element" id="gwt-uid-101">
                        <Target sourceId="gwt-uid-92" />
                        <Source sourceId="gwt-uid-71" />
                        <Points>
```

```
                                        <EndPoint id="gwt-uid-103" x="284.0" y="141.0" previous="gwt-uid-102" />
                                        <StartPoint id="gwt-uid-102" x="222.0" y="154.0" next="gwt-uid-103" />
                                        <Point id="gwt-uid-102" x="222.0" y="154.0" next="gwt-uid-103" />
                              </Points>
                    </InOutFlow>
                    <Function name="Define objective" id="gwt-uid-71" x="142.0"
                              y="116.0" partner="Manager">
                              <SameAs name="Develop goals"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447620" />
                    </Function>
                    <InputMessage name="input" id="gwt-uid-80" x="28.0" y="129.0" />
                    <Function name="Manage offering life-cycle" id="gwt-uid-104"
                              x="144.0" y="203.0" partner="Manager">
                              <SameAs name="Manage offering life-cycle"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447623" />
                    </Function>
                    <InputMessage name="input" id="gwt-uid-85" x="27.0" y="50.0" />
                    <OutputMessage name="output" id="gwt-uid-97" x="262.0"
                              y="33.0" />
                    <Function name="Receive payment" id="gwt-uid-113" x="144.0"
                              y="281.0" partner="Manufacturer">
                              <SameAs name="Receive payment"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447674" />
                    </Function>
                    <Function name="Deliver" id="gwt-uid-122" x="406.0" y="34.0"
                              partner="Manufacturer">
                              <SameAs name="Deliver"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447667" />
                    </Function>
                    <Function name="Obtain order" id="gwt-uid-131" x="400.0" y="100.0"
                              partner="Manufacturer">
                              <SameAs name="Obtain order"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447657" />
                    </Function>
                    <Function name="Pay invoice" id="gwt-uid-140" x="399.0" y="175.0"
                              partner="Client">
                              <NearBy name="Pay"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447760" />
                    </Function>
                    <Function name="Receive delivery" id="gwt-uid-149" x="395.0"
                              y="245.0" partner="Client">
                              <NearBy name="Receive"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447766" />
                    </Function>
                    <Function name="Place order" id="gwt-uid-158" x="397.0" y="308.0"
                              partner="Client">
                              <SameAs name="Place order"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447758" />
                    </Function>
                    <Function name="Develop product design plan" id="gwt-uid-167"
                              x="544.0" y="32.0" partner="Co-designer">
                              <SameAs name="Develop new product/service concept and plans"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447939" />
                    </Function>
                    <Function name="Design and build prototype" id="gwt-uid-176"
                              x="543.0" y="93.0" partner="Co-designer">
                              <NearBy name="Design product and process"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447910" />
                    </Function>
                    <Function name="Prepare production" id="gwt-uid-185" x="540.0"
                              y="162.0" partner="Manufacturer">
                              <SameAs name="Prepare for production"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447935" />
                    </Function>
                    <Function name="Test effectiveness" id="gwt-uid-194" x="538.0"
                              y="230.0" partner="Manufacturer">
                              <SameAs name="Test effectiveness of new or revised products"

          ontologyId="http://www.semanticweb.org/mise/business_ontology.owl#OWLNamedIndividual_0000018575029447943" />
                    </Function>
          </Graph>
</FunctionModel>
```

III.3. XML File of Collaborative Process Cartography

