



**HAL**  
open science

# Supply chains Risk management : A simulation and model-based approach

Saleh Eddine Ben Jbara

► **To cite this version:**

Saleh Eddine Ben Jbara. Supply chains Risk management : A simulation and model-based approach. Computational Engineering, Finance, and Science [cs.CE]. Institut polytechnique de Grenoble, 2018. English. NNT: . tel-01715801

**HAL Id: tel-01715801**

**<https://hal.science/tel-01715801>**

Submitted on 23 Feb 2018

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

**THÈSE**

Pour obtenir le grade de

**DOCTEUR DE LA COMMUNAUTE UNIVERSITE GRENOBLE  
ALPES**

Spécialité : **Genie industriel**

Arrêté ministériel : 25 mai 2016

Présentée par

**Saleh Eddine BEN JBARA**

Thèse dirigée par **Gülgün ALPAN**

Codirigée par **Pierre DAVID**

Préparée au sein du **Laboratoire GSCOP**

dans **l'École Doctorale IMEP2**

**Risk management in supply  
chains: a simulation and model-  
based approach**

Thèse soutenue publiquement le « **30/01/2018** »,

Devant le jury composé de :

**M. Khaled HADJ-HAMOU**

Professeur, INSA Lyon, Président

**M. Hamid ALLAoui**

Professeur, Université d'Artois, Rapporteur

**M. Laurent GENESTE**

Professeur, Ecole nationale d'ingénieurs de Tarbes, Rapporteur

**M<sup>me</sup>. Gülgün ALPAN**

Maître de conférences, HDR, Grenoble-INP, Directrice de thèse

**M. Pierre DAVID**

Maître de conférences, Grenoble-INP, Co-encadrant



«Que je plains ceux que n'éclaire pas, dans leurs heures sombres, le grand jour de l'éternité !»  
Citation d'Anne BARRATIN (1894)



## ACKNOWLEDGEMENT

---

Firstly, I would like to express my sincere gratitude to my advisors Gülgün ALPAN and Pierre DAVID for their continuous support, their patience and the energy they spend for assisting me conducting this Ph.D. work. I want to thank them for their guidance, for their availability and for all the ideas, the corrections and the restructuring they provide. Their advices and comments were very valuable for improving the quality of this work and for upgrading my skills.

Besides my advisors, I would like to thank my friends and the members of my music team “Mosaïque” starting with my music tutor Aziz DHIB and my best musician friend Mohamed Amine DERBAL and I want to thank also the Tunisian cultural center of Grenoble for the activities they organize. Thanks to them I had a great time performing the Luth instrument and enjoying good music. Furthermore, I want to thank my neighbors and friends Amine DANDACHY and Hassen KANJ for their kindness and hospitality.

I want to thank the members of G-SCOP laboratory especially the ones who spend efforts for creating a good working climate through organizing special events that enhance synergy and friendship. Big thanks go to my family members who supported me in each important step of my life and in achieving my goals.

Finally, I want to thank Rhone-Alpes region for funding this Ph.D.

## REMERCIEMENTS

---

Premièrement, Je veux exprimer mes sincères remerciements à mes encadrants Gülgün ALPAN et Pierre DAVID pour l’Energie dépensée pour m’assister dans ce travail. Je les remercie pour toutes les corrections, les restructurations, les idées fournies et pour leur disponibilité. Leurs conseils et commentaires étaient d’une grande valeur dans l’amélioration de la qualité de ce travail et dans le développement de mon niveau en général.

En plus de mes encadrants, je veux remercier mes amis ainsi que les membres de mon groupe de musique Mosaïque commençant par mon professeur de musique Aziz DHIB et mon meilleur ami musicien Mohamed Amine DERBAL et je remercie aussi le centre culturel tunisien de Grenoble pour les activités qu’ils organisent. Grace à eux j’ai passé un bon temps pour maîtriser le luth, toute en profitant de la belle musique. Je remercie également mes voisins Amine DANDACHY et Hassen KANJ pour leur gentillesse et hospitalité.

J’adresse mes remerciements aussi aux membres du laboratoire G-SCOP et surtout ceux qui ont essayé de créer une bonne ambiance de travail en organisant des événements qui améliorent la synergie et l’amitié. J’adresse un grand merci aux membres de ma famille qui m’ont toujours supporté dans chaque étape importante de ma vie et dans l’atteinte de mes objectifs.

Enfin, je tiens à remercier la région Rhône-Alpes pour avoir financé cette thèse.

## RESUME EN FRANÇAIS

Les chaînes logistiques (CL) sont en continuelle évolution. Elles changent de configuration, de taille, d'étendue géographique ou de gestion. De nouveaux types de SCs apparaissent grâce au développement technologique et à la mondialisation. Par exemple, l'introduction de la modularité dans les ordinateurs Dell a transformé la CL pour que les utilisateurs fassent une partie de l'assemblage. Le développement des technologies de recyclage et de récupération ont fait apparaître des problématiques de logistique inverse. Le développement de plateformes sécurisées de paiement électronique a ouvert la voie pour le développement des chaînes logistiques digitales comme celle d'Amazon.

Les chercheurs et les gestionnaires des chaînes logistiques n'ont cessé de développer des approches de gestion pour s'adapter aux changements des SCs. Depuis longtemps les efforts se sont concentrés sur la réduction des coûts, l'amélioration de la rentabilité et la compétitivité. Un exemple de ces approches est le Lean management, développée par Toyota au Japon. L'adoption de ces approches s'accompagne par l'apparition de nouveaux défis. Plusieurs chercheurs ont souligné que les avantages de ces approches (niveaux de stocks réduits, délais de livraison plus courts) ont rendu les CL plus vulnérables aux perturbations locales et mondiales ((Regardez par exemple, (Enyinda et al. 2008), (Pfohl et al. 2013);(Tuncel & Alpan 2010)).

La vulnérabilité est inhérente au développement dynamique des SCs et à leur complexité. Comme l'a révélé (Jüttner 2005), 44% des entreprises couvertes par son étude s'attendent à ce que la vulnérabilité de leurs chaînes logistiques augmente au cours des cinq prochaines années

Au cours des dernières années, de nombreuses chaînes logistiques ont subi des perturbations qui ont eu des répercussions négatives sur leur performance. Selon une étude réalisée par (Simchi Levi et al. 2013) sur un échantillon de 209 entreprises internationales, les perturbations de la CL ont induit des impacts négatifs sur la performance financière pour 54% d'entre elles et 64% ont déclaré une baisse de leurs niveaux de service.

Les entreprises ont parfois des difficultés à surmonter ces perturbations comme l'ont évoqué (Hendricks & Singhal 2005). Un célèbre exemple est celui d'Ericsson. En effet comme l'ont décrit (Norrman & Jansson 2004), suite à un orage, un incendie s'est déclaré dans la "salle blanche" d'un fournisseur. La destruction des équipements a interrompu l'envoi de puces radiofréquences à Ericsson. Comme Ericsson n'avait qu'un seul fournisseur pour ce type de composant, la société a perdu sa capacité à vendre et à livrer un de ses produits phares.

Plusieurs mois de production de téléphones portables ont été perdus poussant Ericsson à stopper son activité de téléphonie mobile. Le coût de cette rupture a été estimé à environ 200 millions de dollars.

Les managers des chaînes logistiques rencontrent des échecs dans la gestion de leurs risques. Comme révélé par (Hind & Craighead 2010) pour des sociétés comme Boeing, Cisco et Pfizer, les pertes et / ou dépenses imprévues ont dépassé 2 milliards de dollars en raison des décisions inefficaces en 2001.

(Chopra & ManMohan 2014) indiquent que plusieurs enquêtes ont montré qu'il est difficile pour les managers d'évaluer les actions de maîtrise de risque d'un point de vue de leur efficacité économique. Ceci entraîne des réticences à investir dans des mesures de prévention. De plus, une étude relayée par (Marchese & Paramasivam 2013) conduite par Deloitte sur 600 CL révèle que de nombreuses entreprises ne maîtrisent pas la gestion des risques.

En effet, seuls 33% des interviewés ont utilisé des approches de gestion des risques pour gérer de manière proactive leurs risques. Aussi 45% ont estimé que leur gestion des risques était peu ou pas efficace.

Plusieurs raisons expliquent les échecs. La raison principale selon (Jüttner 2005), est le faible degré de mise en œuvre des instruments de gestion du risque pour les CL. Même si ils sont implémentés, de nombreuses entreprises présentent des processus de gestion des risques immatures. En effet, comme l'a révélé l'étude de (Simchi Levi et al. 2013), 59% des entreprises étudiées ont mis en place des processus immatures, ni proactifs ni flexibles pour traiter les incidents.

Une autre raison est que les entreprises comprennent mal la gestion des risques des CL. Pour (Jüttner 2005), la maîtrise des risques dans les CL est encore souvent vue comme une tâche spécifique à l'entreprise, alors que celle-ci doit couvrir des risques partagés avec les entreprises partenaires. En effet, comme expliqué par (Chopra & ManMohan 2014), traiter un risque individuellement et oublier les interconnexions peut mener à exacerber d'autres risques. De plus, les actions prises par une entreprise pour traiter un risque peuvent augmenter le niveau de risque pour ses partenaires. Les entreprises ont aussi tendance à faire l'erreur de gérer en priorité les risques récurrents à faible impact tout en ignorant les risques à fort impact et de faible vraisemblance.

Les managers des CL ont besoin d'outils et méthodes pour améliorer leur gestion des risques. Même si plusieurs approches existent et font partie de la boîte à outils des managers, elles ne

couvrent pas encore toutes les exigences. La littérature professionnelle et scientifique s'intéresse de plus en plus à ce sujet.

L'objectif de cette thèse est d'apporter sa contribution pour surmonter certaines lacunes relevées par la littérature. Le premier sujet auquel nous contribuons est le manque d'outils pour permettre aux gestionnaires de CL de construire un modèle conceptuel de leur système et ses risques associés. Notre but est également de fournir un modèle pouvant être traduit en un modèle de simulation. Le but est d'être capable de construire une représentation objective et comparable d'un CL et d'obtenir une représentation exploitable avec les techniques de transformation de modèles.

L'analyse par simulation est un outil puissant adapté à l'évaluation des CL. De nombreux chercheurs à l'instar de ((Wu et al. 2006), (Cigolini et al. 2011)) ont cependant repéré que la difficulté rencontrée dans la construction des modèles de simulation freine l'adoption de ces techniques par la communauté. En effet, pour les managers, simuler une CL reste encore une tâche difficile et coûteuse qui exige des efforts d'apprentissage et des compétences avancées. Une des raisons est que les logiciels de simulation actuels utilisent des blocs de construction qui sont souvent éloignés du domaine des CL ou qui ont un faible niveau de granularité par rapport aux éléments à modéliser.

Par conséquent, certains chercheurs proposent des environnements de modélisation pour la simulation qui définissent des méta-modèles pour les CL. La plupart des solutions proposées dans la littérature n'arrivent pas encore à totalement représenter les connaissances du domaine des CL. De plus, la plupart de ces environnements ne prennent pas en compte la modélisation des risques. Cela s'explique en partie par le manque de consensus sur la définition et la catégorisation des risques dans les CL.

Pour surmonter cela, trois questions de recherche sont posées dans ce travail:

- Comment définir un environnement de modélisation pour la simulation qui puisse être facilement adopté par les gestionnaires des SCs?
- Comment définir des éléments de modélisation qui seront à la fois génériques et capables de couvrir le domaine des SCs?
- Comment intégrer efficacement les risques dans les outils de modélisation pour la simulation?

Notre objectif a donc été de proposer les éléments nécessaires à la constitution d'un environnement de modélisation pour les CL et leurs risques permettant une construction « facile » des modèles conceptuels et en imaginant leur transition vers des modèles simulables.

Pour réussir cela des auteurs comme ((Beamon 1998), (Min & Zhou 2002),... ) ont introduit la nécessité d'utiliser un langage de modélisation pour décrire et analyser dynamiquement les scénarios de la SC.

Pour cela notre travail est d'abord de définir un langage de modélisation. Ceci a été réalisé en proposant un méta-modèle pour exprimer la structure des CL, le comportement des CLs, les risques inhérents à ces systèmes.

Ce méta-modèle définit un ensemble de blocs de construction interconnectables. Il est présenté sous la forme d'un profil SysML, qui peut être instancié pour modéliser tout ou partie d'une CL.

Pour maximiser l'acceptabilité des concepts proposés dans le méta-modèle nous sommes basés sur l'analyse des processus décrits dans la référence SCOR. La référence SCOR est l'une des références des plus utilisées dans l'industrie; elle fournit une description textuelle des processus des CL et les associe à un ensemble d'indicateurs de performances. Les concepts de SCOR sont centrés sur une présentation statique des processus. Nous avons donc du réinterpréter les références SCOR pour à la fois décrire les structures physiques des CL et proposer des versions exécutables des opérations. La proposition peut donc être vue comme une extension de SCOR facilitant la connexion à des activités de simulation.

Nous proposons ainsi, des blocs de modélisation qui couvrent les flux transférés entre les opérations de la CL décrites dans SCOR en tant que entrée/sortie des processus. Nous proposons également une extension de SCOR pour prendre en compte les relations et les interactions qui existent entre les partenaires de la CL.

Nous proposons de définir des algorithmes détaillés pour chaque opération décrite dans les processus de SCOR. Enfin, nous proposons un ensemble de briques de construction pour les risques.



Le deuxième point important de la contribution est le support apporté à la création d'un modèle simulable et facilement paramétrable pour permettre la réalisation de campagnes d'expérimentation.

Suite à l'analyse de la littérature nous avons constaté que peu de travaux fournissent une méthode explicite pour construire un modèle de simulation pour une CL. Certains travaux fournissent des blocs de modélisation spécifiques au domaine de la SC sans préciser comment les traduire pour la simulation, comme le travail de (Persson 2011). Un deuxième point délicat relevé dans la littérature est l'absence d'un consensus sur la façon de simuler des risques. Il est en effet difficile de proposer une catégorisation permettant de couvrir à des fins de simulation la variété des risques qu'un CL peut rencontrer. Pour cela nous avons choisi d'étendre les travaux de (Saleh Ebrahimi et al. 2012) en proposant leur traduction vers des modèles de simulation.

Nous avons donc cherché à répondre aux deux questions suivantes :

- Comment traduire simplement et rapidement un modèle conceptuel de CL en un modèle de simulation ?
- Comment assister les gestionnaires de CL dans l'expérimentation de leur modèle de simulation pour évaluer différents scénarios ?

Nous avons donc développé des routines de traduction permettant de créer à partir d'un modèle conceptuel de la SC exprimé avec notre méta-modèle, un modèle de simulation à mettre en œuvre pour simuler les scénarios voulus.

Notre approche décrit comment traduire les éléments de structure de la SC ainsi que les entrée/sortie des blocs d'opérations et des modules exprimant le comportement et les risques.. La traduction est illustrée par la construction de bibliothèques de modules de simulation ARENA.

La solution fournie permet aux utilisateurs de construire rapidement leurs propres modèles de simulation. Différentes étapes sont suivies allant de la traduction de la structure à l'injection de profils de risques en passant par le paramétrage des politiques de gestion de la chaîne. La façon de paramétrer les modèles pour obtenir différents scénarios est également discutée.

Les solutions développées dans cette thèse ont été testées sur une étude de cas. Les résultats montrent comment elles peuvent favoriser l'analyse des risques. L'étude de cas a fourni également des éléments de vérification des méta-modèles, de leur traduction et des

bibliothèques de modules de simulation. L'analyse des résultats de simulation montre que l'outil est efficace pour évaluer les impacts des risques sur les performances d'une CL.

Les travaux réalisés ouvrent d'autres directions de recherche. Notamment, le développement de bibliothèques pour les politiques de gestion des risques des CL. En effet, plusieurs auteurs ont souligné le besoin de définir des contre-mesures réactives pour faire face aux perturbations auxquelles sont confrontés les CL (Ivanov et al. 2014).

Une autre direction est de développer des algorithmes spécifiques pour optimiser les performances des politiques de gestion de risques en tirant profit des possibilités fournies par les combinaisons possibles entre simulation et optimisation.

Une troisième direction consiste à aborder l'intégration de l'environnement de modélisation pour la simulation dans le processus de gestion des risques de l'entreprise. En effet, un problème classique à aborder consiste à trouver la meilleure façon de modéliser les données recueillies auprès des acteurs de la SC pour alimenter le modèle de simulation. En outre, le développement d'un système dynamique « en ligne » pour la gestion des risques qui comprend un module pour chaque étape du processus de gestion de risques est aussi un objectif pertinent.

## TABLE OF CONTENTS

Acknowledgement .....	3
Remerciements.....	3
Résumé En Français.....	4
Introduction.....	21
Chapter 1: Literature review .....	26
Summary.....	26
Introduction.....	26
1.1 Literature review about supply chains analysis and modeling .....	27
1.1.1 Supply chain analysis methods .....	27
1.1.1.1 Descriptive methods.....	28
1.1.1.2 Quantitative methods .....	31
1.1.2 Modeling frameworks for simulation.....	37
1.2 Literature review on supply chain risk management.....	42
1.2.1 SC risk identification.....	46
1.2.2 SC Risk assessment.....	49
1.2.3 SC Risk treatment .....	56
1.2.4 SC Risk monitoring.....	58
1.3. Research questions .....	59
Conclusion .....	61
Chapter 2: The framework's development methodology .....	64
Summary.....	64
Introduction.....	64
2.1 Development of the modeling approach of the framework .....	66
2.1.1 SysML as a meta-modeling language .....	67
2.1.2 Designing the meta-model constructs .....	68
2.1.2.1 The SCOR reference model .....	69
2.1.2.2 Defining the risks modeling constructs .....	71
2.2 Development of translation guidelines and simulation modules.....	71
2.2.1 DES as simulation formalism for translating the conceptual constructs .....	72
2.2.2 ARENA as an example of a platform for translation .....	72
Conclusion .....	73
Chapter 3: The modeling framework: Creating the conceptual model.....	76
Summary.....	76
Introduction.....	76
3.1 Modeling supply chain's structure .....	77
3.1.1 The Actors' network view.....	78

3.1.2 The Product view .....	80
3.1.3 The infrastructure view .....	81
3.1.4. The transportation network view.....	82
3.2 Modeling the supply chain’s behavior .....	83
3.2.1 Representing flows.....	84
3.2.1.1 The material flows.....	85
3.2.1.2 The financial flows.....	85
3.2.1.3 The information flows .....	85
3.2.2 Representing the SC functions .....	88
3.2.2.1 The Operation block.....	88
3.2.2.2 The Library of supply chain Operations.....	89
3.2.3 Modeling through Roles.....	99
3.2.3.1 The Role block .....	99
3.2.3.2 The Roles library.....	99
3.2.4 Modeling the SC Processes .....	101
3.2.5 Illustrative example .....	102
3.3 Modeling the SC risks .....	104
Conclusion .....	106
Chapter4: A simulation framework: Creating and experimenting the simulation models .....	108
Summary .....	108
Introduction .....	108
4.1 The creation of the simulation models .....	109
4.1.1 Translation of the SC structure.....	110
4.1.1.1 Buffer construct translation example .....	110
4.1.1.2 The Resource construct translation example.....	111
4.1.2 Translation of the SC behavior.....	112
4.1.2.1 Translation of the Operation modeling constructs .....	112
4.1.2.2 Translation of the Flow modeling constructs .....	116
4.1.2.3 Translation of the Process modeling constructs .....	117
4.1.3 Translation of the SC risks .....	118
4.1.3.1 Translation of the propertyChangeRisk .....	119
4.1.3.2 Translation of the OperationModeRisk .....	120
4.1.3.3 Translation of the objectDestructionRisk .....	121
4.2 Experimentation of the simulation model .....	122
4.2.1 Define a scenario characterized by new policies.....	123
4.2.2 Define a scenario characterized by a different SC structure or network .....	123
4.2.3 Define a scenario characterized by risks .....	124

Conclusion .....	126
Chapter 5: Case study: Truck-Much supply chain .....	128
Summary .....	128
Introduction .....	128
5.1 Description of the case .....	128
5.2 Modeling the structure of the SC .....	130
5.2.1 Product view .....	130
5.2.2 Actor's network view .....	131
5.2.3 Infrastructure view .....	132
5.2.4 transportation network view .....	134
5.3 Modeling the behavior of the SC .....	136
5.3.1 Modeling the SC activities .....	136
5.3.2 Modeling the SC processes .....	138
5.4 Creating and verifying the simulation model .....	140
5.5 Simulating the model .....	147
Conclusion .....	158
Chapter 6: Conclusion.....	160
6.1 Resolved research questions and main findings.....	160
6.2 Limits Of SCOR.....	162
6.3 Limits On The Coverage Of The Proposal.....	162
6.4 Perspectives.....	163
Bibliography .....	164
A1 The library of domain specific operations .....	173
A1.1 The ISSUEMATERIAL Operation (sMi.2) .....	173
A1.2 The TEST Operation (sMi.3.2) .....	179
A1.3 The PICKANDPACK operation (C.sDi.9-sDi.10).....	185
A1.4 The LOADVEHICLE Operation (sDi.11 ) .....	192
A1.5 The SHIPPRODUCT Operation (sDi.12+A+sSRi.5).....	199
A1.6 The RECEIVEPRODUCT Operation (sSi.2+A+sDi.13+A+sDRi.3).....	202
A1.7 The VERIFY Operation(sSi.3+A+sDi.13+A+sDRi.3).....	208
A1.8 The TRANSFER Operation (sSi.4+A+sDRi.4+A+ sD1.8).....	214
A2 The library of Roles.....	224
A3 SC Risk countermeasures strategies proposed in literature .....	227
A4 Model of the SC case study .....	229
A4.1 Modeling the SC structure.....	229
A4.1.1 Modeling the Product view of the SC .....	229
A4.1.2 Modeling the Actor's network view of the SC .....	229