```
<?xml version="1.0" encoding="UTF-8"?>
<bpmn20:definitions xmlns:bpmn20="http://www.omg.org/spec/BPMN/20100524/MODEL"
xmlns:bpmndi="http://www.omg.org/spec/BPMN/20100524/DI" xmlns:dc="http://www.omg.org/spec/DD/20100524/DC"
xmlns:di="http://www.omg.org/spec/DD/20100524/DI" xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
xmlns:vprop="http://docs.oasis-open.org/wsbpel/2.0/varprop" xmlns:wsdl11="http://schemas.xmlsoap.org/wsdl/"
xmlns:xs="http://www.w3.org/2001/XMLSchema" exporter="PetalsBPM" expressionLanguage="http://www.w3.org/1999/XPath"
id="_1337952414672id370" targetNamespace="http://com.ebmwebsourcing.petalsbpm/model"
typeLanguage="http://www.w3.org/2001/XMLSchema">
  <bpmn20:collaboration id="_1337952414673id371" isClosed="false">
    <bpmn20:participant id="_1337953099605id418" name="strategy pool" processRef="_1337953099605id419"/>
    <bpmn20:participant id="_1337953104266id427" name="operation pool" processRef="_1337953104267id428"/>
    <bpmn20:participant id="_1337953110181id436" name="support pool" processRef="_1337953110181id437"/>
    <bpmn20:messageFlow id="_1337953180131id466" name="test1" sourceRef="_1337953151220id447"
targetRef="_1337953155315id452"/>
    <bpmn20:messageFlow id="_1337953210140id494" name="test3" sourceRef="_1337953203176id484"
targetRef="_1337953187515id470"/>
    <bpmn20:messageFlow id="_1337953195465id480" name="test2" sourceRef="_1337953187515id470"
targetRef="_1337953167328id462"/>
  </bpmn20:collaboration>
  <bpmn20:process id="_1337953099605id419" isClosed="false" isExecutable="false" processType="None">
    <bpmn20:laneSet>
      <bpmn20:lane id="_1337953099607id421" name="Strategy">
        <bpmn20:flowNodeRef>_1337953151220id447</bpmn20:flowNodeRef>
        <bpmn20:childLaneSet id="_1337953099608id422"/>
      </bpmn20:lane>
    </bpmn20:laneSet>
    <bpmn20:task id="_1337953151220id447" name="strategy test one"/>
  </bpmn20:process>
  <bpmn20:process id="_1337953104267id428" isClosed="false" isExecutable="false" processType="None">
    <bpmn20:laneSet>
      <bpmn20:lane id="_1337953104269id430" name="Operation">
        <bpmn20:flowNodeRef>_1337953187515id470</bpmn20:flowNodeRef>
        <bpmn20:flowNodeRef>_1337953155315id452</bpmn20:flowNodeRef>
        <bpmn20:childLaneSet id="_1337953104269id431"/>
      </bpmn20:lane>
    </bpmn20:laneSet>
    <bpmn20:task id="_1337953187515id470" name="operation test two"/>
    <bpmn20:task default="_1337953187548id477" id="_1337953155315id452" name="operation test one"/>
    <bpmn20:sequenceFlow id="_1337953187548id477" name="" sourceRef="_1337953155315id452" targetRef="_1337953187515id470"/>
  </bpmn20:process>
  <bpmn20:process id="_1337953110181id437" isClosed="false" isExecutable="false" processType="None">
    <bpmn20:laneSet>
      <bpmn20:lane id="_1337953110185id439" name="Support">
        <bpmn20:flowNodeRef>_1337953203176id484</bpmn20:flowNodeRef>
        <bpmn20:flowNodeRef>_1337953167328id462</bpmn20:flowNodeRef>
        <bpmn20:childLaneSet id="_1337953110185id440"/>
      </bpmn20:lane>
    </bpmn20:laneSet>
    <bpmn20:task id="_1337953203176id484" name="support test two"/>
    <bpmn20:task default="_1337953203232id491" id="_1337953167328id462" name="support test one"/>
    <bpmn20:sequenceFlow id="_1337953203232id491" name="" sourceRef="_1337953167328id462" targetRef="_1337953203176id484"/>
  </bpmn20:process>
  <bpmndi:BPMNDiagram resolution="0.0">
    <bpmndi:BPMNPlane bpmnElement="_1337952414673id371" id="_1337952414673id371_diagram">
      <bpmndi:BPMNShape bpmnElement="_1337953099605id418" id="_1337953099605id418_diagram" isExpanded="true"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="200.0" width="500.0" x="116.0" y="42.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="_1337953099607id421" id="_1337953099607id421_diagram" isExpanded="false"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="200.0" width="480.0" x="136.0" y="42.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="_1337953104266id427" id="_1337953104266id427_diagram" isExpanded="true"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="200.0" width="500.0" x="117.0" y="259.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="_1337953104269id430" id="_1337953104269id430_diagram" isExpanded="false"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="200.0" width="480.0" x="137.0" y="259.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="_1337953110181id436" id="_1337953110181id436_diagram" isExpanded="true"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="200.0" width="500.0" x="117.0" y="475.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="_1337953110185id439" id="_1337953110185id439_diagram" isExpanded="false"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="200.0" width="480.0" x="137.0" y="475.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="_1337953151220id447" id="_1337953151220id447_diagram" isExpanded="false"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="50.0" width="100.0" x="278.0" y="117.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="_1337953155315id452" id="_1337953155315id452_diagram" isExpanded="false"
```