A4.1.3 Modeling the infrastructure view of the SC .....	231
A4.1.3.1 Modeling the SC Facilities .....	231
A4.1.3.2 Modeling THE SC Resources .....	232
A4.1.3.3 Modeling the SC Buffers.....	234
A4.1.4 Modeling the transportation view of the SC.....	236
A4.1.4.1 Modeling the SC Routes and Paths .....	236
A4.1.4.2 Modeling the SC TransportationResources and TransferResources .....	237
A4.2 Modeling the SC behavior.....	238
A4.2.1 Modeling the SC activities .....	238
A4.2.2 Modeling the SC processes .....	240

## LIST OF FIGURES

---

Figure 1.1: SCOR performance management process (sE2) (provided by SCC).....	29
Figure 1.2 : Example of a SC response matrix (Source (Alaca & Ceylan 2011)).....	30
Figure 1.3: Evolution of thhe number of simulation articles.....	34
Figure 1.4: Evolution Of the number of articles concerned with modeling frameworks for simulation .....	38
Figure 1.5: Evolution of the number of SC risk articles.....	43
Figure 2.1: The framework support to simulation based on SC risk analysis .....	65
Figure 2.2: The SysML diagrams (Adapted from omgSysml ).....	68
Figure 2.3: The description levels of SCs defined by SCOR .....	70
Figure 3.1: The framework to model the SC and the associated risks .....	77
Figure 3.2: SC Structure meta-model blocks definition diagram.....	78
Figure 3.3: The Actors' network view blocks definition diagram .....	79
Figure 3.4: The Contract Blocks definition diagram.....	80
Figure 3.5: The Product view block definition diagram.....	80
Figure 3.6: The infrastructure view block defenition diagram.....	81
Figure 3.7: Transportation network view blocks definition diagram .....	83
Figure 3.8: The behavior meta-model block definition diagram.....	84
Figure 3.9: The flows of the SCOR Process elements “sS1.4/sS2.4 TranferProduct” and “sD1.15/sD2.15 Invoice”.....	85
Figure 3.10: block definition diagram of information flows of type “Order” .....	87
Figure 3.11: Notification block definition diagram.....	87
Figure 3.12: The Operation interaction .....	88
Figure 3.13: The OperationMode block definition diagram.....	89
Figure 3.14: The Produce Operation block diagram .....	92
Figure 3.15: Details of the used inputs and outputs for the PRODUCE Operation .....	93
Figure 3.16: The PRODUCE Operation state machine.....	94
Figure 3.17: SC Roles block definition diagram.....	100
Figure 3.18: The Process block definition diagram.....	102
Figure 3.19: Trading goods process .....	103
Figure 3.20: SC risk meta-model block definition diagram .....	106
Figure 4.1: The framework support to simulate the SC and the associated risks .....	109
Figure 4.2: The translated Buffer construct.....	111
Figure 4.3: The translated Resource modeling construct .....	112
Figure 4.4: The ARENA dialog window for parametrizing the sMi.3.1 PRODUCE .....	113
Figure 4.5: Flowchart of the sMi.3.1PRODUCE Operation ARENA simulation module .....	114
Figure 4.6: Flowchart of the consumeComponents () method ARENA submodel .....	115
Figure 4.7: The template of the SCOR Operation simulation modules.....	116
Figure 4.8: The PurchaseOrder .....	117
Figure 4.9: The ARENA model of the trading goods process.....	118
Figure 4.10: The ARENA template for the Risk modules .....	119
Figure 4.11: Flowchart of the propertyChangeRisk module in ARENA .....	120
Figure 4.12: Flowchart of the operationModeRisk ARENA module.....	120
Figure 4.13: Flowchart of the ARENA ObjectDestructionRisk module relative to Resource objects .....	121
Figure 4.14: Flowchart of the ARENA ObjectDestructionRisk module relative to Resource objects.....	121
Figure 4.15: Flowchart of the ARENA ObjectDestructionRisk module relative to information flow objects....	122
Figure 4.16: The ARENA dialog window of the destructObjectRisk module relative to Flow objects.....	125

Figure 4.17: The ARENA model of the trading goods process including an instantiation of the destructObjectRisk module relative to Flow objects.....	125
Figure 5.1: Illustration of the Truck-Much Supply Chain.....	129
Figure 5.2: The object diagram of the products CGMV, Bd and Lv.....	131
Figure 5.3: A portion of the object diagram of the Actor’s network view of the structure model .....	132
Figure 5.4: The layout of the GTM facility.....	132
Figure 5.5: The Facility instance of the GTM factory.....	133
Figure 5.6: Example of a Resource instance belonging to the GTM facility .....	134
Figure 5.7: Example of a Buffer instance belonging to the GTM facility.....	134
Figure 5.8: Example of a Route instance linking GTM and DistC1 .....	135
Figure 5.9: Example of an instance of the TransportationResource used for shipping to DistC1 and DistC2 facilities.....	135
Figure 5.10: Example of a TransferResource’s instance linked with the used Path.....	136
Figure 5.11: The instances of the ROLE constructs relative to TRUCKMUCH.....	137
Figure 5.12: Instance of the SMI.3 PRODUCE OPERATION relative to TRUCKMUCH .....	137
Figure 5.13: The Activity diagram modeling the aggregated SC process .....	138
Figure 5.14 : The Activity diagram modeling the GTM sub-process.....	139
Figure 5.15: Example of translation of the sMi.3 Produce Operation instance into an ARENA module .....	142
Figure 5.16: An example of ARENA model where Operation patterns are connected.....	143
Figure 5.17: The ARENA model of the studied SC process .....	143
Figure 5.18: Temporal variation of GTM inventory levels after executing modules .....	146
Figure 5.19: Impacts on the RL2.2 %Delivery Performance To Customer Commit Date .....	155
Figure 5.20: Impacts on The Average Inventroy Per Day.....	155
Figure 5.21: Impacts on The RL1.2 % Of Orders Delivered In Full.....	156
Figure 5.22: Impacts on the Production Resource utilization .....	156
Figure A1.1: The ISSUEMATERIAL Operation block definition diagram .....	173
Figure A1.2: Details of the used inputs and outputs for the ISSUEMATERIAL Operation.....	175
Figure A1.3: State Machine Of The ISSUEMATERIAL Operation’s Standard Mode .....	176
Figure A1.4: The TEST Operation Block definition diagram.....	180
Figure A1.5: Details Of The Used Imputs And Outputs For The TEST Operation .....	181
Figure A1.6: The state machine of the algorithm of the standard Mode of the TEST Operation .....	182
Figure A1.7: The PICKANDPACK Operation Block definition diagram .....	186
Figure A1.8: Details Of The Used Imputs And Outputs For The PICK AND PACK Operation .....	187
Figure A1.9: The state machine of the algorithm of the PICKANDPACK Operation.....	188
Figure A1.10: The LOADVEHICLE Operation Block.....	192
Figure A1.11: Details of the used inputs and outputs for the LOADVEHICLE Operation .....	194
Figure A1.12: The state machine of the Algorithm of the LOADVEHICLE Operation.....	195
Figure A1.13: The SHIPPRODUCT Operation block Definition diagram .....	200
Figure A1.14: Details Of The Used Imputs And Outputs For The SHIPPRODUCT Operation .....	200
Figure A1.15: The state machine of the SHIPPRODUCT Operation .....	201
Figure A1.16: The RECEIVEPRODUCT Operation block Definition diagram.....	204
Figure A1.17: Details Of The Used Imputs And Outputs For The THE RECEIVEPRODUCT Operation.....	204
Figure A1.18: The state machine of the RECEIVEPRODUCT Operation .....	205
Figure A1.19: The VERIFY Operation block Definition diagram.....	210
Figure A1.20: Details Of The Used Imputs And Outputs For The VERIFY Operation .....	210
Figure A1.21: The state machine of the VERIFY Operation .....	211
Figure A1.22: TRANSFER Operation block Definition diagram.....	216
Figure A1.23: Details Of The Used Imputs And Outputs For The TRANSFER Operation .....	217



Figure A1.24: The state machine of the TRANSFER Operation .....	219
Figure A4.1: The Object diagram of The products CGMV, BD AND LV .....	229
Figure A4.2: The Instance Diagram Of The Contract Between TruckMuch And DC1 .....	229
Figure A4.3: The Instance Diagram of The Contract Between TruckMuch And DC2 .....	230
Figure A4.4: The Instance Diagram of The Contract Between TruckMuch And Supplier2 .....	230
Figure A4.5: The Instance Diagram of The Contract Between TruckMuch Supplier1 .....	230
Figure A4.6: The instance Diagram Of The GTM Facility .....	231
Figure A4.7: The instance Diagram Of The DistC1 Facility .....	231
Figure A4.8: The Instance Diagram Of The DISTC2 Facility .....	231
Figure A4.9: The instance Diagram Of The Supplier1 .....	232
Figure A4.10: The Instance Diagram Of The SUPPLIER2 .....	232
Figure A4.11: The resource instances diagram belonging to GTM Facility .....	232
Figure A4.12: The resource instances diagram belonging to DistC1 Facility .....	233
Figure A4.13: The Resource Instances Diagram Belonging To DISTC2 Facility .....	233
Figure A4.14: The resource instances diagram belonging to Supplier1 Facility .....	233
Figure A4.15: The resource instances diagram belonging to Supplier2 Facility .....	234
Figure A4.16: The Buffer instance belonging to GTM Facility .....	234
Figure A4.17: The Buffer instance belonging to DistC1 Facility .....	235
Figure A4.18: The Buffer instance belonging to DistC2 Facility .....	235
Figure A4.19: The Buffer instance belonging to Supplier1 Facility .....	235
Figure A4.20: The Buffer instance belonging to Supplier2 Facility .....	236
Figure A4.21: The SC Route instances .....	236
Figure A4.22: The Path instance belonging to GTM .....	236
Figure A4.23: The SC TransportationResource instances .....	237
Figure A4.24: The SC transferResource instances .....	238
Figure A4.25: The instances of the Roles constructs relative to TRUCKMUCH .....	238
Figure A4.26: The instances of the Roles constructs relative to Supplier1 .....	239
Figure A4.27: The instances of the roles constructs relative to Supplier2 .....	239
Figure A4.28: The instances of the roles constructs relative to DistC1 .....	240
Figure A4.29: The instances of the Roles constructs relative to DistC2 .....	240
Figure A4.30: The activity diagram modeling the aggregated SC Process of the case study .....	241
Figure A4.31: The activity diagram modeling the sub-process of GTM .....	242
Figure A4.32: The activity diagram modeling the sub-process of Supplier1 .....	243
Figure A4.33: The activity diagram modeling the sub-process of Supplier2 .....	243
Figure A4.34: The activity diagram modeling the sub-process of DistC1 .....	243
Figure A4.35: The activity diagram modeling the sub-process of DistC2 .....	244

## LIST OF TABLES

---

Table 1.1: Comparison of the three major simulation formalisms (interpreted from (Heath et al. 2011)).....	37
Table 1.2: Review of modeling frameworks for simulation.....	41
Table 1.3: Review of common SC risk definitions in the literature .....	44
Table 1.4: Some common literature’s definitions of SC risk management .....	45
Table 1.5: Common SC risk management process steps in literature .....	45
Table 1.6: Literature methods for SC risk identification.....	47
Table 1.7: SC risk categories in literature .....	48
Table 1.8: Advantages and limits of SC risk assessment models.....	52
Table 1.9: Review of simulation based risks assessment literature.....	55
Table 1.10 : Supply risk mitigation approaches and related strategies (proposed by (Chopra & ManMohan 2014)) .....	57
Table 1.11: Mitigation approach integration in simulation (proposed by (Talluri et al. 2013)).....	58
Table 2.1: Simulation software evaluation by SC practitioners (proposed by (Cimino et al. 2010)).....	73
Table 3.1: The library of Operations .....	91
Table 3.2 Retained inputs and outputs from SCOR for the PRODUCE Operation .....	92
Table 3.3: Internal variables used in the PRODUCE Operation algorithm.....	95
Table 3.4 : The receiveAndReleaseProductionOrders method .....	95
Table 3.5: The determineTheQuantityToProduce method .....	96
Table 3.6 : The reserveResource method .....	96
Table 3.7: The verifyComponentsAvailability method.....	97
Table 3.8: The consumeComponents method .....	97
Table 3.9: The AdjustTheManufacturedProductInventory Method .....	98
Table 3.10: The releaseResource method.....	98
Table 3.11: The notifyAboutExecution method.....	99
Table 3.12: Correspondence Between the Maker Role and the SCOR process elements .....	100
Table 3.13: Actors’ roles’ configuration .....	102
Table 3.14: Excerpt of Deliver operations relative to the manufacturer (D).....	103
Table 3.15: SC Risks Literature Crosschecked With The Proposed Risk Categories (Saleh Ebrahimi et al.(2012)) .....	105
Table 4.1: The translation of the behavior modeling constructs .....	112
Table 4.2: Declaration of a part of the internal variables of the Produce operation .....	113
Table 5.1: Example of the translation of some of the instances of Truck Much structure .....	141
Table 5.2: Resources settings .....	144
Table 5.3: Transportation Resources settings.....	144
Table 5.4: Transfer Resources settings.....	145
Table 5.5: Buffers settings .....	145
Table 5.6: Theoretical calculation of the outputs of the Issue module .....	147
Table 5.7: Adopted performances metrics .....	147
Table 5.8: Resources settings.....	149
Table 5.9: Transportation Resources settings.....	149
Table 5.10: Transfer Resources settings .....	150
Table 5.11: Buffers settings .....	150
Table 5.12: Demand Arrivals (Final client’s demands) .....	150
Table 5.13: risk experiments .....	151
Table 5.14: Results for the base scenario .....	152
Table 5.15: Results for the supply delay (R1).....	153

Table 5.16: Results for the supply cease (R2) .....	153
Table 5.17: Results for an error in the purchase order quantity (R3) .....	154
Table 5.18: Results for combination of an error for purchase order quantity and a supply cease (R4) .....	154
Table 5.19: Results for combination of a supply delay and a supply cease (R5) .....	155
Table A1.1: Retained inputs and outputs for the ISSUEMATERIAL Operation from the SCOR model.....	174
Table A1.2: The Internal Variables Of The ISSUEMATERIAL Operation algorithm.....	177
Table A1.3: Retained Inputs And Outputs From The SCOR Model For The TEST Operation.....	180
Table A1.4: Internal variables used in the algorithm of the TEST Operation.....	183
Table A1.5: Retained Inputs and Outputs From The SCOR Model For The PICKANDPACK Operation .....	186
Table A1.6: The internal variables used in the algorithm of the PICKANDPACK Operation .....	189
Table A1.7: SCOR retained inputs and outputs for the LOADVEHICLE Operation .....	193
Table A1.8: The internal variables of the algorithm of the standard Mode of the LOADVEHICLE OPERATION .....	196
Table A1.9: Retained Inputs and Outputs From The SCOR Model For The SHIPPRODUCT Operation .....	199
Table A1.10: Internal variables used in the algorithm of the standard mode of the SHIPPRODUCT Operation	201
Table A1.11: Retained Inputs and Outuputs From The SCOR Model For The Operation RECEIVEPRODUCT .....	203
Table A1.12: Internal variables used in the algorithm of the standard mode of The RECEIVEPRODUCT Operation.....	206
Table A1.13 : Retained inputs and outputs for the VERIFY Operation from the SCOR model .....	209
Table A1.14: Internal variables used in the algorithm of the standard mode of THE VERIFY Operation.....	211
Table A1.15 : Retained Inputs and Outuputs From The SCOR Model For The TRANSFER Operation .....	215
Table A1.16: Internal variables used in the algorithm of the standard mode of THE TRANSFER Operation...	220

## LIST OF ACRONYMS

---

SC: Supply Chain,

SCM: Supply Chain Management,

SCRM: Supply Chain Risk Management,

SCOR: Supply Chain Operations Reference,

ABS: Agent Based Simulation,

DES: Discrete Events Simulation,

SDS: System Dynamics Simulation,

SysML: Systems Modeling Language,

UML: Unified Modeling Language,

## LIST OF CONVENTIONS

---

**“ProcessElement”**: The names of the SCOR process elements are given in bold characters between inverted commas.

OPERATION: The name of each operation we propose for the library is put in uppercase.

*PropertyNames*: The properties names of the proposed blocks are put in italic letters.

BlockName: The SysML block names have always the first letter in uppercase.

# INTRODUCTION

The supply chains (SC) keep evolving over the years. They change in configuration, in size, in geographic extent and in the way they are managed. New kinds of SCs appear thanks to technological development and globalization. The emergence of a new product with new characteristics changes the SC configuration. For instance, Dell modular computers are partially assembled by users. The development of recycling technology resulted in the appearance of reverse SCs. The development of safe electronic payment platforms opened the way for cyber supply chains such as Amazon. Supply chains may also disappear due to a lack of demand, such as the disc storage technology SCs. The first SCs analysis studies appeared with consultants who wanted to communicate the need to develop better ways to manage resources and assets. As revealed by (Ellram & Cooper 2014) the first definition for SC management was primarily written more than 30 years ago, first appearing in the practitioner literature in 1982. This uncovered to academicians the need to develop solutions for this issue. The debate about SC management is still open and still growing for academia to keep with the development of SCs. As revealed by (Ellram & Cooper 2014) the Wall Street Journal recently reported that more universities are adding SCM majors and increasing their programs as demand for supply chain management (SCM) majors grows among employers.

For a long time, the focus of SCM was on improving the cost efficiency of SCs as stated by (Christopher & Lee 2004). Many approaches were developed which are concerned with reducing the cost across the entire supply chain and giving companies the opportunity to better compete against other players in the market as stated by (Manuj et al. 2008). SC practitioners make a lot of effort on the implementation of cost effective management techniques. An example of such approaches is Lean management, developed by Toyota in Japan. The wide adoption of these approaches brought more challenges. As stated by many researchers (See for instance, (Enyinda et al. 2008), (Pfohl et al. 2013);(Tuncel & Alpan 2010)) the potential benefits in the shape of decreased inventory levels, shorter lead times, minimal delays and material buffers have made supply chains more vulnerable to local and global disturbances.

The vulnerability is inherent to the dynamic development of SCs and their increased complexity. As revealed by (Jüttner 2005) through their exploratory study, 44% of the responding companies expect the vulnerability of their supply chains to increase within the next five years. (Simchi-levi et al. 2015) provide a set of factors that increases the operational vulnerability of SCs in automotive industry. Among the provided factors, we cite the measures taken by companies to maximize the operational effectiveness. These measures result in more dependency to more concentrated suppliers. Another factor stated by the authors is the company measures for decreasing supply cost through only concentrating on the sources that provide more fiscal incentives and that are more capable of decreasing their products costs. This pushed suppliers to constraint their production capacity and to outsource in emerging unstable markets. Another cited factor is the lack of standardization in products that makes the manufacturing capability concentrated in few suppliers.

(Thun & Hoenig 2011) explain how outsourcing and offshoring increase SC vulnerabilities. They state that outsourcing raises the amount of interfaces and the dependency between companies and the offshoring increases the complexity and the exposition to failures of cross-national connections. Other vulnerability factors were highlighted by (Trkman & McCormack 2009) such as market and technological turbulences. The market turbulence arises from the

heterogeneity and the rapid changes in the composition of customers and their preferences, while the technological turbulence refers to the degree to which technology changes over time within an industry and the effects of those changes on the industry.

In the last years, many supply chains were subject to disruptions and witnessed negative impacts on their performance. According to a study made by (Simchi Levi et al. 2013) on a sample of 209 companies with a global footprint, disruptions incurred negative impact on the business financial performance of many companies. In fact, 54% of the companies said that sales revenue was negatively affected and 64% of them suffered a decline in their customer service levels. Across all the operational KPIs examined, at least 60% of the enterprises reported a 3% or higher loss of value. Furthermore, based on their exploratory quantitative survey, (Hendricks & Singhal 2005) conclude that firms do not recover quickly from the negative effects of disruptions. A famous example of a SC disruption that highly impacted its relative SC is the Ericsson case. As described by (Norrman & Jansson 2004) a lightning bolt hit an electric line in New Mexico which caused a fire at a production “clean rooms” cell of Ericsson’s supplier plant. This fire destroyed the production cell equipments and interrupted the shipment of radio-frequency chips to Ericsson. Since Ericsson had only one supplier of this kind of chips, the company lost its capability to sell and deliver one of its key consumers during its booming “market window”. Many months of mobile phone production were lost which pushed Ericsson’s to decide to withdraw from the mobile phone business. The cost of this supply disruption was calculated as approximately \$200 million.

Supply chain managers encounter failures in managing their risks. As revealed by (Hult & Craighead 2010) companies like Boeing, Cisco, and Pfizer encountered unexpected losses and/or expenses of more than \$2 billion due to ineffective supply chain risk management (SCRM) decisions in 2001. (Chopra & ManMohan 2014) state that surveys have shown that managers do little to prevent incidents since the solutions to reduce risks are not weighed against SC cost efficiency. Also, a recent study by (Marchese & Paramasivam 2013) from Deloitte consulting firm on 600 Supply Chains and top executives revealed that many companies do not master SC risk management. In fact, only 33% used risk management approaches to proactively and strategically manage supply chain risks based on their operating environment conditions and 45% felt that their risk management was only somewhat effective or not effective at all. Many reasons can explain this. The major reason is the low implementation degree of the instruments of supply chain risk management (Jüttner 2005). Even if they are implemented many companies have an immature risk management process. In fact, as revealed by a the study of (Simchi Levi et al. 2013) 59 % out of the investigated companies have immature processes in place to effectively address incidents. Their SCRM processes are neither proactive nor flexible. Another reason is the fact that companies misunderstand SC risk management. As discussed by (Jüttner 2005) SCRM is still understood in many industries primarily as a company-specific task, or companies have not only to focus on their risks but also the risks of their partners. This is what makes SCRM a more difficult task, since dealing with an individual risk and forgetting the inter-connections can end up exacerbating another as stated by (Chopra & ManMohan 2014). Authors argue that actions taken by any company in the supply-chain can increase the risk for any other participating company. Authors highlight that another failure of SCRM in companies is the consideration of recurrent, low-impact risks while ignoring high-impact, low-likelihood risks.



Recently encountered SC failures such as the ones reported above brought the issue of SCRM to the forefront. The awareness about having effective SCRM processes increases every day within the industry. James Steele, the program director of SCRM of CISCO, the global information, and communication Technology Company, explained, in an interview published by (U.S. Resilience Project 2011), how his company's perception of SCRM evolved. He said: "In the past, supply chain operations were "cared-about" only when things went wrong. The focus was not on increasing the business, but on keeping the trains running on time. Over the past 15 years, there has been a sea of change in supply chain management. It has become a strategic capability for many companies, and it continues to get the resources, visibility and focus needed to manage as a platform for growth. For Cisco, this "change" has meant an increase in risk intelligence and agility on supply chain resiliency capabilities, which are a key element in this evolution". Similarly, based on their empirical study of 142 French companies' managers, (Lavastre et al. 2012) suggest integrating SCRM as a management function that is inter-organizational in nature and closely related to strategic and operational realities of the activity in question.

SC practitioners need to be assisted for improving their SCRM. Even if many risk management approaches exist and are part of the toolbox of managers they still do not cover all requirements.

The main issue treated in this Ph.D. is assisting the SC practitioners using simulation for analyzing the risks threatening their SCs, through promoting a quicker and easier construction of simulation models and through enabling risk scenarios' experimentation.

We will conduct an analysis of the relevant literature for identifying why the available tools do not satisfy needs of the SC practitioners. More precisely, we will investigate why the usage of simulation for risk analysis is still modest and what are the difficulties to overcome. Hence, we start by investigating the current analysis methods for SCs with a focus on the frameworks proposed for simulation. We identify a set of requirements for an easy to use and an effective framework. Then we investigate the particularities of the SC risk management domain with a focus on the SC risk analysis methods.

The main proposition of this thesis is a framework for modeling and simulating risks in SCs. The framework integrates a metamodel and modelling elements libraries developed with SysML to represent SCs. It is associated with a translation guideline enabling the construction of simulation models using the defined metamodel.

The work is documented in this dissertation as follows:

In the first chapter we provide a literature review about SC analysis, modeling and risk management, we identify the literature gaps and we cite the main research questions resolved in this dissertation. In the second chapter, we describe the adopted methodology for the development of the framework. In the third chapter, we introduce the part of the framework enabling the creation of conceptual models for SC. In the fourth chapter we introduce the simulation framework and the methodology for translating the conceptual model into a simulation model and for experimenting risk scenarios. The fifth chapter presents a case study exemplifying the deployment of the proposed approach. In the last chapter, we summarize findings and we discuss perspectives.

# CHAPTER 1

## LITERATURE REVIEW

## CHAPTER 1: LITERATURE REVIEW

### SUMMARY

When dealing with the analysis of risks in SCs, two major fields are called. The first one is SC analysis and the second one is SC risk management.

**W**In this chapter we provide a state of the art about those two fields. Namely, we investigate the various methods used for general purpose SC analysis and more specifically for analyzing risks. We provide a snapshot of the current developments of the two fields and we highlight the literature gaps.

Concerning SC analysis, we show that modeling the SC is of prime importance to enable catching its complexity and that the selected modeling method may restrict the reachable analysis results. We review descriptive methods such as the SCOR reference model and quantitative methods that are often specialized for optimization or simulation. Descriptive approaches are easy to handle for SC practitioners but provide poor analysis features while quantitative approaches give interesting analysis results but at a high appropriation cost. We sketch an opportunity to merge modeling approach based on descriptive principles with quantitative modeling to obtain tangible results. Namely, we show that performing simulation of SC with discrete event simulation techniques is particularly adapted. Nevertheless, the literature shows that these simulation techniques are costly to set up and that a structured modeling approach may be of interest.

The second part of this chapter discusses the treatment of risks within SC. Based on the literature analysis we show that the concept of risk has been tackled with various visions in the past. Several taxonomies of risks threatening SC have been used, each one implying a way of regarding risk (for example focus on risk perimeter, origin or magnitude). Therefore, to clarify our purpose, we present the retained definition of risk and precise the vision of the risk analysis process for SCs. We discuss and adopt a risk classification based on risk impacts and oriented to SC simulation.

The chapter is presenting our roadmap to contribute to SC risk management, namely to support the deployment of simulation approaches within the risk assessment phase of the risk management process.

### INTRODUCTION

This first chapter of the dissertation presents the state of the art on the current method and tools utilized for SC analysis. Through this review, we want to highlight the current tendencies on SC modeling, SC analysis and to highlight the problems addressed. SC analysis is very vast, therefore, we give an overview on research on SC and make focuses on specific areas to which we want to contribute. These topics are on modeling and simulating SCs for risk analysis.

In section 1.1, we present the current techniques developed for SC analysis, namely, we investigate the most popular approaches for modeling SCs. We present the descriptive methods aiming at describing the flows, the stakeholders and the relationships existing on the SCs. We review, for example, SCOR model and Value Stream Mapping. We also present

quantitative approaches such as optimization methods and simulation. We discuss the pros and cons of each technique and then we put a stress on simulation frameworks as it is one priority of our contribution. By doing so, we want to rise important requirements for proposing a valuable modeling and simulation framework for SC analysis.

Section 1.1 places the general concepts used for SC study. We then explore the specificities of risk analysis. In section 1.2, we detail the field of SC Risk Management (SCRM). We first propose some framing definitions based on literature analysis. Then, we detail SCRM process through the 4 phases: risk identification, risk assessment, risk treatment and risk monitoring. When analyzing the risk identification literature, we propose the risk categorization that is adopted in this dissertation. Finally, in section 1.3, we position our work on the global map of SCRM and propose our research objectives on the basis of the given literature review.

## 1.1 LITERATURE REVIEW ABOUT SUPPLY CHAINS ANALYSIS AND MODELING

SC analysis is an important task that most of the companies have to conduct. It has the following goals: identify weaknesses and strengths of the current SC (prioritize markets, prioritize products, etc.) and predict their future evolutions, evaluate the various improvement possibilities, define best parameters and configurations (required capacity, inventory security level, inventory replenishment level, best partners, etc.). Supply chain analysis is a well-studied subject in the literature. In this section, a literature review on supply chain analysis is conducted. We investigate the different methods used to analyze the SC, from descriptive methods to quantitative methods

### 1.1.1 SUPPLY CHAIN ANALYSIS METHODS

SC analysis is the group of tasks that aim to understand and evaluate SCs. We call SC a network of organizations that are involved, through upstream and downstream linkages, in the different processes and activities that produce value in the form of products and services in the hands of the ultimate consumer, as defined by (Christopher & Lee 2004). The SC analysis is a prerequisite for SC management. It helps to understand the relations between the SC elements and to identify the way by which the parameters impacting the SC performance need to be modified in order to achieve goals. (Bullinger & Kühner 2010) state that a profound and continuous analysis of the entire SC is necessary to achieve SC excellence. The authors argue that a suitable SC analysis needs to include the definition of performance units, the measurement of holistic performance with the ability to drill-down results and the interpretation of results in term of performance. An important task widely integrated into SC analysis methods is to model the studied system. In fact, as stated by (Bullinger & Kühner 2010) the major research and development activities in the area of supply chain analysis have resulted in modeling concepts. The purpose of modeling is to understand, analyze, and hopefully solve the problems that might appear in the problem domains as stated by (Kasi 2005). Modeling enables better SC decisions by helping firms to highlight the synergy of inter-functional and inter-organizational integration and coordination across the supply chain (Min & Zhou 2002).

We identify two categories of methods used for SC analysis:

- **Descriptive methods:** They propose the way to collect information about a set of metrics and the way to evaluate their evolution. The metrics might be textual or quantitative. Usually, the methods integrate calculation formulas and provide suggestions and best practices in order to react against a given evolution.
- **Quantitative methods:** They propose a way to design a model to capture the dynamics of the SC and the way to analyze it. They often result in mathematical or computational models. Unlike the descriptive methods, the resulting models can be “executed” or “resolved” in order to evaluate the performance of the SC.

#### 1.1.1.1 DESCRIPTIVE METHODS

Various descriptive methods are proposed by researchers and are used by SC practitioners. A set of these methods provides textual taxonomies to describe the SC parts (A generic description of SC domain knowledge), in order to facilitate modeling, analysis and performance evaluation. Examples of these methods are supply chain operations reference (SCOR), value reference model (VRM) and value stream mapping (VSM). The methods generally integrate both qualitative and quantitative prescriptions. The qualitative prescriptions address the way to describe the performed functions, to select the performance measures, to gather related information and to analyze them. The quantitative prescriptions address the way that some of the metrics need to be calculated. The calculation is usually simple (e.g., SCOR provides the formula “[Total Perfect Orders] / [Total Number of Orders]” to calculate the “RL1.1<sup>1</sup> Perfect Order Fulfillment” performance metric).

#### SUPPLY CHAIN OPERATIONS REFERENCE (SCOR)

---

A well-known descriptive method is SCOR (Supply chain Council (2012)). The SCOR model provides a framework for measuring the performance of the SC at different levels: From top to bottom, starting with a business process to end with the SC process elements or operations. SCOR provides performance measures (e.g. Return on Working Capital) that enable linking strategic objectives of SC to operational ones (e.g. Produce and test cycle time). The framework is based on a generic description of SC operations that start from the process (plan, deliver, make, source and return) to end up with subprocess elements. This is to permit comparability (to compare different supply chains and different supply chain strategies) and root cause analysis (e.g., to find the root cause of a degraded value of Perfect Order Fulfillment metric). The framework supports the design of the SC by providing a set of best practices mapped into the process elements. For example, the “Perfect Order Fulfillment” metric provides a good indication on how well every facet of a supply chain (planning, sourcing, manufacturing, and delivery) are tuned and coordinated to meet customer demand. Furthermore, the SCOR model contains the Perfect Order Fulfillment metric definition, calculation methods, and best practices. Managers can implement one of the proposed best practices that fits with the studied gap for correction.

SCOR provides a three-step process that describes how performance management needs to be handled: Performance measurement, performances analysis, and improvement. In each of

---

<sup>1</sup> Reference used in SCOR for the Perfect Order Fulfillment.

these steps, the manager needs to use concepts provided by SCOR such as SCOR metrics. This process is shown in figure 1.1. (Bullinger & Kühner 2010) propose an approach based on SCOR performance metrics. This one incorporates a measurement methodology integrating bottom-up and top-down performance measures as a hybrid balanced measurement approach. The measurement approach integrates SCOR metrics into the proposed supply network scorecards to form an integrated measurement system. Different stages of the SCM activities, as well as different perspectives on the value creation process are covered by the measurement system by following the principles of a balanced measurement method introduced in (Sellitto et al. 2015).

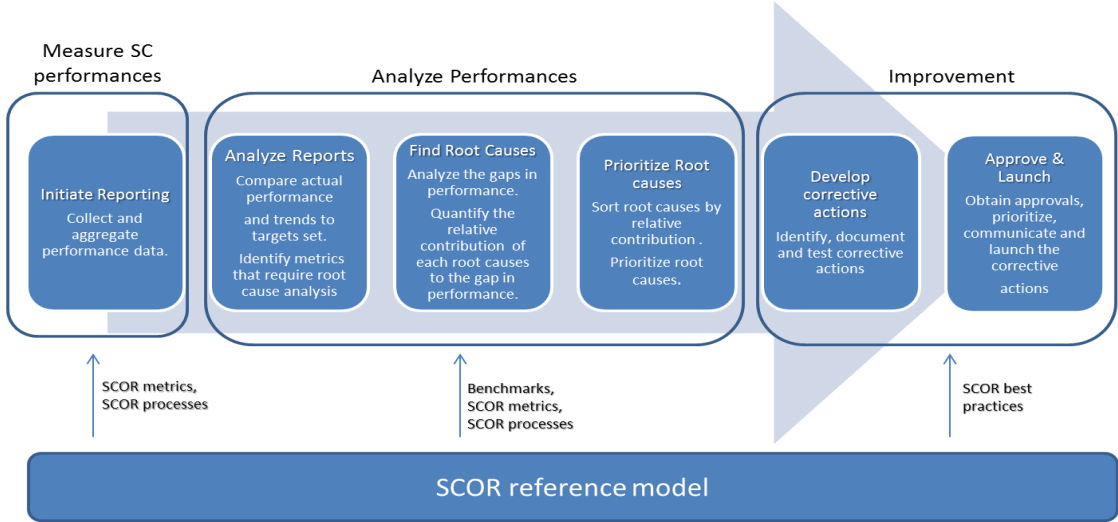


FIGURE 1.1: SCOR PERFORMANCE MANAGEMENT PROCESS (SE2) (PROVIDED BY SCC)

**VALUE STREAM MAPPING (VSM)**

Another popular and highly employed method is the Value stream mapping (VSM). VSM is a lean-management method that may be used to analyze the current state of a SC and to design a future state described as a value stream. A value stream is an end-to-end set of activities that are collectively valuable to a customer as stated by (Brown 2009). The customer may be the ultimate, external customer or an internal user of the value stream. As stated by (Hines & Rich 1997), the difference between the traditional supply or value chain and the value stream is that the former includes the whole activities of all the companies involved, whereas the latter refers only to the specific parts of the firms that actually add value to the specific product or service under consideration. The SC response matrix, one of the famous VSM tools, is based on a mapping approach that seeks to portray in a simple diagram the critical lead-time constraints, as stated by Taylor et al. (2001). The tool permits the analysis of the relation between lead times and the inventory level in different steps of the SC. Figure 1.2 provides an illustration of this tool. The horizontal axis in response matrix represents a cumulative lead-time for the operations plan and transfer in the supply chain. The vertical axis represents cumulative inventory in days in every stage of the supply chain.

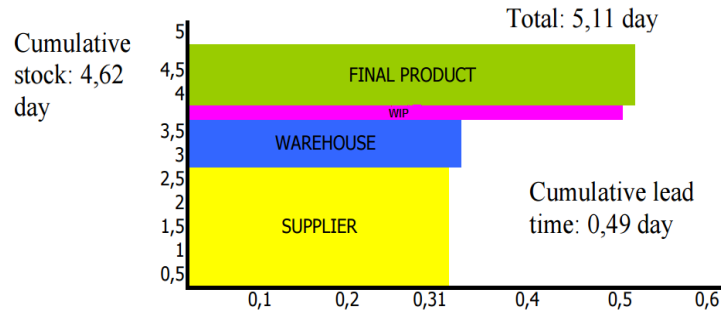


FIGURE 1.2 : EXAMPLE OF A SC RESPONSE MATRIX (SOURCE (Alaca & Ceylan 2011))

## OTHER DESCRIPTIVE METHODS

Other descriptive methods are proposed in the literature. Some of them propose an approach similar to the SCOR model. Namely, the VRM approach defines a textual description of the operations performed within the SC (Brown 2009). Some methods focus on the procedure of performance metrics measurement and analysis (for example see (Cai et al. 2009) while others focus on facilitating building SC descriptive models (for example see (Kim & Rogers 2005)). For instance, (Cai et al. 2009) propose a systematic approach for measuring the performance of SCs that integrates a descriptive part consisting of the definition of the relationships among SC metrics. The description of the intricate relationships among SC KPI enables analyzing the deviations leading to the non-achievement of SC goals.

Another descriptive method is the one proposed by (Kim & Rogers 2005). The method defines a procedure for building SC object-oriented models using the unified modeling language (UML). The procedure includes five steps to define different views of the model to be built (Vision view (defines visions and goals), function view (functional domains and functional requirements), process view (defines business processes), structure/static view (defines resources and organization), behavior/dynamic view (interaction analysis)) using a set of classes such as the process class. Authors propose two steps to integrate business rules into the model as follows: Identifying business rules and expressing them in the object-oriented model.

## SUMMARY OF DESCRIPTIVE METHODS

Thanks to the literature review of the descriptive analysis methods, we find that these methods focus on three aspects:

- First, providing a detailed description of the SC components with the associated performance metrics (See, for example, SCOR, VRM, and VSM).
- Second, providing a way to measure the performance metrics and the way to analyze them (See, for example, (Cai et al. 2009)).
- Third, providing a procedure to build SC models (see, for example, (Kim & Rogers 2005)).

The first aspect has the benefit of enhancing the understanding of SC managers about their SCs through clarifying the various functions and also has the benefit of providing consistent performance metrics for measuring the SC attributes. In fact, the taxonomy proposed by SCOR provides SC practitioners with a common framework that can easily be used to compare and communicate about the SC measured performances. SCOR is stated (Albores et



al. 2006) to become a *de facto* standard that has gained considerable popularity with practitioners.

The second aspect has the benefit of improving the accuracy of the way by which the manager defines metrics, measures their values and interprets their evolution.

The third aspect has the benefit of enhancing the understanding of the modeled system. In fact, as stated by (Robinson 2004) 50% of the benefits of analysis can be obtained only by modeling the system, since, while modeling, modelers gain a thorough understanding of the system.

#### 1.1.1.2 QUANTITATIVE METHODS

The second type of SC analysis methods is the quantitative methods. They are based on the analysis of the outputs of the computational experiments for a model describing the SC dynamics. Different from the descriptive methods, the quantitative methods provide a model that does not only serve to enhance understanding about the SC and measure current performance but may also serve to make computational experiments to have quantitative results. We distinguish two main types of quantitative models widely used in literature. They are the optimization models and the simulation models. Other models exist but more specific for some SC functions and are not discussed here (e.g. forecasting models). In next sections, we investigate the two main types of quantitative models (optimization and simulation) with a focus on simulation models since they are adopted in this Ph.D. work. Optimization models are used to solve a given decision problem. They have been well studied in operations research over the last 50 years and they have been extensively used in SC modeling and analysis. We do not have the ambition to review this extensive literature in detail in this thesis since there are already numerous comprehensive literature reviews on this issue. Recent literature reviews focus on a given current prominent field rather than tackling the use of optimization models in SC analysis in general. Hence, as stated by (Asgari et al. 2016) the current prominent fields are risk management, sustainability, and globalization. (Fahimnia et al. 2015) provide a literature review of the quantitative models used for SC risk management. (Snyder et al. 2016) provide a literature review of the models used for disruption modeling. They organized the reviewed works into six categories: evaluating supply disruptions, strategic decisions, sourcing decisions, contracts and incentives, inventory and facility location. (Seuring 2013) provides a literature review of modeling approaches for sustainable SC management. The author highlights a weak line among papers using multi-objective programming which is the focus on a single company/supply chain. Also, (Brandenburg et al. 2014) provide a literature review on the usage of quantitative models for sustainable SC management. The authors state that managerial decision-making is often supported by optimization methods. (Matinrad et al. 2013) highlight some of the trends in SC network modeling such as the increased consideration for uncertainty and multi-echelon/stage supply chains, while a decreased interest for multi-period modeling.