```
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="50.0" width="100.0" x="252.0" y="308.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="_1337953167328id462" id="_1337953167328id462_diagram" isExpanded="false"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="50.0" width="100.0" x="269.0" y="512.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNEdge bpmnElement="_1337953180131id466" id="_1337953180131id466_diagram">
        <di:waypoint x="324.0" y="167.0"/>
        <di:waypoint x="305.0" y="308.0"/>
      </bpmndi:BPMNEdge>
      <bpmndi:BPMNShape bpmnElement="_1337953187515id470" id="_1337953187515id470_diagram" isExpanded="false"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="50.0" width="100.0" x="461.0" y="302.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNEdge bpmnElement="_1337953187548id477" id="_1337953187548id477_diagram">
        <di:waypoint x="352.0" y="330.0"/>
        <di:waypoint x="461.0" y="328.0"/>
      </bpmndi:BPMNEdge>
      <bpmndi:BPMNEdge bpmnElement="_1337953195465id480" id="_1337953195465id480_diagram">
        <di:waypoint x="487.0" y="354.0"/>
        <di:waypoint x="341.0" y="512.0"/>
      </bpmndi:BPMNEdge>
      <bpmndi:BPMNShape bpmnElement="_1337953203176id484" id="_1337953203176id484_diagram" isExpanded="false"
isHorizontal="false" isMarkerVisible="false" isMessageVisible="false">
        <dc:Bounds height="50.0" width="100.0" x="445.0" y="515.0"/>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNEdge bpmnElement="_1337953203232id491" id="_1337953203232id491_diagram">
        <di:waypoint x="369.0" y="536.0"/>
        <di:waypoint x="448.0" y="536.0"/>
      </bpmndi:BPMNEdge>
      <bpmndi:BPMNEdge bpmnElement="_1337953210140id494" id="_1337953210140id494_diagram">
        <di:waypoint x="499.0" y="512.0"/>
        <di:waypoint x="509.0" y="354.0"/>
      </bpmndi:BPMNEdge>
    </bpmndi:BPMNPlane>
  </bpmndi:BPMNDiagram>
</bpmn20:definitions>
```

IV. Codes of Ontology Tree

IV.1. OntologyTreePanel.java

```
package com.mise2.client.geasydiagrameditorsample.ui.tree;

import java.util.Arrays;

import com.extjs.gxt.ui.client.data.ModelData;
import com.extjs.gxt.ui.client.data.ModelIconProvider;
import com.extjs.gxt.ui.client.event.ButtonEvent;
import com.extjs.gxt.ui.client.event.SelectionListener;
import com.extjs.gxt.ui.client.store.Store;
import com.extjs.gxt.ui.client.store.TreeStore;
import com.extjs.gxt.ui.client.util.IconHelper;
import com.extjs.gxt.ui.client.widget.ContentPanel;
import com.extjs.gxt.ui.client.widget.button.Button;
import com.extjs.gxt.ui.client.widget.form.StoreFilterField;
import com.extjs.gxt.ui.client.widget.grid.ColumnConfig;
import com.extjs.gxt.ui.client.widget.grid.ColumnModel;
import com.extjs.gxt.ui.client.widget.layout.FitLayout;
import com.extjs.gxt.ui.client.widget.toolbar.ToolBar;
import com.extjs.gxt.ui.client.widget.treegrid.TreeGrid;
import com.extjs.gxt.ui.client.widget.treegrid.TreeGridCellRenderer;
import com.google.gwt.user.client.ui.AbstractImagePrototype;
import com.google.gwt.xml.client.Document;
import com.google.gwt.xml.client.Element;
import com.google.gwt.xml.client.Node;
import com.google.gwt.xml.client.NodeList;
import com.google.gwt.xml.client.XMLParser;
import com.mise2.client.geasydiagrameditorsample.customicons.Resources;

public class OntologyTreePanel {

          //tree model attributes
          private String ontologyXML = "";
          private String OBJECTIVE =
"http://www.semanticweb.org/mise/business_ontology.owl#OWLClass_00000000693993948392";//set value
          private String FUNCTION =
"http://www.semanticweb.org/mise/business_ontology.owl#OWLClass_00000000694108630566";//set value
```

```
    private String MESSAGE =
"http://www.semanticweb.org/mise/business_ontology.owl#OWLClass_00000000694110790058";//set value

    private OntologyTreeNode rootNode;
    private OntologyTreeNode objectiveNodes;
    private OntologyTreeNode functionNodes;
    private OntologyTreeNode messageNodes;