To explain how optimization models are used for SC analysis, we present some examples of use in SC risk analysis since this field is the focus of our Ph.D. work. In fact, (Sawik 2015) propose a stochastic MIP model for a multi-stage supply chain under disruption risks. The



networks are subject to independent random local disruptions of each supplier individually and to global disruptions of all suppliers simultaneously. The model integrates supplier selection, order quantity allocation, and customer orders scheduling. The objective is to have risk neutral and risk adverse solutions that minimize, respectively, expected cost and expected worst-case cost. (Hasani & Khosrojerdi 2016) propose a mixed integer, a nonlinear model for the design of a robust supply chain network under uncertainty. The considered uncertain parameters are the customers' demands and the part procurement cost. The effect of some flexible and resilience strategies is investigated via sensitivity analysis. A robust optimization based on the uncertainty budget approach is considered.

Optimization models have advantages in dealing with particular problems where the objectives are well defined. In fact, they are effective in determining best SC parameters that optimize a given function while respecting system constraints. Nevertheless, optimization models have some limits: Optimization models for large systems are difficult to build. The design of a given model has to consider the resolution capability. In fact, the complexity of the model influences the resolution time and the used memory capacity. Modelers have to limit the number of variables and constraints and consequently the perimeter of the study. Since a whole SC is too complex to be described by mathematical equations, most of the described literature works are forced to make simplifications (Wan et al. 2005). Furthermore, it is difficult to create an optimization model that captures at the same time different configurations of the same system. Finally, even though the optimization models have the ability to integrate stochastic parameters and to model uncertainties, stochastic optimization models are difficult to manipulate and to resolve for large systems.

## **SIMULATION MODELS**

---

The simulation models are used to emulate the real dynamics of the SC over time. The registered results of the executed emulation enable analyzing the behavior of SCs facing various conditions.

Simulation models have advantages compared to some of the limitations of optimization models. In fact, as stated by (Wan et al. 2005), compared to other methods, simulation provides the flexibility to accommodate arbitrary stochastic elements and generally allows modeling of all of the complexities and dynamics of real-world supply chains without undue simplifying assumptions. Furthermore, (Pirard et al. 2008) highlight that simulation permits the integration of policies (e.g. inventory control, production policy, production order's assignment) easier than the optimization. Also, simulation may provide a better understanding of how supply chain attributes influence the behavior of the whole chain and how the attributes interact including the stochastic behavior as stated by (Longo & Mirabelli 2008). Simulation is used to provide insights about how some causes and effects relationships impacts supply chain performances.

Some of the works provide a literature review about the use of simulation for SC studies. For instance, (Jahangirian et al. 2010) propose a literature review of the use of simulation in manufacturing and business application that covers the period from 1997 to 2006. The authors highlight an interesting finding: despite that discrete event simulation (DES) is the most popular formalism it attracts less attention from stakeholders. The authors explained this by

the difficulty and the time needed for data gathering and modeling. Furthermore, authors highlight an increased interest in hybrid simulation.

Simulation has long been used for different kinds of SC analysis studies. It has been widely used for operations management, logistics and supply chain management ((Shafer & Smunt 2004); (Terzi & Cavalieri 2004); (Kleijnen 2005); (Evers & Wan 2012)) for order release mechanisms evaluation, (Chan et al. 2002) for evaluating the design and performance of business process and inventory control parameters, (Jain et al. 2001) for uncertainty impact analysis (Petrovic 2001) and for SC risk analysis ((Tuncel & Alpan 2010) , (Schmitt & Singh 2012), (Talluri et al. 2013)).

Different simulation formalisms have been developed in the last years. (Kleijnen & Smits 2003) provide a categorization of supply chain simulation formalisms. They are as follows:

- Spreadsheet simulation: Refers to the use of a spreadsheet to represent the model, do the sampling and perform experiments. A spreadsheet has a table structure that permits the organization of calculations and results. The spreadsheet has four important limitations as stated by (Seila 2004): (1) Only simple data structures are available, (2) complex algorithms are difficult to implement, (3) spreadsheets are slower than some alternatives and (4) data storage is limited.
- Business games: The simulation process integrates interaction with a set of players. A player has the ability to redefine the simulation rules and current state. They are used for training purposes.
- System dynamics simulation (SDS): It is based on the representation of system structure in terms of stocks and flows where the change occurs continuously over time. It was developed by Forrester during the 1950's (Forrester 1968).
- Discrete event simulation (DES): is based on the representation of a system as a network of queues and activities where state changes occur at discrete points of time.

There is another simulation formalism widely adopted that is not mentioned by Kleijnen et al. (2003) which is the agent-based simulation.

- Agent-based simulation (ABS): It is based on the representation of a system as a set of individual, autonomous, interacting agents. The global behavior of the system is the result of the interaction between the behaviors of many agents.

The research community that deals with simulation of supply chains, manufacturing, and production systems provide more interest to DES than other simulation formalisms as shown in figure 1.3. Figure 1.3 shows the evolution of the number of articles mentioning a given simulation formalism in the paper title. The number of articles is determined for every couple of years between 1994 and 2015. Those numbers are found thanks to "Google scholars" research engine. The Boolean logical operators (AND/OR) are used to combine keywords and to refine the research. The keywords used for the search are as follows:

- For the discrete event simulation: simulation, discrete event(s), Petri net(s), ARENA, manufacturing, production, and supply chain.
- For the agent-based simulation: agent, system dynamic(s), manufacturing production, and supply chain.

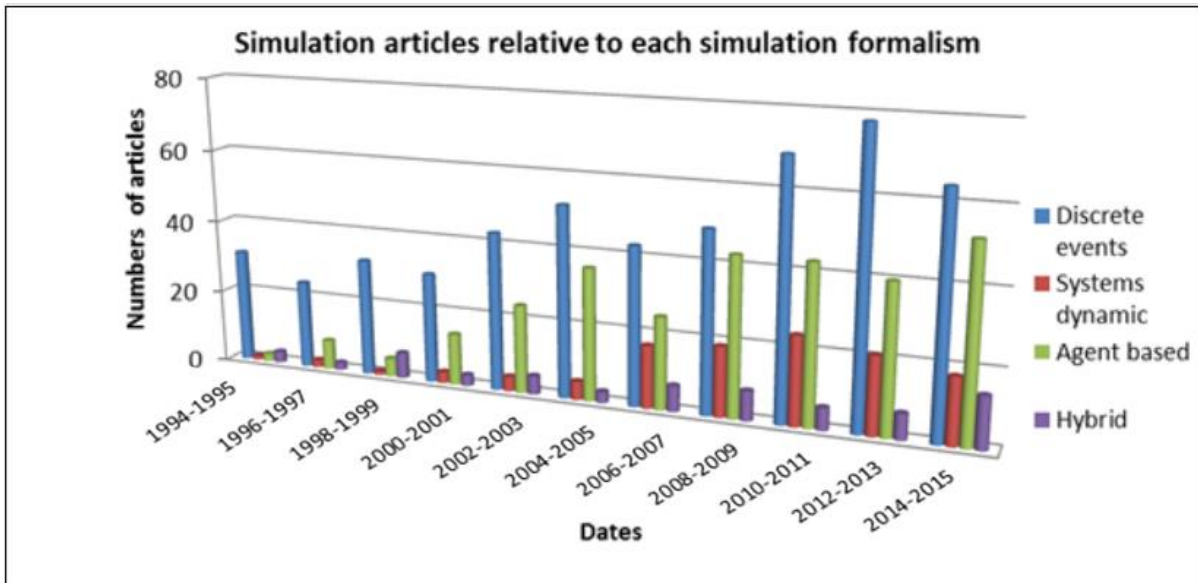


FIGURE 1.3: EVOLUTION OF THE NUMBER OF SIMULATION ARTICLES

- For the system dynamics simulation: simulation, system dynamic(s), manufacturing, production, and supply chain.
- And finally, for hybrid simulation: hybrid simulation, system dynamic(s), manufacturing, production, and supply chain.

This important interest given to DES can be explained by the fact that most of the operations within the SC are discrete in nature and that simulation studies focus more on tactical and operational decision level. SDS took less interest and has the lowest publication rate over years. ABS is the second most used formalism in the literature. Historically DES is the most important simulation formalism but since 2003 ABS becomes a strong competitor of DES. This can be explained by the fact that simulation technology becomes more mature and more available for research studies. The hybrid simulation, which is a combination of different simulation formalisms, is the less used in literature studies and witnessed less expansion. This can be explained by the fact that research simulation studies do not tackle more than one decision level at once and that the technology of use is less available and less developed.

In the next section, we investigate major simulation formalisms, we highlight the field of their use and the specificities of their use illustrated with some examples.

### SYSTEM DYNAMICS SIMULATION (SDS)

SDS aims to capture how organizational structure, policy variables, and time delays (in decisions and actions) interact to influence the performance of companies. The system dynamics' logic is based on the representation of system structure in terms of stocks and flows, which measure the accumulation and dissipation of material or information over a period of time. Feedback loops serve as building blocks for expressing the relationships between the variables and overall dynamic behavior of complex interdependencies on the system. Feedback loops are connected to stocks and flows. System dynamics was first developed by Jay Forrester during the 1950s to model large scale systems (Forrester 1968). It enables taking into account complex interdependencies between causes and effects and rejects

the simple linear representation of "cause" and "effect" since the "effect" might also affect the "cause" (Sterman 2000).

In the supply chain management discipline, SDS has been used to deal with inventory planning/ management, bullwhip effects and information sharing as stated by (Tako & Robinson 2012) in their literature review (See for example, (Pirard et al. 2008), (Ge et al. 2004), (Janamanchi & Burns 2007), (Campuzano, F., & Bru 2011) and (Peng et al. 2014)). Some recent works use SDS for SC risk analysis ( see, (Li 2013), (Guertler & Spinler 2015), (Langroodi & Amiri 2016) and (Udenio et al. 2015) ).

SDS has some advantages in the analysis and the redesign of supply chain models that exhibit non-linearity. This is due to the simplicity of the data required, ease of building a simulation model and reduced execution time as stated by (Tako & Robinson 2012). Recent works on system dynamics are looking to enlarge the scope of application of this one. (Saleh et al. 2010) propose a method to build simplified and linearized models of nonlinear complex supply chain systems.

SDS has some limits. In fact, SDS is stated to be limited to operational problems that tend to be described as a discrete process rather than continuous. Furthermore, as stated by (Sumari & Ibrahim 2013) building a model for a big system may become too complex and may include many errors since it is not an easy task to identify the various relations.

#### AGENT-BASED SIMULATION (ABS)

Agent-based simulation is a powerful technique that has been developed recently. This formalism provides a particular way to model and simulate systems as a set of interacting autonomous entities called agents. ABS gained attention in the early 1990s in the fields of social and economic sciences, game theory, artificial intelligence and cognitive science. By the mid-1990 ABS became more popular, due to the publication of the defining work of *Growing Artificial Societies* by (Epstein & Axtell 1996) and also thanks to the release of the Swarm simulation system by (Minaer et al. 1996). Since then, the domain of application is extended to many fields including the supply chain analysis.

ABS has some advantages for SC analysis. This is due to the fact that the supply chain is formed naturally by a set of interacting actors and functions. ABS captures the emergent behavior resulting from the interaction of multiple groups of entities. The modeling process is intuitive since the modeling concepts are similar to SC real world elements as stated by (Long 2014). Furthermore, ABS facilitates distributed simulation.

ABS has some limits. There is a lack of ABS tools adaptable for SC studies. In fact, as stated by (Long 2014) the current ABS platforms (e.g. Repast developed by Social Science Research Computing at the University of Chicago and Swarm developed by the Swarm Development Group (SDG) are difficult to use for constructing SC simulation models. Furthermore, ABS does not have its proper simulation language to define the behavior of agents. This increases the complexity of modeling the system units as agents by respecting the common agent structure. (Chatfield et al. 2007) highlight another difficulty when using ABS for supply chain studies which are the order-driven nature of SCs, which is hardly captured by ABS. Furthermore, (Sumari & Ibrahim 2013) state that ABS requires high skills for computation when used for large systems.

## DISCRETE EVENT SIMULATION (DES)

---

DES is a method used to build models where the value of the state variables change at discrete points in time, as stated by (Heath et al. 2011). The discrete point in time when one or more state variables change is termed an “event”. DES does not include variables that change continuously with respect to time.

DES is widely used for various kinds of SC analysis studies. (Tako & Robinson 2012) provide a literature review of DES simulation works. The authors state that the most frequent issues handled using DES are system performance, inventory planning/management, production planning and scheduling. DES has also been used for SC risk analysis studies.

DES has many advantages in performance evaluation: First, it enables to build models including an extensive level of details if required. Second, it enables to represent different kinds of flows such as information flow, material flow, etc. Third, it enables to analyze both steady state and transitional state behavior of the system. As stated by (Van Der Zee & Van Der Vorst 2005), in many cases, DES is a natural approach in studying supply chains as their complexity obstructs analytic evaluation. (Persson et al. 2012) state that DES models can handle the stochastic behavior of the SC and hence, queues and other phenomena dependent upon uncertainty in operation and transportation times can be evaluated.

DES has some limits. In fact, as stated by (Tako & Robinson 2012) many literature works suggest that DES is not suitable for strategic modelling as it does not normally represent systems at an aggregated level ( (Baines & Harrison 1999),(Law & Law 2008), (Oyarbide et al. 2003)). Furthermore, many authors highlighted the difficulties encountered when building DES simulation models especially for large size system, as the collection of the required data to feed the DES model and the validation of the created model are difficult.

## COMPARISON OF COMMON SIMULATION FORMALISMS

---

The selection of simulation formalism is obviously important for our work. (Heath et al. 2011) gave a comparison of the most common formalisms. We summarize this comparison in table 1.1. Each line of the table is dedicated for a criterion. Hence, we define eight criteria. The first criterion is “the level of aggregation”. It refers to the level by which the constructs are close to the SC elements. Hence, the higher the level of aggregation is, the closest the constructs’ language is to the SC elements and the simpler the model is.

The second criterion refers to the “decision-making level” (from strategic to operational). The third criterion is “the data requirement” which refers to the size of the data used as input to build and to initiate models. The fourth criterion is “Change of behavior while execution” which refers to the capacity of a basic construct to adapt its behavior while it is executed and receiving an external signal. The fifth criterion is “modeling procedure type” which refers to the category of the modeling procedure used for the formalism. Here the type refers to whether the procedure is “bottom-up” or “top-down”.

In the “bottom-up” case, the procedure starts by modeling sub-systems to end up with modeling the whole. In the “top-down” case the procedure starts by modeling the global system without details in the first place and then modeling its subsystems in the second place. The sixth criterion is “models complexity” which may be evaluated by the number of constructs needed to build a model. The last criterion is “Time advance mechanism” which



refers to the way by which time is advanced when the simulation is executed. Two types of time advance mechanism exist. The “Time step” mechanism refers to an increment of time by a constant value. The simulated times are multiple of this quantity. The second mechanism is the “next event” mechanism. It refers to an increment of time associated with events.

As highlighted by (Heath et al. 2011), each of the simulation formalisms is specific for a decision-making level and presents different levels of simulation technologies’ maturity. Namely, the SDS is mostly used to treat problems at strategic levels and is reported to be difficult to adapt for operational levels since it includes a lot of assumptions. Furthermore, the construction of SDS models requires an important intellectual effort since modeling a SC using the SDS modeling constructs (feedback loops...) is not so intuitive.

ABS is reported to have the largest scope: it is used for treating the problems of strategic, tactical and operational levels. But, ABS presents a weakness in SC simulation studies. It presents a simulation technology that is not so mature and that is still under development. For instance, it does not integrate a language permitting the definition of agents’ behavior and that captures the discrete nature of SCs.

The adopted simulation formalism for this Ph.D. work is DES. Besides its capability of covering problems of different decision levels and besides the maturity of its technology, DES is stated to be a natural approach for studying SCs as it has the ability to capture their complexity. Persson et al. (2002) state that DES has the capability of handling the stochastic behavior of SCs and that it enables the analysis of the uncertainty in SC parameters.

TABLE 1.1: COMPARISON OF THE THREE MAJOR SIMULATION FORMALISMS (INTERPRETED FROM (Heath et al. 2011))

Criteria	System dynamics (SDS)	Discrete event (DES)	Agents based (ABS)
Levels of aggregation	High	Low	Medium
Decision-making levels	Strategic	Tactical and operational	Strategic, tactical and operational
Data requirements	Low	High	Medium
Construct behavior change while execution	Yes	No	Yes
Types of Modeling procedure	Top-down	Bottom-Up	Bottom-Up
Models complexity	Low	High	Medium
Time Advance mechanisms	Time step	Next event	Time step or next event
Maturity of the simulation Technology	Mature	Mature	Needs development

In this paragraph, we gave a review of the most used simulation formalisms, their advantages, weaknesses and differences. In next, we review the literature proposition for integrating those formalisms within frameworks that aims to facilitate SC simulations and analysis.

### 1.1.2 MODELING FRAMEWORKS FOR SIMULATION

In this section, we review the literature about modeling frameworks for simulation and we analyze their features. This is to identify the gaps that we want to fill in through the framework that we propose.

A modeling framework for simulation is a support tool that assists SC practitioners to create an executable simulation SC model. Some frameworks integrate a method to create a conceptual model and a method to translate the conceptual model into a simulation formalism to get an executable model. Other frameworks provide an approach to directly build an executable simulation model. The interest of the research community in providing modeling frameworks for simulation is motivated by the fact that there is a lack of adoption of simulation for SC studies by SC practitioners as highlighted by (Cigolini et al. 2011). (Cigolini et al. 2011) explain this by the lack of user-friendly commercial solutions. Another reason is the expertise required to build a simulation model using the current simulation formalisms (e.g. DES). In fact, as stated by (Dai et al. 2014) an effective modeling and simulation approach should not be complicated for users.

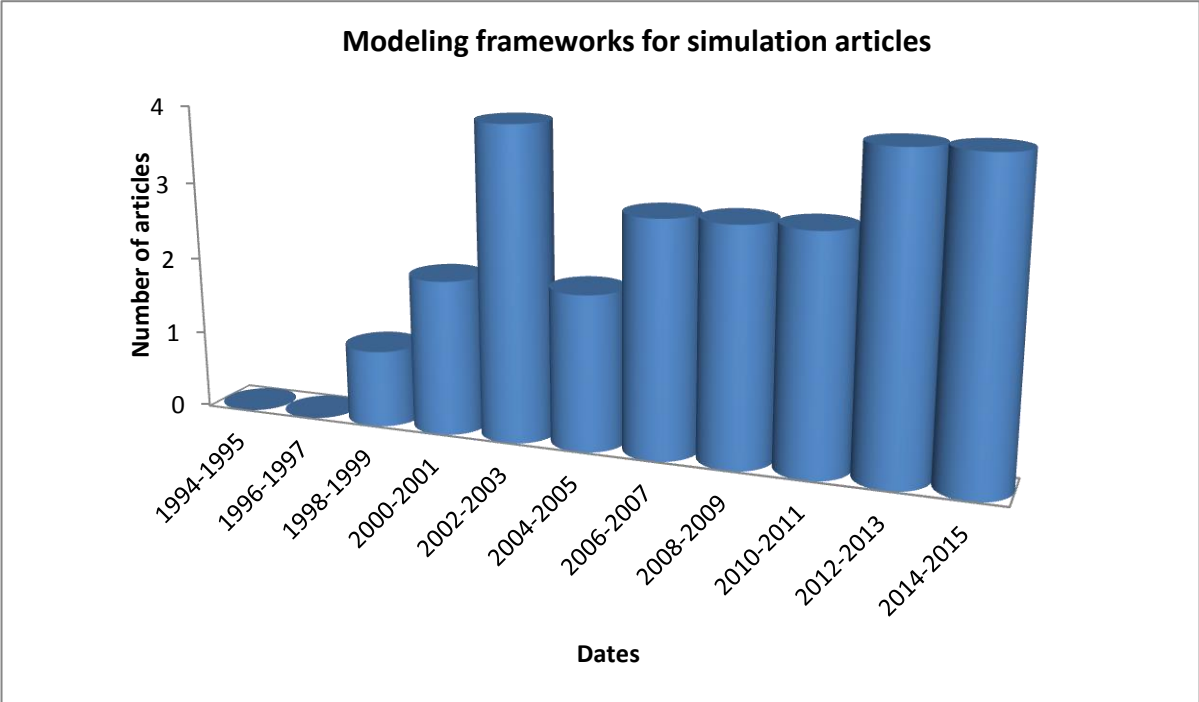


FIGURE 1.4: EVOLUTION OF THE NUMBER OF ARTICLES CONCERNED WITH MODELING FRAMEWORKS FOR SIMULATION

Figure 1.4 shows the evolution of the number of articles providing a modeling framework for simulation. The number of articles is determined thanks to “Google scholars” research engine. The number is defined for each couple of years starting from 1994 till 2015. The keywords used for the research in the searched articles title are (simulation modeling supply chain framework OR approach OR method). The Boolean logical operators (AND/OR) are used to combine keywords and to refine the research. As shown in figure 1.3, the research community contributions in this area started to be perceived in 2001. The number of articles published in this theme reached 26 at the end of 2015. This is a modest number. Among these articles, we select 13 that seems to be interesting. In fact, the articles that are specific to a given sector (e.g. mining) or the articles that are concerned with optimization are excluded. The articles that we select are the ones focusing on giving a framework for a generic SC.

To study the selected literature we define a set of features that characterize frameworks and may influence the SC practitioners' decisions to use one. "Technology Acceptance Model" proposed by (Davis 1989) suggests that when users are presented with a new technology, two major factors influence whether they will use it or not.

- Perceived usefulness (PU): refers to the degree to which users believe that using a particular technology would enhance their job performance.
- Perceived ease-of-use (PEOU): refers to the degree to which users believe that using a particular system would be free from effort.

The features we use are either increasing the PU, or the PEOU of the framework. The features are indicated in the columns of Table 1.2, which provides a review of the selected literature works. In this paragraph the first feature is concerned with the scalability of the framework; the following set of features is concerned with the modeling approach while the third set of features is concerned with the simulation task. The proposed features are as follows:

- The scalability refers to the modeling capability and is indicated with the dimension of the SC that can be modeled and simulated using the framework. It influences the PU of the frameworks.

The modeling approach features are:

- The SC domain knowledge refers to the source or the method by which knowledge is captured. It influences the PU of the proposed framework.
- Meta-model definition indicates whether the meta-model is presented in the paper or not and informs about the used meta-modeling language. We note that proposing modeling constructs and their relationships in the paper is more helpful for SC practitioners than only describing the modeling procedure. Second, the way the meta-model (the set of constructs and relationships) is presented, which influences the PEOU of the framework. Indeed, using a well-formulated meta-modeling language to communicate about the constructs and their relationships facilitate the modeling tasks.
- Modeling constructs provide an indication about the generality of the defined constructs or their specificity to SC domain. It is linked to the PU of the framework.
- Risk constructs: indicates whether some constructs are proposed to tackle the risk modeling and simulation or not. It influences the PU of the framework and enlarges the usage scope of the framework.
- Modeling procedure: provides an indication of whether the proposed modeling procedure to build SC models is detailed or not. It influences the PEOU of the framework.

The simulation features are as follow:

- The simulation library provides an indication of whether the simulation library is provided or not. It is linked to the PEOU of the framework.
- Definition paradigm refers to the paradigm used to graphically, textually or formally defining the simulation constructs. For the papers where the simulation patterns are not provided, we indicate the definition paradigm used for the case study descriptions. It is linked to the PEOU of the framework.
- The simulation formalism refers to the simulation formalism used to define the simulation constructs.

When there is no indication about a criterion in a paper we put the symbol (NI), which means "not included" in the corresponding case.



The literature review shows that not all the papers provide modeling constructs. Most of the works do not provide a complete definition or presentation of their meta-model. When the meta-model is not provided, usually the authors define the modeling procedure and illustrate it with an example (such as the work of (Labarthe et al. 2007)). Since the meta-models are the basics for the definition of modeling, not presenting the meta-model limits the adoption of the provided framework by the researchers and/or the SC practitioners. Only the works of (Cigolini et al. 2011), (Saleh Ebrahimi et al. 2012) and (Chatfield et al. 2007) provide a complete description of their meta-model using the object-oriented paradigm. The constructs provided by (Cigolini et al. 2011) are “manufacturer pull”, “manufacturer push”, “distributor push”, “distributor pull”, “retailer push” and finally “retailer pull”. The meta-model is limited to two SC actors and only one product. The meta-model of (Chatfield et al. 2007) presents general constructs such as nodes, or arcs.

(Saleh Ebrahimi et al. 2012) use SysML to describe meta-model constructs and their relationships to generic SCs. The constructs are domain-specific and are based on SCOR. But the meta-model correctness and effectiveness are not proved since the translation into simulation is not tackled and no case studies are conducted.

The way the modeling constructs are defined is important for the perceived ease of use. In fact, as stated by (Chatfield et al. 2007) “forcing modelers to confirm their understanding of a subsystem to a non-natural viewpoint may increase model building difficulty”. We identify two methods for the definition of the SC modeling constructs: The first method (M1) relies on the definition of a set of general constructs that are instantiated and customized in order to model the elements of a specific SC. This method provides high flexibility to the modeler in describing different scenarios but requires some customization to specify details and hence the modeling process is complex and time-consuming. To give some examples, (Van Der Zee & Van Der Vorst 2005) propose the generic construct called “Job”. This construct refers to the activity associated with specific transformation of goods and/or data. (Chatfield et al. 2007) propose “the action construct” to define process structure. The action construct defines an activity. The inputs and the outputs are defined for every action. The method M1 can also be applied to develop execution constructs. In fact, (Van Der Zee & Van Der Vorst 2005) define the construct “transformer” and the construct “Buffer”, which are responsible for the execution of the activities and the processes. The second method (M2) relies on the definition of a set of specific constructs extracted from the SC domain. Those constructs can easily be instantiated into company elements. It has the benefit of providing an easier and faster modeling approach to the modeler (i.e. numerous predefined constructs, hence low customization effort), but it has the disadvantage of reducing the freedom of the modeler. To give some examples, (Persson et al. 2012), (Long 2014) and (Saleh Ebrahimi et al. 2012) provide sets of domain-specific constructs for SC modeling based on SCOR model. Those constructs are customizable in order to model real processes and activities of any supply chain actor. (Persson et al. 2012) and (Long 2014) use level two and three of SCOR in an aggregated way that does not cover the different possibilities in which an operation can be executed and without specifying features of the exchanged variables defined by SCOR.

TABLE 1.2: REVIEW OF MODELING FRAMEWORKS FOR SIMULATION

Articles	Scalability	Modeling					Simulation		
		Meta-model	Modeling constructs	SC domain knowledge	Risk constructs	Modeling procedure	Simulation Library	Definition language	Simulation formalism
(Cigolini et al. 2011)	One product, Two actors.	Totally defined using UML	General: (Node, policy...)	Defined by authors	NI	NI	NI	NI	DES
(Mohammadi et al. 2011)	Network SC	Defined by authors.	general (event, resource, process, dependency)	Defined by authors	NI	Designed (Grammatical approach)	NI	NI	DES
(Persson et al. 2012)	Network SC	NI	Domain specific (Process Inquiry AndQuote...)	Extracted from SCOR	NI	NI	Not totally detailed	Intuitive graphs	DES with ARENA
(Casella et al. 2005)	Network SC	NI	Domain specific (Base company, consumer, ...)	Defined by authors	NI	NI	Detailed	Object-oriented	SDS with Modelica
(Cope et al. 2007)	Network SC	NI	Domain specific (Process, resource ...)	Extracted from SCOR	NI	IDEF +Automatic generation	Not totally detailed	NI	DES with ARENA
(Sprock & McGinnis 2014)	Network SC	Partially defined using SysML	Domain specific	Extracted from SCOR	NI	NI	Not provided	NI	DES with SimEvents
(Umeda & Zhang 2008)	Network SC	NI	Domain specific	Defined by authors	Ni	NI	Provided but not detailed	algorithmic	Hybrid: DES+SDS
(Kitagawa et al. 2000)	Network SC	Defined textually	Domain specific	Defined by authors	NI	NI	NI	NI	DES
(Long 2014)	Network SC	Defined textually as agents	Domain specific	Defined by authors	NI	Ni	Provided but not detailed	Agent	ABS
(Labarthe et al. 2007)	Cutomer centric SC	NI	NI	NI	NI	Designed (conceptualization+ operationalisation )	NI	Agent AUML	ABS
(Van Der Zee & Van Der Vorst 2005)	Network SC	NI	General (Agent, job, flow)	Defined by authors	NI	NI	Ni	NI	ABS
(Chatfield et al. 2007)	Network SC	Totally Defined using UML	General ( Nodes, arcs, components, actions..)	Defined by authors	NI	NI	Provided	Java classes	ABS
(Saleh Ebrahimi et al. 2012)	Network SC	Defined using SysML	Domain specific	Extracted from SCOR	Included	NI	NI	NI	NI

---

Providing domain specific constructs makes modeling easier than providing general constructs. For domain-specific constructs, the meta-model needs to capture the domain knowledge. The question here is how to capture the SC knowledge. One of the common methods for that is to use a commonly adopted textual descriptive framework such as the SCOR model. This improves the truthfulness of the modeling constructs. The works of (Saleh Ebrahimi et al. 2012), (Persson et al. 2012) (Cope et al. 2007), (Sprock & McGinnis 2014) propose specific constructs based on the SCOR reference model. Only the work of (Saleh Ebrahimi et al. 2012) integrates risk concepts.

Some works associate a modeling procedure with their framework such as the work of (Mohammadi et al. 2011), which provide a grammatical procedure to build a model. Some of the works provide constructs to build SC models and a general scheme for organizing the constructs (Cigolini et al. 2011); others go further and provide a methodical perspective guiding modelers during the modeling process, implementation and use (Cope et al. 2007). A minority only provides the details of simulations constructs to be used for translating the conceptual model into a simulation model (Chatfield et al. 2007).

Most of the works do not provide simulation library, only the work of (Casella et al. 2005) define simulation patterns using an object oriented paradigm. The simulation library is very useful for simulation software developers who seek for a well-established simulation patterns that cover domain knowledge to be included in their SC simulation software. Various simulation formalisms are used such as DES, ABS, and hybrid (DES +ABS). Some works, such as the works of (Long 2014), give an interest to distributed simulation.

## 1.2 LITERATURE REVIEW ON SUPPLY CHAIN RISK MANAGEMENT

The analysis of the literature given in the previous sections enables us to identify the gaps to overcome in the current SC analysis tools and to propose a set of requirements to consider. These requirements have to be complemented by the specificities of risk analysis activities. In this section, a literature review for the supply chain risk management (SCRM) is conducted. We present a state of the art about the works that tackle the SCRM process and its steps. The researchers provide a set of methods and techniques to assist managers in implementing the SCRM process. For every step of the SCRM process, we investigate the proposed methods and techniques with a focus on simulation based techniques.

Supply chain risk management (SCRM) permits a company to protect itself from the internal and external events that may incur negative impacts on SC performances, and assets. The interest of the research community in the field of SC risk management (SCRM) increased in the last 10 years. As shown in figure 1.5, the number of articles with a title that includes the keywords: [Supply chain AND risk] reached 2262 in 2016 as indicated by Google Scholar search engine. This reflects the efforts made by the research community in order to help SCs rising up to the new challenges of this era. The SC networks are witnessing an increase in the occurrence of risks. According to a survey (Simchi Levi et al. 2013) conducted in 2013 by a consultancy agency PwC with 209 companies, more than 60 % of the surveyed companies said that performance have dropped by 3% or more as a result of SC disruptions in the past twelve months. This is due to the evolution of SC features linked to globalization, a geographic extension of SC (multiple countries are involved in one SC), economic crises,

natural catastrophes (tsunami waves, hurricanes...), wars, rapid technological development, etc.

SCRM emerged from different disciplines: safety management, business continuity management, crisis management and enterprise risk management. Even if those disciplines provide various tools and methods, SCRM is still in need of new tools. SC managers need to know how to manage their risks and to get the required tools.

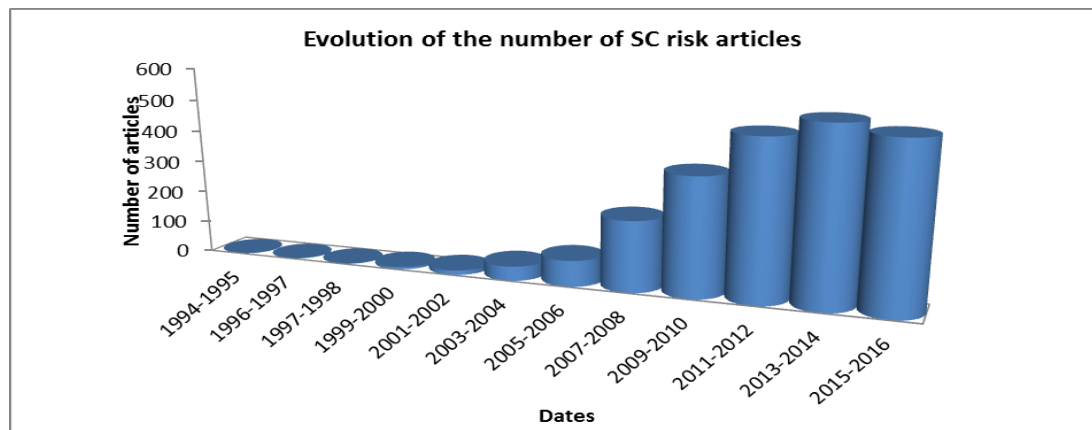


FIGURE 1.5: EVOLUTION OF THE NUMBER OF SC RISK ARTICLES

Researchers define supply chain risk in different manners. Some of the common SC risk definitions are shown in Table 1.3.

Most of the proposed SC risk definitions try to specify the features of the risk consequences. For instance, negative consequences for the focal firm by Wagner et al. (2006), variation in outcomes by (March & Shapira 1987), negative consequences to the system by (Tang & Nurmaya Musa 2011). This can be explained by the multiplicity of the source events that can lead to supply chain risk consequences.

SCRM is not only concerned with the focal company but it is stated to have a cross-company orientation, to be collaborative and to consider the SC as a whole as stated by (Tang 2006) and (Jüttner et al. 2003). This is why the scope of the SC has to cover various participants of the SC, not a particular one. Some authors succeed in considering the SC as a whole as in the definition of (March & Shapira 1987), while others limit the scope of their definitions to the focal company (Wagner & Bode 2006).

With reference to these discussed points and with reference to the provided general definition of risk, we would like to adopt the following definition for this work:

“SC risk is a scenario triggered by an event originating within or outside the SC which incurs negative effects on the objective of one or more SC elements. The realization of the scenario depends on both the source event and the state of the SC when the event occurs”.

As for SC risk, many authors tried to propose a definition for SC risk management. Table 1.4 provides a review of the common SC risk management definitions encountered in the literature. Different from the risk management definition in the systems engineering domain, the SCRM definition stresses on the following specific features: the SCRM approach is collaborative, coordinated between SC partners and is cross-organizational.

TABLE 1.3: REVIEW OF COMMON SC RISK DEFINITIONS IN THE LITERATURE

References	Definitions	Scopes	Effects Extents
(March & Shapira 1987)	The variation in the distribution of possible supply chain outcomes, their likelihood, and their subjective values.	General risk	Supply chain
(Zsidisin 2003)	The probability of an incident associated with inbound supply due to individual failures of suppliers (or supply market), that causes the inability of the purchasing firm to meet customer demand or cause threats to customer life and safety.	Only supply risks	Focal firm and customers
(Ellis et al. 2010)	An individual's perception of the total potential loss associated with the disruption of the supply of a particular purchased item from a particular supplier.	Only supply risk	Focal firm
(Wagner & Bode 2006)	The negative deviation from the expected value of a certain performance measure, resulting in negative consequences for the focal firm.	General risk	Focal firm
(Tang & Nurmaya Musa 2011)	(i) Events with small probability but may occur abruptly and (ii) these events bring substantial negative consequences to the system.	General risk	Supply chain
(Heckmann et al. 2015)	The potential loss of a supply chain in terms of its target values of efficiency and effectiveness evoked by uncertain developments of supply chain characteristics whose changes were caused by the occurrence of triggering events.	General risk	Supply chain

The goal of SCRM is described as the reduction of SC risks, the reduction of the vulnerabilities ensuring profitability and continuity, and as identifying, evaluating, monitoring events or conditions. The SCRM is an integrated part of SCM as stated by (Kersten et al. 2007). (Tuncel & Alpan 2010) highlight that if SCM that does not consider risk issues in a systematic perspective, it leads to sub-optimal results and inconsistent processes. In this work, we adopt the definition of (Ho et al. 2015) which seems to cover many features of SCRM. The definition is as follows: "SCRM is an inter-organizational collaborative endeavor utilizing quantitative and qualitative risk management methodologies to identify, evaluate, mitigate and monitor unexpected macro and micro level events or conditions, which might adversely impact any part of a supply chain".

Researchers define various steps for the SC risk management process. A review of the commonly cited steps in the literature is shown in Table 1.5. The steps of the SC risk management process differ in descriptions in the literature but we can depict a typical SCRM process as follow:

- **SC Risk identification:** In this step, the companies identify the threats that may degrade the capability of achieving objectives. It includes the identification of the source events, the mapping of the network, the propagation and the possible consequences.
- **SC risk assessment:** It involves the qualification of risks and their comparison and prioritization with regards to a set of criteria, such as magnitude, discovery or likelihood.
- **SC risk treatment:** It involves the selection, the design and the implementation of the risks countermeasures in order to decrease the risks level into tolerable level. Hence, a risk treatment plan has to be generated and implemented.
- **SC risk monitoring:** It involves the continuous revision of SC partners' performances, the information exchange about critical paths of partners, the monitoring of the

environment. The critical elements that may lead to trigger risk propagation process are observed permanently, frequently or event based.

TABLE 1.4: SOME COMMON LITERATURE'S DEFINITIONS OF SC RISK MANAGEMENT

References	Definitions	Features of the approach
(Kersten et al. 2007)	A part of Supply Chain Management which contains all strategies and measures, all knowledge, all institutions, all processes, and all technologies, which can be used on the technical, personal and organizational level to reduce supply chain risk	Contains all strategies and measures, all knowledge, all institutions, all processes, and all technologies Aim: Reduce SC risk.
(Jüttner et al. 2003)	The identification and management of risks for the supply chain, through a coordinated approach amongst supply chain members, to reduce supply chain vulnerability as a whole.	A coordinated approach amongst members. Aim: Reduce SC vulnerability as a whole.
(Tang 2006)	The management of supply chain risks through coordination or collaboration among the supply chain partners so as to ensure profitability and continuity	Includes coordination or collaboration amongst partners, Aim: ensure profitability and continuity.
(Ho et al. 2015)	An inter-organizational collaborative endeavor utilizing quantitative and qualitative risk management methodologies to identify, evaluate, mitigate and monitor unexpected macro and micro level events or conditions, which might adversely impact any part of a supply chain.	An inter-organizational collaborative endeavor. Aim: identify, evaluate, mitigate and monitor unexpected macro and micro level events or conditions.
ISO 31000 (Leitch 2010)	The coordinated activities to direct and control organization with regard to risk	Coordinated activities. Aim: direct and control organization.

TABLE 1.5: COMMON SC RISK MANAGEMENT PROCESS STEPS IN LITERATURE

References	Steps of SC risk management process
(Jüttner et al. 2003)	(1) Assessing the risk sources, (2) Identification of risk concepts, (3) Tracking the risk drivers, (4) Mitigating risks.
(Kleindorfer & Germaine 2013)	(1) Specifying sources of risks and vulnerabilities, (2) Assessment, (3) Mitigation.
(Harland et al. 2003)	(1) Map supply network (structure factors, key measures, ownership) (2) Identify risk and its current location (type, potential loss), (3) Assess risk (likelihood of occurrence, stage in lifecycle, exposure, likely triggers, likely loss); (4) Manage risk (develop risk position and scenarios); (5) Form collaborative supply network strategy, (6) Implement collaborative supply network strategy.
(Manuj et al. 2008)	(1) Risk identification, (2) Risk assessment and evaluation, (3) Selection of appropriate risk management, (4) Implementation of supply chain risk management strategy and mitigation of supply chain risks.
(Hallikas et al. 2004)	(1) Risk identification, (2) Risk assessment, (3) Decision and implementation of risk management actions, (4) Risk monitoring.



---

In the following, we explain SC steps and we review the proposed methods in literature with a focus on “risk identification” and ”risk assessment” since the contribution of this thesis concerns these two steps.

### 1.2.1 SC RISK IDENTIFICATION

In this step, the threats that can harm the SC, their sources, and their consequences are identified. The interrelations between the risks need to be mapped in order to have a complete picture of the risks threatening the SC. (Trkman & McCormack 2009) state that even though organizations might not be able to manage the source of the risk exposure, it is vital to identify the sources of potential problems and possible consequences. There is not a lot of literature about SC risk identifications.

The methods proposed for risk identification are summarized in Table 1.6. We define four categories to analyze the literature mentioned in the second column. They are as follows:

- Scenario-based: the identification of risks is done through the analysis of possible functioning scenarios of the SC.
- Objective-based: the identification includes a step of a top-down decomposition of objectives to identify causes of deviations.
- History-based: the identification of risks is done through the analysis of historical data to identify the feature of the events and the propagation scheme of risks. This method is limited when it comes to rare events with strong impacts.
- Taxonomy based: the identification of risks is done through checking the list of the SC risks belonging to each category defined by the taxonomy.

The scenario based, the history based and the objective based identification methods provide only a guideline to be followed by the SC practitioners. While the taxonomy based identification methods provide not only a guideline but also a database to facilitate identification. Many authors are interested in the taxonomy-based risk identification methods, and propose a taxonomy of risks, which includes factors and categories. For instance, the taxonomy proposed by (Blos & Miyagi 2015) is based on a vulnerability map and the taxonomy proposed by (Saleh Ebrahimi et al. 2012) is based on SCOR model to assist managers in the identification of their risks.