    //tree grid attributes
    private ContentPanel treeGridPanel;
    private ToolBar toolBar;
    private Button uploadButton;

    public OntologyTreePanel(){
        rootNode = new OntologyTreeNode();
        objectiveNodes = new OntologyTreeNode("Objective");
        functionNodes = new OntologyTreeNode("Function");
        messageNodes = new OntologyTreeNode("Message");

        toolBar = new ToolBar();
        uploadButton = new Button("Upload your client ontology");
        uploadButton.setIcon(Resources.ICONS.ontology_tab_icon());
        uploadButton.addSelectionListener(new SelectionListener<ButtonEvent>(){
            public void componentSelected(ButtonEvent ce){
                FileUploadWindow window = new FileUploadWindow();
                window.getWindow().show();
            }
        });
        toolBar.add(uploadButton);

        treeGridPanel = new ContentPanel();
        treeGridPanel.setBodyBorder(false);
        treeGridPanel.setHeaderVisible(false);
        treeGridPanel.setBorders(false);
        treeGridPanel.setLayout(new FitLayout());
        treeGridPanel.setFrame(true);
        treeGridPanel.setHeight(400);
        treeGridPanel.setBottomComponent(toolBar);
    }

    public void setOntology(String ontologyXML){
        this.ontologyXML = ontologyXML;
        createTreeModelData();
        createTreeGridPanel();
    }

    //--------------create tree grid data
    public void createTreeModelData(){
        Document ontologyDom = XMLParser.parse(ontologyXML);
        NodeList IndividualNodes = ontologyDom.getElementsByTagName("NamedIndividual");

        System.out.println("[OntologyTreePanel] IndividualNodes length: "+IndividualNodes.getLength());
        for(int i=0;i<IndividualNodes.getLength();i++){
            Node IndividualNode = IndividualNodes.item(i);

            String nodeID = ((Element)IndividualNode).getAttribute("rdf:about");//change

            Node IndividualNodeType = IndividualNode.getChildNodes().item(1);
            String nodeType = ((Element)IndividualNodeType).getAttribute("rdf:resource");//change

            Node IndividualNodeLabel = IndividualNode.getChildNodes().item(3);
            String nodeLabel = IndividualNodeLabel.getFirstChild().getNodeValue();//change

            String nodeName = "no name";
            Node IndividualNodeName = IndividualNode.getChildNodes().item(5);

    if(IndividualNodeName.getNodeName().equals("business_ontology:OWLDataProperty_00000005984939660766")) nodeName =
IndividualNodeName.getFirstChild().getNodeValue();//change

            if(nodeType!=null&&nodeType.equals(OBJECTIVE)){
                OntologyTreeNodeElement element  = new
OntologyTreeNodeElement(nodeName,nodeLabel,nodeID);
                objectiveNodes.add(element);
            }
            if(nodeType!=null&&nodeType.equals(FUNCTION)){
                OntologyTreeNodeElement element = new
OntologyTreeNodeElement(nodeName,nodeLabel,nodeID);
                functionNodes.add(element);
            }
            if(nodeType!=null&&nodeType.equals(MESSAGE)){
                OntologyTreeNodeElement element  = new
OntologyTreeNodeElement(nodeName,nodeLabel,nodeID);
                messageNodes.add(element);
            }
        }

        rootNode.add(objectiveNodes);
```

```
                    rootNode.add(functionNodes);
                    rootNode.add(messageNodes);
          }

          //--------------create tree grid panel
          public void createTreeGridPanel(){

                    TreeStore<ModelData> store = new TreeStore<ModelData>();
                    store.add(rootNode.getChildren(), true);
                    //--------------column start
                    ColumnConfig name = new ColumnConfig("name","Name",70);
                    name.setRenderer(new TreeGridCellRenderer<ModelData>());
                    ColumnConfig label = new ColumnConfig("label","Label",70);
                    ColumnConfig individualID = new ColumnConfig("individualID","ID",70);
                    ColumnModel columnModel = new ColumnModel(Arrays.asList(name,label,individualID));
                    //--------------filter start
                    StoreFilterField<ModelData> filter = new StoreFilterField<ModelData>() {


                                        @Override
                                        protected boolean doSelect(Store<ModelData> store,
                                                            ModelData parent, ModelData record, String property,
                                                            String filter) {
                                                  // TODO Auto-generated method stub
                                                  if (record instanceof OntologyTreeNode) {
                                          return false;
                                        }
                                        String name = record.get("name");
                                        name = name.toLowerCase();
                                        if (name.startsWith(filter.toLowerCase())) {
                                          return true;
                                        }
                                        return false;
                                        }