TABLE 1.6: LITERATURE METHODS FOR SC RISK IDENTIFICATION

Methods	Categories	Principles	References
<b>Value focused process engineering method</b>	Objectives-based	<p>The list of SC activities is generated and a performance objective and a risk objective are associated with every activity.</p> <p>A completely decomposed risk objectives structure is created starting from higher levels (system and processes).</p> <p>A map of activities risks objectives is generated through the synchronization of the two decompositions of risk objectives. This is by using the value-focus thinking rules (Keeney &amp; McDaniels 1992).</p> <p>Risk sources are identified by analyzing the extended-event-driven process chain (Scheer &amp; Nüttgens 2000).</p> <p>Risk sources are linked to risk objectives.</p>	(Neiger et al. 2009)
<b>HAZOP</b>	Scenario-based	<p>Generate the supply chain flow diagram (SCFD), which depicts the topology of the supply chain, entity information, and flow information.</p> <p>Generate the work-flow diagram (WFD) which describes the sequence of tasks performed by a functional entity (used resources, input, and output flow).</p>	(Adhitya et al. 2008)
<b>SCRIS</b>	Taxonomy based	<p>The identification is made using a knowledge base that contains facts and rules about potential risks. The knowledge based is integrated within a program called “knowledge-based system”. This program generates a description of the list of identified SC risks and the interrelationships.</p> <p>The program is fed with user input information about internal SC network, external SC network, and the SC structure.</p>	(Kayis & Dana Karningsih 2012)
<b>AHP</b>	Objectives-based	<p>Define the critical points for the achievement of every SC objective.</p> <p>Identify the risk factors of every critical point and the dependencies between them (using a matrix and flow chart (such as an Ishikawa diagram))</p>	(Gaudenzi & Borghesi 2006)
<b>Conceptual model</b>	Taxonomy based	<p>A taxonomy (called model by the author) is provided that specifies the SC characteristics, its structure, supplier’s attributes and performance, modified by factors in the supplier’s specific environment, namely exogenous and endogenous uncertainty.</p>	(Trkman & McCormack 2009)
<b>FMEA</b>	History-based +Scenario-based +Taxonomy based.	<p>The steps of this method integrate the identification of risk categories and the identification of potential risks. Usually, identification of risks is based on historical data or based on a scenario analysis but it can be also based on a predefined taxonomy.</p>	(Tuncel & Alpan 2010)

We are interested in the taxonomy based identification methods that define risk categories. We believe that it is more effective to focus on finding solutions for each risk category apart.



TABLE 1.7: SC RISK CATEGORIES IN LITERATURE

<b>Categorization principles</b>	<b>References</b>	<b>Risks categories</b>
<b>Origins</b>	(Jüttner et al. 2003)	<ul style="list-style-type: none"> <li>• Network-related risk,</li> <li>• Organizational risk,</li> <li>• Environmental risk.</li> </ul>
	(Christopher & Peck 2004)	<ul style="list-style-type: none"> <li>• External to the network,</li> <li>• External to the firm but internal to the supply chain network,</li> <li>• Internal to the firm.</li> </ul>
	(Trkman & McCormack 2009)	<ul style="list-style-type: none"> <li>• Endogenous risks,</li> <li>• Exogenous risks.</li> </ul>
	(Wu & Olson 2010)	<ul style="list-style-type: none"> <li>• Internal risks,</li> <li>• External risks.</li> </ul>
<b>Impacts</b>	(Tang & Nurmaya Musa 2011)	<ul style="list-style-type: none"> <li>• Material flow risks,</li> <li>• Financial flow risks,</li> <li>• Information flow risks.</li> </ul>
	(Cavinato 2004)	<ul style="list-style-type: none"> <li>• Physical,</li> <li>• Financial,</li> <li>• Informational,</li> <li>• Relational,</li> <li>• Innovational risks</li> </ul>
	(Christopher & Lee 2004)	<ul style="list-style-type: none"> <li>• Sales,</li> <li>• Customer service,</li> <li>• Operations,</li> <li>• Marketing,</li> <li>• Raw material supply.</li> </ul>
	(Bogataj & Bogataj 2007)	<ul style="list-style-type: none"> <li>• Supply,</li> <li>• Demand,</li> <li>• Process,</li> <li>• Environmental.</li> </ul>
	(Min & Zhou 2002)	<ul style="list-style-type: none"> <li>• Competitive strategy risks,</li> <li>• Tactical risks,</li> <li>• Operational routine risks.</li> </ul>
	(Talluri et al. 2013)	<ul style="list-style-type: none"> <li>• Disruption,</li> <li>• Distortion,</li> <li>• Delay.</li> </ul>
	(Saleh Ebrahimi et al. 2012)	<ul style="list-style-type: none"> <li>• Changing an operation by a degraded one,</li> <li>• Changing object attributes,</li> <li>• Destroying objects or associations.</li> </ul>
	<b>Likelihood of realization</b>	(Chopra & Meindl 2007)
(Tomlin 2006)		<ul style="list-style-type: none"> <li>• Short but rare,</li> <li>• Long but frequent.</li> </ul>
<b>Controllability</b>	(Byrne 2007)	<ul style="list-style-type: none"> <li>• Controllable risks,</li> <li>• Incontrollable risks.</li> </ul>

---

SC risk categorization helps managers in identifying the risks that threaten their SC. It assists them in selecting the required methods for SC risks evaluation. For instance, some authors (such as (Simchi-levi et al. 2015)) suggest not considering the likelihood of realization when analyzing the risks that belong to disruption category. Furthermore, the SC categorization permits the selection of the more adapted countermeasures to implement. For instance, (Chopra & Meindl 2007) suggest that their categorization of risks assists SC practitioners to design mitigation strategies.

As stated by (Ho et al. 2015) at least 20 research papers give an interest in providing an SC risks categorization. The most cited categorizations in literature are listed in Table 1.7. Every categorization highlights a given SC risk attribute. We observe four important risk attributes that are considered by researchers to build their categorization: the likelihood of realization, the origin of the risk, the controllability (i.e. the capability of controlling the triggers) and impacts. As seen in Table 1.7, most of the categorizations are based on the “impact” risk attribute. In fact, some of the proposed categorization refers to “impacted elements” of (Tang & Nurmaya Musa 2011) and (Cavinato 2004), others refer to “impacted functions” (Bogataj & Bogataj 2007), others refer to the “nature of impacts” (Talluri et al. 2013), others refer to “level of impacted activities” (Min & Zhou 2002). An interesting categorization in this group is the one proposed by (Saleh Ebrahimi et al. 2012), which focuses on the manner by which the model elements are impacted. This categorization is model oriented and concentrated on the best way to model risks and to emulate impacts. Indeed, it is more efficient for risk assessment to define a modeling way for every risk category rather than defining a modeling way for each risk apart. Since the field of our study is SC modeling we will adopt and refine the categorization proposed by (Saleh Ebrahimi et al. 2012) for our study in the upcoming sections.

### 1.2.2 SC RISK ASSESSMENT

The goal of the assessment step is to orient the risk treatment efforts to significant risks to assure the effectiveness and the efficiency of the actions to be implemented. As stated by (Zsidisin et al. 2008) prioritization is needed since it can be an extensive task to look across and down an entire SC in order to understand all the risks. Assessment permits bounding the possible values of risk attributes (such as impact level) for sorting and then treating them according to their importance. Researchers define this step of the SCRM process in various manners. For instance (Yates 1992) state that risks assessment involves: identifying potential losses, establishing the extent of losses, understanding the likelihood of potential losses, assigning significance to potential losses, and appraising overall risk, while (Steele & Brian H. Court 1996) state that the SC risk assessment consists of determining the probability of a risk event occurring, estimating the likely problem duration and investigating the business impact of the risk event.

The way the risk is understood influences the way the risk assessment is done. The adopted definition of SC risk determines how the risks are assessed. As stated before, we define the SC risk as follows: “SC risk is a scenario triggered by an event originating within the SC or outside which incurs negative effects on the objective of one or more elements of the SC. The realization of the scenario depends on both the realization of the source event and the state of the SC when the risk occurs”.

We deduce the following three risk components: the source event, the system state, and the impacts. Namely, the source event refers to the first event that triggers the chain of reactions

generating impacts. The system state component refers to a valuation of the system parameters that influence the propagation and the realization of the chain of risks events, while, the impacts refer to the effects on the performances of the system.

### 1.2.2.1 MODELS FOR SC RISK ASSESSMENT

Researchers use modeling to assess SC risks. Modeling permits to structure the relationships that exist between system variables for predicting its behavior. The more the model covers deeper details of a system, the more hidden variables are considered and the better is the predictive capability.

Some models are not effective in considering system state while others are effective in integrating all the risk components (e.g. cause system state and impact). Some of the proposed models only capture an aggregated view of the SC and risks while other models integrate a detailed view of the SC and risks. We propose to define and present two categories of models used for SC risk assessment: Risk network models and system oriented risk models.

#### RISK-NETWORK MODELS

Those models capture the faults propagation chain that leads to risks effects realization. In fact, those models are based on a mapping of the risk cause-effects relationships, which enable them to track and to characterize the critical paths that lead to severe risk impacts.

Researchers proposed many assessment methods based on these models. An example is the bow-tie model, which is based on the principles of event tree and fault tree diagrams. Bow-tie model is used to estimate the aggregated likelihood of the risk effects based on the estimated likelihood of risk causes (faults). As shown in figure 1.6, the three main components of the bow-tie are the risk factors (causes) to the left, the risk event in the middle, and the risk impact to the right. (Aqlan & Mustafa Ali 2014) use the Bow-tie model within a fuzzy inference system that permits calculating scores for risks. Risk likelihoods are estimated using fuzzy sets. Another example of use is the model proposed by (Klimov & Merkurjev 2008) who considers only the reliability attributes. The considered reliability attribute for every SC component is the probability that the component will not fail before the predicted time.

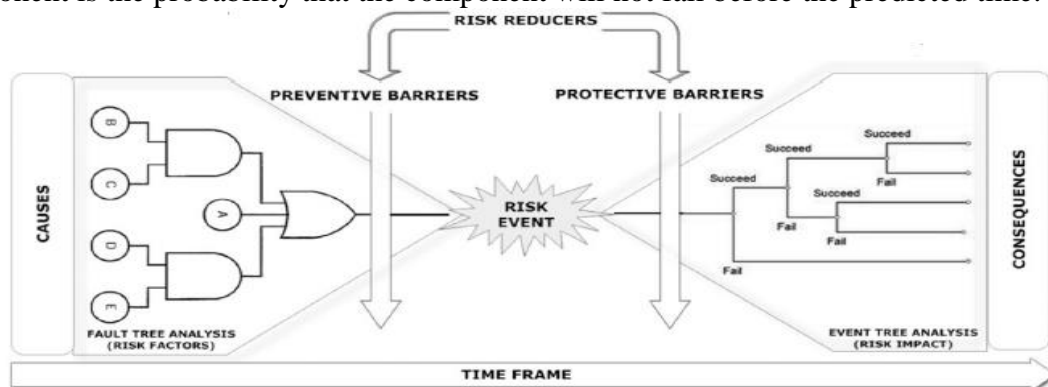


FIGURE 1.6: BOW-TIE DIAGRAM (BY (Aqlan & Mustafa Ali 2014))

The risk network models have the ability to capture the dynamic evolution of risks and enable the quantification of the likelihood attribute of impacts. They permit capturing the characteristics of the risk faults propagation chain. Namely, they take into account many risks triggering events and capture the dependency to the system states defined by the current values of the system parameters (e.g., the current inventory level, the current state of resources...).

---

The risk network models present limitations. Within the risk network model, the SC states are partially captured using a set of events defined based on expert judgments. In fact, since the SC is a complex system discretizing their features into events and proposing a subjective likelihood for them is not an easy task. For instance, most of the SC attributes are not considered (such as SC resource capacities and inventory policies) in the model of (Klimov & Merkurjev 2008). In fact, only reliability attributes are considered. The outputs are subjective and are based on experts' judgment. The expert's subjective judgment is not so precise to integrate system states within the risk network model. An example of an event for which the expert has to define the likelihood is "The inventory level becomes less than a given value". Since the inventory level may evolve rapidly, it is difficult to define the likelihood of being in a given level.

#### SYSTEM ORIENTED RISK MODELS

---

The limitations of the risk network models pushed researchers to propose more developed models that better consider the supply chain states in the risk propagation process. The models integrate both a model of the studied system and a model of the fault propagation chain. (Oehmen et al. 2009) provide a set of requirements that concerns SC states consideration for the model. They are as follows:

- Need to address the network characteristics of the supply chain relationships.
- Need to illustrate the dynamic behavior of the system.
- Must support hierarchical structuring.

Authors provide a set of requirements that concern the modeled risks. They are as follows:

- Address the network characteristics of the supply chain risks.
- Need to include risk causes and risk effects.
- Need to show the interrelations among different supply chain risks and the possible propagation paths of risks.

An example of the system oriented risk models is the one proposed by (Oehmen et al. 2009). In fact, (Oehmen et al. 2009) propose a model for risks and a model for the SC behavior linked through "truth functions". The truth functions are Boolean functions that indicate if the supply chain is in a given state or not. This function works through monitoring the attributes of the system in the "risk structure model" relative to a given state. A risk model is a state machine where states are linked to each other by transitions. Here the final states are the critical failures (i.e. the risk impacts). The SC model is a system dynamic model that presents the SC and its constituents as a set of causal loops diagrams. Another example of the system oriented risk model is the one proposed by (Aqlan & Mustafa Ali 2014). The authors propose a model for the SC network behavior that integrates risk factors. The risk factors are modeled as transitions using the High-level Petri net modeling formalism. The proposed simulation model has the benefit of integrating both risk network and supply chain behavior. A third example of these models is the one proposed by (Lockamy III 2014). In fact, the authors propose two models: The first model enables capturing the SC features to assess the SC vulnerability state and the second model captures the propagation network of risk faults to determine the likelihood that a supplier fault impacts the revenue of the focal company. The SC model links the following attributes:

- Relationship factors (influence, levels of cooperation, power, alignment of interests);
- Past performance (quality, on-time delivery, shortages);

- Human resource (HR) factors (unionization, relationship with employees, level of pay compared to the norm);
- Supply chain disruptions history;
- Environment (geographic, political, shipping distance and method, market dynamics);
- Disaster history (hurricane, earthquake, tornado, flood);
- Financial factors (ownership, funding, payables, receivables).

The second model proposed by (Lockamy III 2014) is the risk model that represents a tree of nodes referring to risks (such as operational risk) linked to a final node that refers to “supplier revenue impact”.

The system-oriented risk models have the advantage of considering both the propagation of faults to produce undesirable effects and the system states. So the system states instances are not the information given by experts but rather an output of a SC behavior model. Furthermore, those models make it easier to track and characterize the critical paths that lead to severe effects. In fact, they provide greater details of what is happening inside the system, the supply chain, in our case. Still, the construction of models for SCs is a difficult task due to the complex nature of supply chains.

#### COMPARISON OF THE SYSTEM ORIENTED RISK MODELS AND THE RISK NETWORK MODELS

Both categories of risk assessment models have their advantages and limits. A summary is provided in Table 1.8. For instance, the risk network models have the benefit of capturing the risk fault propagation chain to produce undesirable effects. They have the benefit to cover some practical lacks in SC risk assessment by considering some risk instances, which are frequently neglected. In fact, as highlighted by (George et al. 2012) most of the companies neglect indirect risks that can have a more significant impact than direct risks. The system-oriented risk models have the advantage of providing a more effective assessment since they integrate a developed representation of both the SC and risks.

**Table 1.8: Advantages and limits of SC risk assessment models**

Categories	Advantages	Limits
Risk network models	Permit good risks quantification since they consider the faults propagation chain.	Not all risk components attributes are considered. Not so precise in considering system states within the fault propagation chain. Also, information about system states is only provided by experts.
System oriented risk models	More accurate than risk network models thanks to the fact that information about system states and fault propagation are generated within the model.	Some difficulties may be encountered when building the model.

The selection of a model category is based on the available data and the possibility to model the risk network and the supply chain network. When the SC is too complicated to be modeled or when the persons in charge of the analysis are time constrained, the risk network models are selected. For an effective assessment, the best choice is the system oriented risk models since they consider both system states and risk propagation process.

---

#### 1.2.2.2 SUPPLY CHAIN RISKS ASSESSMENT TECHNIQUES

Many techniques are proposed to assess risks in supply chains. These techniques can be differentiated based on the adopted metrics and also based on the calculation method. For example, some authors such as (Simchi-levi et al. 2015) propose not to consider the likelihood attributes for the case of the disruptive faults. This is to take into account the difficulty or the impossibility to calculate the likelihood of disruptive faults as stated by (Chopra & ManMohan 2014). (Simchi-levi et al. 2015) propose to focus on the magnitude attributes of the cause events, of the “system states” and of the “undesirable effects”. For example, as magnitude attributes for the system states, the authors propose the “time to recover” for measuring the resilience of suppliers.

The SC assessment techniques can also be categorized based on how the metrics values are obtained. The metrics values can be determined through a simple calculation using tables or through running a simulation model.

We differentiate two categories of techniques: the table based assessment techniques and the simulation-based assessment techniques. In the next section, we review these two categories of techniques.

#### TABLE BASED ASSESSMENT TECHNIQUES

---

These are the techniques that provide a measure of the risk level based on a simple calculation of the likelihood of the fault realization and the magnitude of the consequence. The used data for the likelihood and the magnitude are provided by experts. Many of these methods integrate simplifications due to their limited capability in covering all of the risk components (such as the SC impacted system states). A popular simplification assumption is the usage of the likelihood of the fault realization instead of using the likelihood of the effects realization. For instance, classical FMEA (failure mode and effects analysis) technique proposes to calculate the probability of the causes that generate the failure mode instead of the probability of risk effects realization. (Chen et al. 2012) propose a technique to assess supply risks based on FMEA. The product of severity, occurrence, and detection called RPN (Risk Priority Number) is calculated for each selection criterion quantifying the supplier failure.

Another example of table based assessment techniques is the one proposed by (Hallikas et al. 2002). The proposed technique defines the risk index to prioritize risks. The risk index is calculated by multiplying the probability of the cause by the severity of the consequence.

The table based assessment techniques can be combined with computational engines to do the calculation when the expert inputs are probability distributions. To give an example, (Vilko & Hallikas 2012) calculate a risk profile for each risk driver through Monte Carlo simulation. The profile is the sum of the risk factors weights. The weight is found by multiplying the probability measures of the risk drivers by the delay distributions.

The table based assessment techniques have the advantage of providing a simple way to assess risks and they do not require a lot of data. They are mostly based on experts' judgment that is improved thanks to techniques such as AHP (Gaudenzi & Borghesi 2006). The table based assessment techniques have some limitations. In fact, they have limited capability in considering all the components of risks. Usually, the “system state” is weakly covered. For instance, the vulnerability and the recovery capabilities are not quantified. Furthermore, the expert judgment is subjective and integrates a lot of uncertainties.



These are the techniques that are based on the utilization of the output data of the simulation models. The SC risk assessment models discussed in the previous section can be simulated to quantify risks. As argued in the first chapter, simulation is a recommended technique that is adequate with the complex nature of SCs.

The number of the research studies that use simulation for SC risk assessment has increased in the last 7 years as shown in figure 1.7. Figure 1.7 shows the number of articles by a couple of years from 2001 to 2015. According to the search engine “Google Scholar”, the number of articles, which include the words “Supply chain”, “risk”, “simulation”, “analysis OR assesment” in their titles reached 14 between 1994 and 2015. The number of articles concerned with SC risk studies based on simulation is higher since not all the articles include the key words mentioned in their titles.

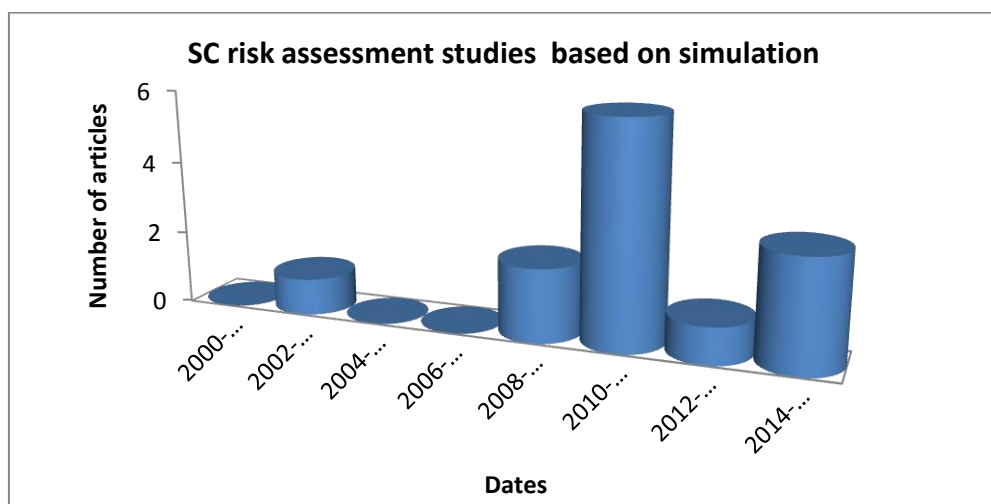


FIGURE 1.7: EVOLUTION OF THE NUMBER OF SIMULATION BASED SC RISK ASSESSMENT STUDIES ARTICLES

The number is still modest compared to other research fields but is still increasing. This tendency is explained by the fact that the methodologies for supply chain risk simulations recently appeared.