              };
                    filter.bind(store);
                    //--------------create tree
                    TreeGrid<ModelData> tree = new TreeGrid<ModelData>(store,columnModel);
                    tree.setBorders(true);
                    tree.setAutoExpandColumn("name");
                    tree.setTrackMouseOver(false);
                    tree.setIconProvider(new ModelIconProvider<ModelData>(){

                              @Override
                              public AbstractImagePrototype getIcon(ModelData model) {
                                        // TODO Auto-generated method stub
                                        if(model instanceof OntologyTreeNodeElement){

                                                  OntologyTreeNodeElement element = (OntologyTreeNodeElement) model;
                                                  String type = element.getParent().toString();

                                                  if(type == "Objective"){
                                                            return IconHelper.createPath("images/tree_objective.png");//------
--------set value
                                                  }
                                                  if(type == "Function"){
                                                            return IconHelper.createPath("images/tree_function.png");//-------
-------set value
                                                  }
                                                  if(type == "Message"){
                                                            return IconHelper.createPath("images/tree_message.png");//--------
------set value
                                                  }
                                        }
                                        return null;
                              }});
                    treeGridPanel.setTopComponent(filter);
                    treeGridPanel.add(tree);
          }

          public ContentPanel getTreeGridPanel() {
                    return treeGridPanel;
          }

          public void setTreeGridPanel(ContentPanel treeGridPanel) {
                    this.treeGridPanel = treeGridPanel;
          }

          public OntologyTreeNode getRootNode() {
                    return rootNode;
          }

          public void setRootNode(OntologyTreeNode rootNode) {
                    this.rootNode = rootNode;
          }
}
```

IV.2. OntologyTreeGridDragPanel.java

```
package com.mise2.client.geasydiagrameditorsample.ui.tree;

import java.util.Arrays;

import com.extjs.gxt.ui.client.Style.HorizontalAlignment;
import com.extjs.gxt.ui.client.Style.Orientation;
import com.extjs.gxt.ui.client.data.ModelData;
import com.extjs.gxt.ui.client.data.ModelIconProvider;
import com.extjs.gxt.ui.client.dnd.TreeGridDragSource;
import com.extjs.gxt.ui.client.dnd.TreeGridDropTarget;
import com.extjs.gxt.ui.client.event.DNDEvent;
import com.extjs.gxt.ui.client.store.Store;
import com.extjs.gxt.ui.client.store.TreeStore;
import com.extjs.gxt.ui.client.util.IconHelper;
import com.extjs.gxt.ui.client.util.Margins;
import com.extjs.gxt.ui.client.util.Padding;
import com.extjs.gxt.ui.client.widget.ContentPanel;
import com.extjs.gxt.ui.client.widget.LayoutContainer;
import com.extjs.gxt.ui.client.widget.MessageBox;
import com.extjs.gxt.ui.client.widget.Window;
import com.extjs.gxt.ui.client.widget.button.Button;
import com.extjs.gxt.ui.client.widget.form.StoreFilterField;
import com.extjs.gxt.ui.client.widget.grid.ColumnConfig;
import com.extjs.gxt.ui.client.widget.grid.ColumnModel;
import com.extjs.gxt.ui.client.widget.layout.FitLayout;
import com.extjs.gxt.ui.client.widget.layout.HBoxLayout;
import com.extjs.gxt.ui.client.widget.layout.HBoxLayoutData;
import com.extjs.gxt.ui.client.widget.layout.RowLayout;
import com.extjs.gxt.ui.client.widget.layout.TableLayout;
import com.extjs.gxt.ui.client.widget.layout.BoxLayout.BoxLayoutPack;
import com.extjs.gxt.ui.client.widget.treegrid.TreeGrid;
import com.extjs.gxt.ui.client.widget.treegrid.TreeGridCellRenderer;
import com.extjs.gxt.ui.client.event.ButtonEvent;
import com.extjs.gxt.ui.client.event.DNDListener;
import com.extjs.gxt.ui.client.event.SelectionListener;
import com.google.gwt.user.client.ui.AbstractImagePrototype;
import com.mise2.client.geasydiagrameditorsample.MyFrameLayout;

public class OntologyTreeGridDragPanel extends Window{

        private OntologyTreeNode rootNode;
        private ContentPanel mainVPanel;
        private ContentPanel treeGragSourcePanel;
        private ContentPanel treeGragTargetPanel;