For further analysis, a set of articles is selected from the 14 articles found by “Google Scholar” research engine. We only consider journal or conference articles or that focus on a SCRM process step different than assessment. We include other interesting articles that are different from the ones mapped in figure 1.6 and that are cited by the selected articles.

We analyze the articles with regard to a set of criteria as shown in Table 1.9. For every study, we identify the category of the risk assessment model. We identify also the used simulation formalism. We refer the reader to chapter 1 that investigates common simulation formalisms for SC analysis. We identify the analyzed risks and the evaluation metrics besides the considered attributes to simulate risks.

The papers are classified based on the study type. Three types of studies are investigated:

- “Case specific” studies: they focus on the assessment of risks for a particular SC. At least half of the reviewed studies are case specific.
- “Prescriptive” studies: focus on giving insights on how risks impact SCs or focus on giving a prescription on the best strategies to adopt based on the assessments made. We think that these prescriptions are not normative and need to be reinforced with empirical justifications.

- The third kind of study is “Methodological” studies. They provide a methodology for risk assessment based on simulation. They are the less numerous.

TABLE 1.9: REVIEW OF SIMULATION BASED RISKS ASSESSMENT LITERATURE

References	Study types	Model categories	Simulation formalism	Risks	Evaluation metrics	Modeled Risk attributes
(Arisha & Mahfouz 2010)	Case specific	System oriented risk model	DES	Rush order risk	Cost and cycle time,	Editing time
(Carvalho et al. 2012)	Case specific		Transport disruption	Lead time ratio and total Cost	Disturbance intensity + disturbance duration	
(Tuncel & Alpan 2010)	Case specific		Quality, transportation and system failures	Customer order fill rate, total revenues	Transition having a probability and effects	
(Guertler & Spinler 2015)	Methodological		Operational risks	Risk magnitude	Stock level + Random evolution + magnitude	
(Deleris et al. 2004)	Case specific		General semi Markov + Monte Carlo	hazard events	Property damage lost production costs and the mean downtime.	Frequencies and severities
(Wu et al. 2013)	Prescriptive		ABS	Stock out	the market share	Stock out duration
(Seck et al. 2015)	Methodological		Forecasting errors and disruptions	Fill rate, asset utilization, and the inventory level.	Forecasting standard deviation + Capacity reduction and its duration	
(Talluri et al. 2013)	Prescriptive		DES	Disruptions+ delays+ Distortions.	Operating performance (i.e., customer service level, inventory turns, etc.) Costs expected	Change in time, capacity and order quantity values
(Schmitt & Singh 2009)	Case specific		DES	Disruptions	Fill rate	Frequency Duration
(Manuj et al. 2014)	Prescriptive		DES	Variations in lead time, cost, quality, and demand	Revenue, cost, and profit	Variability
(Kleijnen & Smits 2003)	Case specific	DES	Environmental factors randomness	Cost value and variance	Randomness	
(Miller & Engemann 2008)	Prescriptive	Risk network model	Monte Carlo	Disasters	Production index	Probability+ Degradation level
(Ghadge et al. 2013)	Methodological		SDS	Various SC risks	Cost and delay	Probability, cost and time
(Heckmann et al. 2015)	Methodological	Not defined	Not defined	SC factors variations	Risk line, performance indicators	System states ( SC factors), Deterioration of performance indicators

Thanks to the conducted analysis we enumerate the choices made by the researchers for developing their studies. They are as follows:

- Most of the works develop system oriented models (e.g. the work of (Deleris et al. 2004)), This can be explained by their advantages compared to other types of models. For instance, they are more capable of considering both the dynamic behavior of the risk



---

network and the dynamic behavior of the SC. The reviewed works cover various types of risks from disruptions to forecasting errors and stock-outs. Some of them give an interest in proposing a general model for some specific risks (e.g. (Guertler & Spinler 2015), (Ghadge et al. 2013)). This facilitates the adoption of their approach by SC practitioners.

- Regarding simulation, the researchers used all of the previously described formalisms with a preference for DES. We notice also that the usage of ABS increased in the last years, we believe that this is linked to the new developments of their technical platforms. To evaluate the simulation results most of the works used SC performance metrics such as the fill rate, lead times, asset utilization, inventory levels, and costs.
- Despite that the simulation is reported to be a difficult task, few works give methodological perspectives and frameworks for assisting SC practitioners in analyzing their risks. Most of the proposed works, that use simulation to assess SC risk impacts, are case specific. They usually provide a specific simulation model for a specific SC and for a specific risk.

We believe that assisting SC practitioners in creating the simulation models of their SC risks is a promising research direction. Some works propose general models for some specific risks. Their perspective is thus limited by the kind of risks studied. Hence, we think that proposing general models for generic risks presenting similar features is an interesting direction. Providing frameworks for facilitating the integration of risk models within SC models and their simulation is hence the adopted research direction in this thesis.

The assessment of risks using simulation is one of the major issues of this dissertation. Hence, in chapter 2, we explain the developed modeling framework for simulation-based risk assessment.

### 1.2.3 SC RISK TREATMENT

In this SCRM phase, countermeasures have to be defined in order to treat the analyzed risks. The risk assessment phase provides the inputs for countermeasures design. The manager has the choice of whether accepting the risk or reducing it or transferring it (e.g. insurance contracts). In order to determine the most efficient countermeasure a more profound analysis needs to be conducted.

The design of appropriate countermeasures is not an easy task and requires a lot of research. Current, SC practitioners encounter failures in implementing risk countermeasures. As revealed by (Hult & Craighead 2010) companies like (Boeing, Cisco, and Pfizer) encountered unexpected losses and/or expenses of more than \$2 billion due to ineffective supply chain risk management decisions. (Chopra & ManMohan 2014) state that surveys have shown that managers do little to prevent incidents since the solutions to reduce risks are not weighed against SC cost efficiency.

The designed countermeasure needs to respond to some properties in order to be effective and efficient. (Chopra & ManMohan 2014) state that managers need to find an answer to this question: How to lower SC's exposures without giving up hard-earned financial performance gained from improved SC cost efficiency? The challenge here is to design countermeasures that reduce risk levels and permit financial gain at the same time. An example of a countermeasure that responds to this property is: "Sourcing from an effective additional low-cost supplier in order to hedge supply risk". This countermeasure permits both a financial gain and a reduction of the exposure to the effects of the primary supply disruption.

(Tang 2006) suggests other properties for the countermeasures to be designed. The first one is: "countermeasures must enable managing the inherent fluctuations efficiently regardless of

the occurrence of major disruptions”. The second propriety is: “Countermeasures have to enable the supply chain to become more resilient facing major disruptions”. (Tang 2006) highlight that it is difficult to reduce the likelihood of most unpredictable disruptions but it is easier to reduce the exposure.

Two kinds of countermeasures can be distinguished. The proactive countermeasures that require a firm to act in advance and the reactive countermeasures that treat risk effects after risk fault realization. Proactive countermeasures are better studied in the literature than the reactive countermeasures. In fact, (Ivanov et al. 2014) highlight a lack in the description of control process (a part of reactive countermeasures) and their impacts in case of different deviations and disturbances.

Many researchers propose a categorization of the strategies and approaches used to design countermeasures (e.g, (Jüttner et al. 2003), (Manuj et al. 2008), (Tang 2006), (Shao 2012) ...). To give an example, (Chopra & ManMohan 2014) propose a very interesting set of proactive mitigation approaches and implementation strategies. The strategies are based on the trade-off between cost and risk that need to be considered when tailoring a given mitigation countermeasure. The mitigation approaches are shown in Table 1.10 with their corresponding strategies. The other approaches are cited in Annex A3.

TABLE 1.10: SUPPLY RISK MITIGATION APPROACHES AND RELATED STRATEGIES (PROPOSED BY (Chopra & ManMohan 2014))

Mitigation approaches	Tailored strategies
Increase capacity	Focus on low-cost, decentralized capacity for predictable demand. Build centralized capacity for unpredictable demand. Increase decentralization as the cost of the capacity drop.
Acquire redundant suppliers	Favor more redundant supply for high volume products, less redundancy for low volume products. Centralize redundancy for low volume products in few flexible suppliers.
Increase responsiveness	Favor cost over responsiveness for commodity products Favor responsiveness over cost for short live cycle products
Increase flexibility	Favor cost over flexibility for predictable, high volume products. Favor flexibility for low volume unpredictable products. Centralize flexibility in a few locations if it's expensive.
Pool or aggregate demand	Increase aggregation as unpredictability grows.
Increase capability	Prefer capability over cost for high-value, high-risk products. Favor cost over capability for low-value-commodity products. Centralize high capability in the flexible source.

Different approaches are used in the literature in order to investigate the effectiveness and the efficiency of risk countermeasures and to select the best ones. Some of the literature works used simulation models for this issue ((Tuncel & Alpan 2010), (Schmitt & Singh 2012), (Manuj et al. 2014), (Talluri et al. 2013), (Berger et al. 2004), (Lundin 2012), (Chen et al. 2000), (Hishamuddin et al. 2012)...). We describe some examples of these works. For instance, (Tuncel & Alpan 2010) provide a Petri net model to evaluate the added value of risk mitigation actions. The mitigation actions are evaluated by their effects on cost. (Schmitt & Singh 2012) investigate the effect of inventory placement and backup facilities on supply

chain disruption through a discrete event simulation model. (Manuj et al. 2014) investigate the adaptability of a set of countermeasure strategies under different supply chain vulnerability conditions using simulation.

In SC risk simulation literature, few works give an interest to the way risk countermeasures are integrated into simulation models. (Talluri et al. 2013) present how the simulation model settings are parameterized to integrate risk countermeasures. The integrated mitigation countermeasures are proactive; they require a modification of the simulation scenario parameters and a modification of the structure of the SC model instance as shown in Table 1.11. A limited number of works tackled the issue of reactive risk countermeasures integration into simulation models. In fact, (Ivanov et al. 2014) highlight a lack in the description of control processes and their impacts in case of different deviations and disturbance. (Ho et al. 2015) point out the scarcity of studies that treat risk recovery.

The risk treatment is not the focus of this dissertation, but it will be implicitly treated when speaking about adapting simulation models for the experimentation of a given policy.

TABLE 1.11: MITIGATION APPROACH INTEGRATION IN SIMULATION (PROPOSED BY (Talluri et al. 2013))

Mitigation approaches	Settings
Increase capacity	+20% capacity
Increase inventory	+20% cycle and safety stock
Increase responsiveness	20% cycle time
Increase flexibility	20% production quantity
Aggregate demand	+cross filling
Increase capability	+transshipment
Redundant suppliers	+supplier

#### 1.2.4 SC RISK MONITORING

SC Risk monitoring involves the continuous revision of SC partners' performances, the information exchange with partners about risks' critical paths, the monitoring of the environment and the SC internal states. The critical elements which may lead to triggering risk propagation process are observed: permanently, frequently or event based.

SC risk monitoring did not attract a lot of attention in the literature. But due to the increased complexity of supply chains and the increase of supply chain vulnerability, SC practitioners and researchers recognize its importance and the need for further developments. As stated by (Blackhurst et al. 2005) SC practitioners become more aware of the need of integrating risk monitoring (named awareness by authors) to become a part of daily supply chain operations. Authors highlight the need to develop dynamic and real-time measures such as a dynamic risk index tools mapped into different SC attributes such as (area/port/location, global calendar, volume, and capacity...). Authors also highlight the need to focus on the prediction of capacity bottlenecks (both long and short term capacity overloads) in global transportation networks. (Sheffi & Rice Jr. 2005) defines a condition for integrating SC risk monitoring into SC management. The condition is to create a culture that allows "maverick" information to be heard, understood and acted upon. Management needs to be sensitive enough to identify a disruption before its cause is apparent. A research study (Simchi Levi et al. 2013) made by PwC consulting firm in collaboration with MIT in 2013 puts risk monitoring as an important criterion to categorize SC risk processes as mature. The study puts as maturity requirements: setting up sensors and predictors of change and variability and monitoring partners for their resilience levels. The study also states that to have a fully flexible response to risks the use of

---

real-time monitoring and analytics is required. Risk monitoring requires information sharing between the partners as found by (Hall & Saygin 2012) based on their investigation. Authors state that appropriate level of information sharing and operational visibility can mitigate the effects of risks relative to delivery to some extent. (Christopher & Towill 2000) state that to share information, SC processes need to be integrated through collaborative working between buyers and suppliers, joint product development, common systems.

Despite its importance, SC risk monitoring attracted little attention from researchers. (Ho et al. 2015) suggest that researchers have to extend the literature by developing an early warning monitoring system with adaptive risk indicators for various types of supply chains and validating the system empirically. (Zhang et al. 2011) propose an integrated abnormality diagnosis model, combining the fuzzy set theory and the radial basis function neural network, to provide pre-warning signals of production quality in the food production supply chain. (Blackhurst et al. 2005) describe a transportation event management system used for risk monitoring by SC practitioners to effectively identify potential problems based on calculated predicted lead times for different global channels. The transportation event management system provides a snapshot when something is wrong, but it is not capable of effectively predicting problems a priori. (Sheffi & Rice Jr. 2005) suggest adopting “near miss” methodologies famous in safety movement that pay attention to small disruptions as an indication of bigger problems. (Giannakis & Louis 2011) integrate risk monitoring into a framework to design multi-agent based decision support system for disruptions management and mitigation in manufacturing SCs. The framework integrates a monitoring agent that is responsible for collecting and analyzing data from partners and is responsible for triggering an alarm if an abnormal situation is detected. But authors did not provide the details of the tasks processed by this agent. (Blackhurst et al. 2008) propose a tool to track, to measure and to analyze supplier risk index evolution over time in order to detect a dangerous change of risk levels.

Risk monitoring is stated to be an important task that determines the maturity of the SC risk management processes and deserves to get more interest from researchers. Nevertheless, risk monitoring is not the focus of our work; therefore we will not get into further details concerning this step of the SCRM.

### 1.3. RESEARCH QUESTIONS

In this section, we summarize the major findings of the literature review. Hence, we start by citing the identified literature gaps that we want to overcome through this work, then we enumerate the major research questions answered in this dissertation and finally, we provide an outline of the resolution approach.

The literature review reveals that SC risk management attracted an increased interest of researchers in the last years. Therefore, the number of articles dealing with this subject increased significantly. This is due to the reconsideration of its importance with regard to the evolution of SCs. For instance, some authors such as (Tuncel & Alpan 2010) consider that not managing risks systematically provides sub-optimal SC management performances.

Our focus in this dissertation is to contribute to the state of the art through assisting the SC practitioners in analyzing the risks threatening their SCs using simulation. This is to cover the

---

lacks encountered in modeling frameworks for simulation proposed in the literature. The reviewed literature works highlight numerous advantages of simulation for SC risk studies. For instance, (Chopra & ManMohan 2014) suggest conducting stress testing to evaluate “what if” scenarios and hence to assess risks. Also, simulation is stated to have many advantages over optimization models for many SC problems. For instance, (Longo & Mirabelli 2008) state that simulation is better for capturing the stochastic behavior of SCs and (Pirard et al. 2011) state that optimization is better for the evaluation of SC policies.

The reviewed literature works highlight a problem of adoption of simulation despite its capacity to capture the dynamics of complex SCs. For instance, (Wu et al. 2006) explain the problem by the lack of usability of its current tools. While (Cigolini et al. 2011) explain it by the lack of user-friendly commercial solutions, the lack of internal skills and/or the lack of time to develop a simulation model from scratch.

Our analysis highlights that the difficulties encountered by the SC practitioners for constructing simulation models is due to the fact that the simulation software’s building blocks defined relative to the review simulation formalisms are far from the SC domain and have a low level of aggregation regarding the elements to be modeled. This is true for the three major simulation formalisms that are DES, ABS, and DSS.

Some of the reviewed literature works tried to tackle this problem, but still, they are not meeting the needs and the problem is waiting for new solutions. Namely, we noticed a lack of the quality and the number of methodological studies. We found that at least half of the reviewed studies are case specific. To resolve this problem, some authors (e.g. (Beamon 1998), (Min & Zhou 2002)) recommend developing specific modeling language for the description and/or the dynamic analysis of SC scenarios.

To tackle the problem, other reviewed works propose modeling frameworks for simulation (such as the works of (Saleh Ebrahimi et al. 2012), (Persson et al. 2012), (Cope et al. 2007) and (Sprock & McGinnis 2014)). Unfortunately, the frameworks do not cover all the requirements that make them easy to use: most of the frameworks failed in proposing a well established meta-model which enables a good communication of the results and a better description of modeling constructs and their relationships at the same time. Also, most of the frameworks failed in capturing the domain knowledge of SCs, in most cases, they propose constructs based on their own understanding of the SC domain without justifying the capability of those constructs to capture the SC domain. Most of the works do not integrate a building procedure of model instances. Furthermore, most of the frameworks define some modeling constructs for SCs without providing their translation into the simulation. Only the work of (Saleh Ebrahimi et al. 2012) provides modeling constructs for creating system oriented risk models for SCs. Our analyses highlight that system oriented risk models are the most appropriate for risk analysis, thanks to their capability of capturing the dependency of the risk propagation process to the SC states.

In this chapter, we also reviewed the works on simulation-based risk assessment techniques to verify if there are some methods or frameworks for assisting the SC practitioners in simulating the risks threatening their SC. We found that few works provide generic risk models that can be adapted to be simulation modules. This is due to the lack of a consensus on the definition and the grouping of risks that pushes researchers treating each specific risk a part.

---

The research question is how to develop a modeling framework for simulation that makes simulation easy to use and more useful for increasing its adoption by SC practitioners in risk assessment (considering the two criteria proposed by (Davis 1989) for technology adoption).

Hence, we analyzed extensively similar literature frameworks for identifying a set of recommended practices for designing frameworks of good quality. They are as follow:

- Providing a graphical description of a meta-model is very beneficial. Choosing a good definition meta-modeling language increases the expressiveness of the meta-model and the perceived ease of use.
- Capturing the SC domain knowledge using reference model increases the fidelity of the modeling constructs to the modeled reality.
- Including concepts for risks will enhance the analysis capability of the framework outputs and enlarge the scope of its use.
- Proposing a modeling procedure relative to the meta-model increases users' adoption by improving the perceived ease of use.
- Providing a simulation library helps SC practitioners to build their own simulation models rapidly and easily.

For resolving the raised issues, we develop a modeling framework for simulation enabling a quick building of SC and risk models and their translation into simulation models associated with a procedure for experimenting risk scenarios. Furthermore, we provide a set of ready to use simulation modules integrated within a simulation tool. Hence, we follow the next steps for developing our research:

- Develop a meta-model for the SC structure, behavior, and risks.
- Translate the meta-model into a simulation model. This step requires programming efforts to convert the meta-model into an executable format so that different SC scenarios can be tested.
- Integrate a method to build simulation model instances.
- Test the developed tool on a case study.

## CONCLUSION

In this chapter, we provide an analysis of the literature relative to SC analysis, SC modeling, and SC risk management. The investigation aims to identify the gaps in the literature and to set the research questions. Hence when investigating the SC analysis methods we found that despite that many papers dealt with SCs simulation through presenting case studies, its adoption by SC practitioner is still limited. Furthermore, we found that even though many researchers uncovered those lacks, the proposed frameworks do not integrate all the elements necessary to make the analysis of risks easier. Hence we identified a set of requirements to be considered when developing modeling frameworks for simulation.

When investigating the literature about SC risk management, we found that the integration of risk models within the system oriented risk models for SC risk analysis did not take enough attention. Hence, we identified a categorization that defines general risks to be used as a basis for developing generic risk models.



---

So the adopted research directions for dealing with the identified lacks are: The development of a modeling framework for simulation that meets the identified requirements and their enrichment with a risk layer providing generic models for generic risks.

In the next chapter, we explain in details the adopted methodology for the development of this work.

---

CHAPTER 2

THE FRAMEWORK'S  
DEVELOPMENT METHODOLOGY



---

## CHAPTER 2: THE FRAMEWORK'S DEVELOPMENT METHODOLOGY

### SUMMARY

In this chapter we present the methodology for developing the modeling framework for simulation for SC risk analysis.

We provide a theoretical introduction for meta-modeling and we define the strategies adopted to design constructs satisfying the quality requirements identified in the literature review. We explain the adoption of SysML as a metamodeling language used to express the meta-model. The main reason is the capability of SysML to express the constructs' relations thanks to its object oriented diagrams. Then we explain the adoption of SCOR as a basis for developing domain specific modeling libraries motivated by its large adoption by the SC practitioners. We discuss the selection of the risk categorization of (Saleh Ebrahimi et al. 2012). This categorization is used for defining a set of modeling constructs for each group of risks. Finally, we explain the selection of DES as the simulation formalism and the selection of ARENA software to develop simulation modules that translate the proposed modeling constructs.