        private TreeGrid<ModelData> tree_nearby;
        private TreeGrid<ModelData> tree_sameas;
        private TreeStore<ModelData> store_sameas;
        private TreeStore<ModelData> store_nearby;

        private TreeGrid<ModelData> tree;
        private TreeStore<ModelData> store;
        private OntologyTreeNodeElement sameasElement;
        private OntologyTreeNodeElement nearbyElement;

        public OntologyTreeGridDragPanel(){
                //------------------------normal panels setting
                this.rootNode = MyFrameLayout.getInstance().getEastPanel().getTreePanel().getRootNode();
                mainVPanel = new ContentPanel();
                mainVPanel.setLayout(new RowLayout(Orientation.VERTICAL));
                mainVPanel.setFrame(false);
                mainVPanel.setHeaderVisible(false);
                mainVPanel.setBodyBorder(false);
                mainVPanel.setBorders(false);
                mainVPanel.setButtonAlign(HorizontalAlignment.CENTER);

                treeGragSourcePanel = new ContentPanel();
                TableLayout sourcelayout = new TableLayout(2);
                sourcelayout.setCellPadding(3);
                treeGragSourcePanel.setLayout(sourcelayout);
                treeGragSourcePanel.setHeaderVisible(false);
                treeGragSourcePanel.setBodyBorder(false);
                treeGragSourcePanel.setBorders(false);

                treeGragTargetPanel = new ContentPanel();
                TableLayout targetlayout = new TableLayout(1);
                targetlayout.setCellPadding(2);
                treeGragTargetPanel.setLayout(targetlayout);
                treeGragTargetPanel.setHeaderVisible(false);
                treeGragTargetPanel.setBodyBorder(false);
                treeGragTargetPanel.setBorders(false);
```

```
                    this.CreateTreeGragSourcePanel();
                    this.CreateTreeGragTargetPanel();

                    //------------------------button setting
                    mainVPanel.add(treeGragSourcePanel);
                    Button validateButton = new Button("Validate");
                    Button cancelButton = new Button("Cancel");
                    ContentPanel panel = new ContentPanel();
                    panel.setBorders(false);
                    panel.setBodyBorder(false);
                    panel.setHeaderVisible(false);

                    LayoutContainer c = new LayoutContainer();
                    HBoxLayout layout = new HBoxLayout();
                    layout.setPadding(new Padding(5));
                    layout.setPack(BoxLayoutPack.END);
                    c.setLayout(layout);
                    HBoxLayoutData layoutdata = new HBoxLayoutData(new Margins(0,5,0,0));
                    c.add(validateButton,layoutdata);
                    c.add(cancelButton,layoutdata);

                    panel.add(c);
                    mainVPanel.add(panel);

                    validateButton.addSelectionListener(new SelectionListener<ButtonEvent>(){
                              public void componentSelected(ButtonEvent ce){
                                        if(tree_sameas.getStore().getAt(0)!=null) {
                                                  sameasElement = (OntologyTreeNodeElement) tree_sameas.getStore().getAt(0);
                                        }
                                        if(tree_nearby.getStore().getAt(0)!=null){
                                                  nearbyElement = (OntologyTreeNodeElement) tree_nearby.getStore().getAt(0);
                                        }
                                        hide();
                              }
                    });
                    cancelButton.addSelectionListener(new SelectionListener<ButtonEvent>(){
                              public void componentSelected(ButtonEvent ce){
                                        hide();
                              }
                    });
                    //------------------------window setting
                    setSize(630,380);
                    setPlain(true);
                    setModal(true);
                    setHeading("Choose same/near elements");
                    setLayout(new FitLayout());
                    add(mainVPanel);
          }

          //------------------------------------------------create drag source tree panel (which on the left)
          public void CreateTreeGragSourcePanel(){
                    ContentPanel panel = new ContentPanel();
                    panel.setLayout(new FitLayout());
                    panel.setSize(300, 300);
                    panel.setHeading("Ontology Data:");
                    store = new TreeStore<ModelData>();
                    store.add(rootNode.getChildren(), true);
                    //--------------filter start
                    StoreFilterField<ModelData> filter = new StoreFilterField<ModelData>() {
                                        @Override
                                        protected boolean doSelect(Store<ModelData> store,
                                                            ModelData parent, ModelData record, String property,
                                                            String filter) {
                                                  // TODO Auto-generated method stub
                                                  if (record instanceof OntologyTreeNode) {
                                          return false;
                                        }
                                        String name = record.get("name");
                                        name = name.toLowerCase();
                                        if (name.startsWith(filter.toLowerCase())) {
                                          return true;
                                        }
                                        return false;
                                        }
              };
                    filter.bind(store);
                    //--------------create tree
        tree = new TreeGrid<ModelData>(store,this.getColumnModel());
                    this.setTree(tree);
                    new TreeGridDropTarget(tree);
                    TreeGridDragSource source = new TreeGridDragSource(tree);
                    source.addDNDListener(new DNDListener(){
                              @Override
                              public void dragStart(DNDEvent e){
                                        OntologyTreeNodeElement data = (OntologyTreeNodeElement)
tree.getSelectionModel().getSelectedItem();
                                        if(data!=null && data.getChildCount()>0){
```

```
                    e.setCancelled(true);
                    e.getStatus().setStatus(false);
                    MessageBox.alert("Warning", "Please choose leaf node", null);
                    return;
                }
                super.dragStart(e);
            }
        });
        panel.setTopComponent(filter);
        panel.add(tree);
        this.treeGragSourcePanel.add(panel);
    }
    //---------------------------------------------create drag target tree panel (which on the right)
    public void CreateTreeGragTargetPanel(){
        ContentPanel upPanel = new ContentPanel();
        upPanel.setLayout(new FitLayout());
        upPanel.setSize(300, 150);
        upPanel.setHeading("Same as:");

        ContentPanel bottomPanel = new ContentPanel();
        bottomPanel.setLayout(new FitLayout());
        bottomPanel.setSize(300, 150);
        bottomPanel.setHeading("Near by:");

        store_sameas = new TreeStore<ModelData>();
        tree_sameas = new TreeGrid<ModelData>(store_sameas,this.getColumnModel());
        this.setTree(tree_sameas);
        TreeGridDropTarget target_sameas = new TreeGridDropTarget(tree_sameas);
        target_sameas.addDNDListener(new DNDListener(){
            @Override
            public void dragEnter(DNDEvent e){
                if(tree_sameas.getStore().getCount()>0){
                    e.setCancelled(true);
                    e.getStatus().setStatus(false);
                    MessageBox.alert("Warning", "Please choose one same as element.", null);
                }
                super.dragEnter(e);
            }
        });
        new TreeGridDragSource(tree_sameas);

        store_nearby = new TreeStore<ModelData>();
        tree_nearby = new TreeGrid<ModelData>(store_nearby,this.getColumnModel());
        this.setTree(tree_nearby);
        TreeGridDropTarget target_nearby = new TreeGridDropTarget(tree_nearby);
        target_nearby.addDNDListener(new DNDListener(){
            @Override
            public void dragEnter(DNDEvent e){
                if(tree_nearby.getStore().getCount()>0){
                    e.setCancelled(true);
                    e.getStatus().setStatus(false);
                    MessageBox.alert("Warning", "Please choose one near by element.", null);
                }
                super.dragEnter(e);
            }
        });
        new TreeGridDragSource(tree_nearby);

        upPanel.add(tree_sameas);
        bottomPanel.add(tree_nearby);
        this.treeGragTargetPanel.add(upPanel);
        this.treeGragTargetPanel.add(bottomPanel);
        this.treeGragSourcePanel.add(this.treeGragTargetPanel);
    }
    //---------------------------------------------set the normal things for trees
    public void setTree(TreeGrid<ModelData> tree){
        tree.setBorders(true);
        tree.setAutoExpandColumn("name");
        tree.setTrackMouseOver(false);
        tree.setIconProvider(new ModelIconProvider<ModelData>(){

            @Override
            public AbstractImagePrototype getIcon(ModelData model) {
                // TODO Auto-generated method stub
                if(model instanceof OntologyTreeNodeElement){

                    OntologyTreeNodeElement element = (OntologyTreeNodeElement) model;
                    String type = element.getParent().toString();

                    if(type == "Objective"){
                        return IconHelper.createPath("images/tree_objective.png");//------
--------set value
                    }
                    if(type == "Function"){
                        return IconHelper.createPath("images/tree_function.png");//-------
-------set value
                    }
                    if(type == "Message"){
                        return IconHelper.createPath("images/tree_message.png");//--------
------set value
                    }
                }
                return null;
            }});
    }
    //----------------------------------------------get normal columns
    public ColumnModel getColumnModel(){
        ColumnConfig name = new ColumnConfig("name","Name",70);
        name.setRenderer(new TreeGridCellRenderer<ModelData>());
        ColumnConfig label = new ColumnConfig("label","Label",70);
        ColumnConfig individualID = new ColumnConfig("individualID","ID",70);
        ColumnModel columnModel = new ColumnModel(Arrays.asList(name,label,individualID));
        return columnModel;
    }
    //------------------------getters and setters

    public OntologyTreeNodeElement getSameasElement() {
        return sameasElement;
    }

    public void setSameasElement(OntologyTreeNodeElement sameasElement) {
        this.sameasElement = sameasElement;
    }

    public void setSameasElement(String id){
        //System.out.println(id);
        if(id!=null&&id.length()>0){
            OntologyTreeNodeElement model = (OntologyTreeNodeElement) store.findModel("individualID", id);
            //System.out.println(model);
            store_sameas.add(model, false);
            store.remove(model);
            setSameasElement(model);
        }
    }

    public OntologyTreeNodeElement getNearbyElement() {
        return nearbyElement;
    }

    public void setNearbyElement(OntologyTreeNodeElement nearbyElement) {
        this.nearbyElement = nearbyElement;
    }

    public void setNearbyElement(String id){
        //System.out.println(id);
        if(id!=null&&id.length()>0){
            OntologyTreeNodeElement model = (OntologyTreeNodeElement) store.findModel("individualID", id);
            //System.out.println(model);
            store_nearby.add(model,false);
            store.remove(model);
            setSameasElement(model);
        }
    }

}
```

V. References

(Campbell-Kelly, 2009) Campbell-Kelly, M. (2009). The rise, fall, and resurrection of software as a service. Communications of the ACM *52*, 28–30.

(Dwyer, 2008) Dwyer, J. (2008). Getting Started Let's GWT Down to Business. In Pro Web 2.0 Application Development with GWT, (Apress), pp. 17–44.

(Gold et al., 2004) Gold, N., Mohan, A., Knight, C., and Munro, M. (2004). Understanding service-oriented software. Software, IEEE *21*, 71–77.

(Gupta, 2008a) Gupta, V. (2008a). Accelerated GWT: Building Enterprise Google Web Toolkit Applications (Apress).

(Gupta, 2008b) Gupta, V. (2008b). GWT Architecture and Internal Features. In Accelerated GWT, (Apress), pp. 27–55.

(Kang et al., 2010) Kang, S., Myung, J., Yeon, J., Ha, S., Cho, T., Chung, J., and Lee, S. (2010). A General Maturity Model and Reference Architecture for SaaS Service. In Database Systems for Advanced Applications, H. Kitagawa, Y. Ishikawa, Q. Li, and C. Watanabe, eds. (Springer Berlin / Heidelberg), pp. 337–346.

(Mäkilä et al., 2010) Mäkilä, T., Järvi, A., Rönkkö, M., and Nissilä, J. (2010). How to Define Software-as-a-Service – An Empirical Study of Finnish SaaS Providers. In Software Business, P. Tyrväinen, S. Jansen, M.A. Cusumano, W. Aalst, J. Mylopoulos, M. Rosemann, M.J. Shaw, C. Szyperski, W. Aalst, J. Mylopoulos, et al., eds. (Springer Berlin Heidelberg), pp. 115–124.

(Plante, 2006) Plante, F. (2006). Introducing the GMF Runtime.

(Sääksjärvi et al., 2005) Sääksjärvi, M., Lassila, A., and Nordström, H. (2005). Evaluating the software as a service business model: From CPU time-sharing to online innovation sharing. In Proceedings of the IADIS International Conference e-Society, pp. 177–186.

(Schaffer, 2009) Schaffer, H.E. (2009). X as a service, cloud computing, and the need for good judgment. IT Professional *11*, 4–5.

(Slender, 2009) Slender, G. (2009). Overview of Ext GWT and GWT. In Developing with Ext GWT, (Apress), pp. 1–7.

(Smeets et al., 2008a) Smeets, B., Boness, U., and Bankras, R. (2008a). GWT UI Components. In Beginning Google Web Toolkit, (Apress), pp. 59–87.

(Smeets et al., 2008b) Smeets, B., Boness, U., and Bankras, R. (2008b). Introducing Google Web Toolkit (GWT). In Beginning Google Web Toolkit, (Apress), pp. 21–44.

(Sun et al., 2010) Sun, W., Zhang, K., Chen, S.K., Zhang, X., and Liang, H. (2010). Software as a service: An integration perspective. Service-Oriented Computing–ICSOC 2007 558–569.

(Vohra, 2009) Vohra, D. (2008). Ajax with Java-GWT. In Ajax in Oracle JDeveloper, (Springer Berlin Heidelberg), pp. 61–87.