### INTRODUCTION

The main objective of this thesis is to aid the SC practitioner in analyzing the risks threatening his/her SC using simulation. This is through providing a framework that enables a faster and easier creation of SC models including risks, their translation into simulation models (without having deep knowledge about simulation languages) and conducting a set of experiments on them. The framework is supported by a set of tools and the method of their use.

The modeling framework for simulation aims to fill in some of the lacks encountered in the literature. To cite some of them, first of all, there is a scarcity in the number of the framework proposed in the literature. In fact, most of the reviewed works are case specific. Second, most of the frameworks proposed in literature fail in covering all the aspects that enable an easy use and that enable better usefulness (including the coverage of the SC domain knowledge, the communication of well-structured modeling building blocks...). Furthermore, few works provide modeling building blocks and generic simulation model for risks.

Thanks to the literature review discussed in Chapter 1, we identified a set of best practices for the development of an effective modeling framework for simulation that permits overcoming the gaps identified in the literature. The recommended practices are as follows:

- Providing a graphical description of a meta-model is very beneficial. Choosing a good definition meta-modeling language increases the expressiveness of the meta-model and the perceived ease of use.
- Capturing the SC domain knowledge using reference model increases the fidelity of the modeling constructs to the modeled reality.
- Including the concepts for risks will enhance the analysis capability of the framework outputs and enlarge the scope of its use.

- Proposing a modeling procedure relative to the meta-model increases users' adoption by improving the perceived ease of use.
- Providing a translation guideline and a simulation library to help SC practitioners rapidly and easily build their own simulation models based on conceptual models.

We integrate the recommended practices in the development of the framework to fill in the literature gaps.

The framework that we will present assists the SC practitioner in the steps that he has to conduct to analyze the risks that are capable of threatening the SC. Figure 2.1 illustrates the support provided by the framework for each step of the SC risk analysis mission.

In fact, to analyze the risks threatening the SC, the first step that needs to be conducted by the SC practitioner is to build the conceptual model of the SC. The conceptual model needs to capture the elements that form the SC. It needs to capture the SC structure, the SC behavior, the risks threatening the SC and the interactions between the risks and the SC elements. The support provided by the framework for this step is a structure and a behavior meta-model and modeling libraries that specify a set of building blocks and how they can be connected together to model a given SC. Furthermore, we provide a meta-model of risks that covers numerous risks cited in the literature.

The second step that has to be conducted by the SC practitioner is to translate the conceptual model into a simulation model. The support provided by the framework for this step is a translation guideline that enables translating each element of the conceptual model into an element of the simulation model. This translation is also simplified by using a library of simulation modules.

Finally, the SC practitioner needs to experiment its model to test risk scenarios, the framework assists the SC practitioners through defining methods for the definition and the analysis of the scenario.

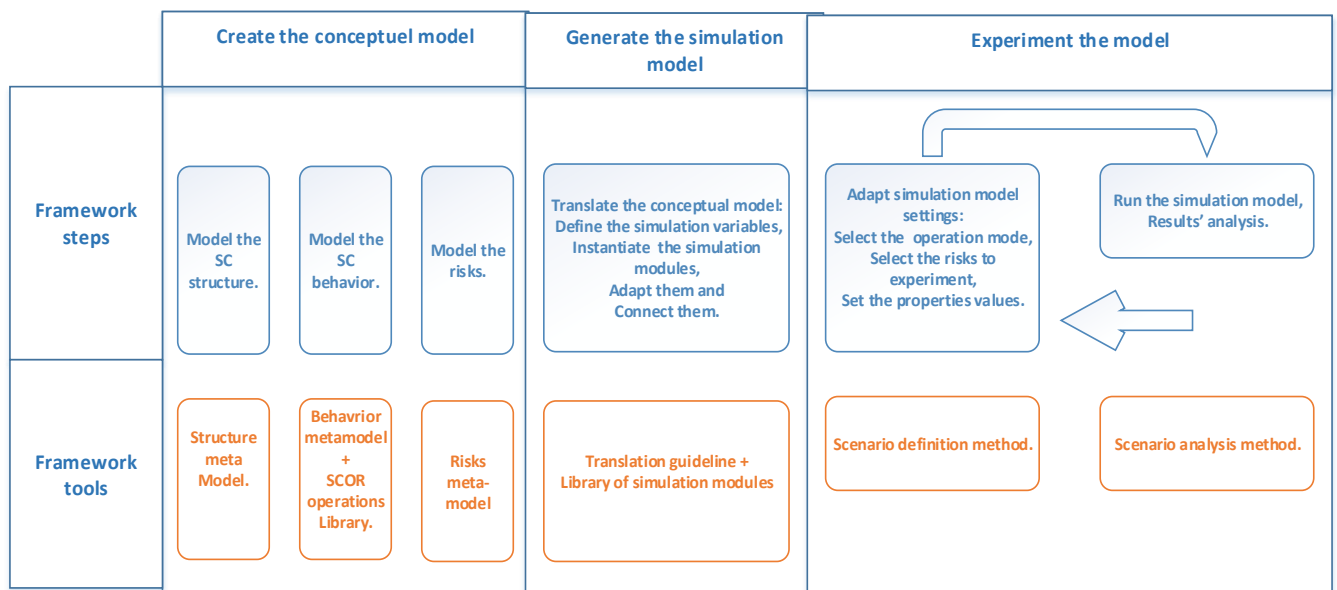


FIGURE 2.1: THE FRAMEWORK SUPPORT TO SIMULATION BASED ON SC RISK ANALYSIS

---

In this chapter, we describe the methodology adopted to develop the framework and the associated tools. Here, we explain the theoretical foundation of their development. This is by detailing the main choices made to develop the framework: We first explain the adoption of meta-modeling, the used tools and the followed principles to define the meta-model of the structure, the meta-model of the behavior and the meta-model of risks. Second, we explain the choices made for translating the modeling constructs and the conceptual model into an executable simulation model. We also explain the adoption of the discrete event simulation and the usage of ARENA as an example of simulation platform.

## 2.1 DEVELOPMENT OF THE MODELING APPROACH OF THE FRAMEWORK

In this section, we explain the choices made to develop the part of the framework that is concerned with assisting the SC practitioner in modeling its SC and the associated risks. In fact, we provide an overview of the theoretical foundation of meta-modeling, then we explain the choices made to develop the meta-models of the proposed framework. We start by explaining the adoption of “SysML” as a meta-modeling language, the adoption of SCOR as the source of knowledge about the SC domain that is necessary to build specific constructs. Then, we explain the adopted approach to define specific constructs to model risks.

Meta-modeling refers to the definition of a modeling language that permits the creation of models using the vocabulary of the domain knowledge. The modeling language is similar to human language that permits the creation of sentences which reflect our perception of the world and that are understandable by others.

The meta-model and the instantiation method are the main tools to express the features of a modeling language. The meta-model describes a set of building blocks and their relations. The users need only to follow the instantiation method to instantiate the building blocks, to customize them to get the required model.

The way a meta-model is designed determines its usefulness and its capability to cover the SC domain. (Clark et al. 2015) cite a set of criteria for good meta-models. In fact, they state that the best meta-models are the ones that have a well-defined semantic, that integrate a complete formalization of concrete syntax, that are completely defined and that are tested.

We identify two main capabilities that determine the quality of a meta-model. The first one is the capability to capture the domain knowledge. This capability depends on the manner by which the constructs are defined. The second one is the capability of expressing the captured domain knowledge. This capability depends on the manner by which the meta-model is presented.

In the first chapter of this thesis, we cited a set of criteria related to the capability of capturing the domain knowledge (that increases the perceived usefulness and the perceived ease-of-use explained in chapter 1) and we reviewed the success of the literature works in mastering this capability. We recall that the retained criteria are as follows:

- Enable a rapid and easy construction of SCs models. This means to reduce the time required for creating a model instance. This can be achieved by reducing the number of constructs and by reducing the complexity of customizing a given construct.

- 
- Enable to capture the SC domain and to create models for SCs that have a network structure.
  - Enable to create models of SC risks that capture the first impact and that enables capturing the chain of effects.

Furthermore, we cited a set of criteria related to the capability of expressing the captured domain knowledge and we reviewed the literature to understand if the existing meta-models are successful in mastering this capability or not. In fact, we found that many of meta-models in the literature did not provide a complete description of their meta-model (Such as the works of (Labarthe et al. 2007), (Casella et al. 2005) and (Cope et al. 2007)). Furthermore, many papers failed in communicating about the semantic of their meta-models when they provide a graphical description of them (such as the works of (Kitagawa et al. 2000) and (Long & Zhang 2014)).

In the next section, we describe the strategy adopted to design the meta-model of SCs and risks that master the previously described capabilities (capturing the domain knowledge and expressing it) and that enable the satisfaction of the cited quality requirements.

### 2.1.1 SysML AS A META-MODELING LANGUAGE

As stated before, an important quality criterion for the developed modeling language is to be well expressed. In fact, the features and the various relationships of the SC domain need to be well described by the developed meta-models in order to be well understood by adopters.

To assure that this criterion is respected and to follow the identified best practices, we use a meta-modeling language. In fact, this one provides a unified and platform independent way to capture the key features of a modeling language (abstract syntax, concrete syntax and semantic) as stated by (Clark et al. 2015).

The meta-modeling language provides a normalized syntax and a semantic to express a meta-model in a way that enhances the communication capability and the understanding of the developed meta-models.

Meta-modeling languages have been proposed in the literature. Unified Modeling Language (UML) is developed by the “Object Management Group” in the field of software engineering. It proposes a set of diagrams and a methodology to design software. UML is used, at the same time, as a modeling language and as a metamodeling language for specific areas of interests.

Another well-known metamodeling language is the Meta Object Facility (MOF) that was also proposed by the “Object management group” to specify UML. In fact, UML is an instance of the meta-model integrated within the MOF language.

SysML is an object-oriented (OO) modeling language that extends UML. We adopt it to develop the meta-models for the SCs and for the risks capable of threatening it. SysML takes advantage of the capability of the OO constructs (Such as packages, classes, and blocks) to provide a natural way to capture complex systems. In fact, as stated by (Coad et al. 1991), the OO concepts are aligned with the natural interpretation of the SCs expert domain to view the system as collections of related objects, including attributes of those objects, sub-components of those objects, and groupings of similar objects. In difference to UML that is developed for computer software design, SysML is developed for systems design. This difference is the main reason why we select SysML as a meta-modeling tool in this thesis work instead of

UML. SysML supports modeling through proposing a set of platform-independent graphical diagrams. SysML proposes nine diagrams. They are shown in figure 2.2. Four of them are concerned with behavior modeling; the other four are concerned with structure modeling and the last one is concerned with requirements modeling. The requirement diagram and the parametric diagram are specific to SysML, while the rest are adopted from UML 2.

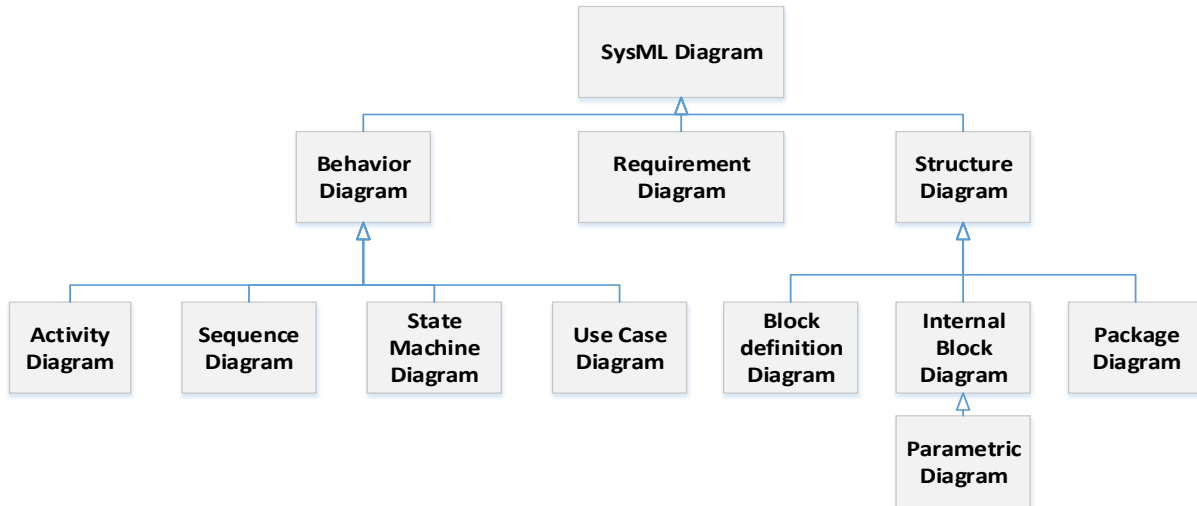


FIGURE 2.2: THE SYSML DIAGRAMS (ADAPTED FROM OMGSYSML<sup>2</sup>)

### 2.1.2 DESIGNING THE META-MODEL CONSTRUCTS

In this section, we explain our strategy to design the meta-model constructs that enable modeling the SC and the risks that are capable of threatening it. We refer to the literature review of chapter 1 that highlights two strategies adopted by researchers when designing the meta-model constructs. They relate to the definition of general constructs or the definition of domain-specific constructs or both.

To propose constructs that are easy to use and to enable capturing the SC domain, we choose to mix the two strategies. In fact, we propose general constructs that provide high flexibility and freedom to the modeler in describing different scenarios and we reduce the customization effort required to specify the details of the general constructs through providing libraries of domain specific constructs.

The libraries of domain specific constructs permit an easier and faster modeling approach to the modeler (i.e. numerous predefined constructs, hence low customization effort).

By following the recommended best practices, we extract the SC domain knowledge from a well-adopted reference model to define the libraries' specific constructs. In fact, as highlighted before, this will increase the fidelity of the modeling constructs to the modeled reality. The used reference model for this purpose is the SCOR reference model.

As stated in the previous section, the meta-model constructs will be expressed as blocks of SysML. To define the variables of the constructs that are expressed as properties of the meta-

<sup>2</sup> [www.omgsysml.org](http://www.omgsysml.org)

---

model blocks, we opt for the following strategy: First, we propose a set of variables based on SCOR, then we test them in simulation, the retained ones are the simplest to use and that well express the features of the simulated scenarios.

In next we will detail the choices made to design the proposed constructs (or building blocks).

#### 2.1.2.1 THE SCOR REFERENCE MODEL

One of the identified best practices for building a modeling framework for SCs is to extract the SC domain knowledge from a reference model. As stated before, we choose to use the SCOR reference model as a basis for the development of the library constructs.

The Supply Chain Operations Reference-model (SCOR) has been developed and endorsed by the Supply-Chain Council (SCC) in 1996 as a cross-industry standard diagnostic tool for supply chain management. It provides a unified terminology and standard descriptions of processes that can be used to describe supply chains. The SC functions are captured by SCOR model as a set of processes in three hierarchical levels. The fourth level, which is the implementation level that decomposes process elements, is out of the boundary of the SCOR model. SCC states that it is up to the company to decompose their own specific process elements. The proposed description levels are shown in figure 2.3. They are as follows:

- The level 1 (as named in the SCOR model) is the top level that defines the process types: it defines the scope and content of the SCOR model. It consists of five global process types: Plan, Source, Make, Deliver, Return and Enable
- The level 2 (as named in the SCOR model) is the configuration level that defines the process categories: Those categories enable a company to implement their operations strategy through the configuration they choose for their supply chain. Three policies for managing the supply chain are defined for the processes Source, Make and Deliver: policies linked to stock, linked to order and linked to engineering.
- The level 3 (as named in the SCOR model) is the process element level that defines for every process category the different elements that compose it (see the Comments column in figure 2.3 for these elements).







		Level		Examples	Comments	
		#	Description			
Within scope of SCOR	↑	1		Process Types (Scope)	Plan, Source, Make, Deliver, Return and Enable	Level-1 defines scope and content of a supply chain. At level-1 the basis-of-competition performance targets for a supply chain are set.
		2		Process Categories (Configuration)	Make-to-Stock, Make-to-Order, Engineer-to-Order Defective Products, MRO Products, Excess Products	Level-2 defines the operations strategy. At level-2 the process capabilities for a supply chain are set. (Make-to-Stock, Make-to-Order)
		3		Process Elements (Steps)	<ul style="list-style-type: none"> <li>• Schedule Deliveries</li> <li>• Receive Product</li> <li>• Verify Product</li> <li>• Transfer Product</li> <li>• Authorize Payment</li> </ul>	Level-3 defines the configuration of individual processes. At level-3 the ability to execute is set. At level-3 the focus is on the right: <ul style="list-style-type: none"> <li>• Processes</li> <li>• Inputs and Outputs</li> <li>• Process performance</li> <li>• Practices</li> <li>• Technology capabilities</li> <li>• Skills of staff</li> </ul>
Not in scope	↓	4		Activities (Implementation)	Industry-, company-, location- and/or technology specific steps	Level-4 describes the activities performed within the supply chain. Companies implement industry-, company-, and/or location-specific processes and practices to achieve required performance

FIGURE 2.3: THE DESCRIPTION LEVELS OF SCs DEFINED BY SCOR

We use the process elements proposed by level 3 of SCOR to define domain-specific operation constructs (e.g. we use the “**sM1.2/sM2.2 issueMaterial**” to define the operation sMi.2 ISSUEMATERIAL). Furthermore, we use the inputs and the outputs that are specified by SCOR for each process element to define a block of flows and a set of properties that model them.

SCOR was adopted by many literature works: (Pundoor & Herrmann 2006) propose a framework to build discrete-event simulation models using the SCOR textual syntax. (Persson et al. 2012) and (Long 2014) used level two and three of SCOR in an aggregated way that does not cover the different possibilities in which an operation can be executed and without specifying features of the exchanged variables defined by SCOR. (Gensym 2008) propose the e-SCOR simulation tool where the building blocks are designed using SCOR processes. (Dong et al. 2006) propose the IBM SmartSCOR, which is a simulation and optimization tool that uses the SCOR model to design their modeling constructs (Sprock & McGinnis 2014) propose an SCOR-compliant supply chain reference architecture that permits simulation models generation.

In difference to the above-mentioned works, we propose to go further and to define advanced constructs to capture SC features from SCOR. We note that, most of the SCOR based frameworks proposed in literature do not provide a description of how they capture the functioning of the SCOR behavior elements such as the works of (Sprock & McGinnis 2014) and (Long & Zhang 2014). Hence, we propose to define detailed algorithms for the operations captured from SCOR.

Furthermore, the previous works do not describe how they capture the inputs/outputs of the SCOR process elements. We propose to capture the flows transferred between SCOR functions through well-defined modeling constructs.

---

As highlighted by many authors the relationships between the SC actors are not clearly described when presenting the SCOR processes. Hence, we propose specific constructs to extend SCOR for capturing the relations and the interactions that exist between the SC partners.

Finally, aside from the work of (Saleh Ebrahimi et al. 2012), most of the SCOR based frameworks do not provide special considerations for risks. Hence, we propose to consider the interaction between risks and the SCOR elements.

### 2.1.2.2 DEFINING THE RISKS MODELING CONSTRUCTS

The question that we want to answer in this section is “What is the best strategy for defining the constructs that capture the risks threatening the SC?”

To respond to this question, we will refer to the results of the literature review. First, the analysis of literature shows that the system-risk network model based approaches are the most powerful for SC risk analysis and that it is important to specify constructs that can be integrated easily with these kinds of models. Second, the literature review shows a need to define generic models or patterns for risks. This is since the risks are numerous and a small number of specific risks are treated in the literature works at the moment.

Since the risks threatening the SC are numerous, the best way to handle them is to classify them in groups that capture the shared behavioral and structural features and to provide modeling constructs for each group. The modeling constructs need to enable an easier construction of SC system-risk network models.

Hence, in this thesis, we choose to define constructs for classes of risks based on how they impact the SC elements. We opt for the classification proposed by (Saleh Ebrahimi et al. 2012) which is oriented for modeling and that is crosschecked with the risks provided in the literature. This categorization will be refined in order to integrate all risk aspects: the categorization provides a set of generic risks. For each group of risks we provide a meta-model that specifies its attributes and its interaction with other SC elements. Hence, the SC practitioners only need to identify the group to which their risk belongs and to model it using the corresponding construct of the meta-model. Furthermore, in difference to the literature works providing simulation models for some case specific risks, we define the translation of the risk meta-models into simulation modules. Those simulation modules are generic models defined in a low-level simulation language.

## 2.2 DEVELOPMENT OF TRANSLATION GUIDELINES AND SIMULATION MODULES

One of the current difficulties in using simulation for risks analysis is the construction of the simulation models. The SC practitioner needs to understand and to capture the various features of the SC and to express it as a simulation model using the syntax provided by the simulation software. Even if the current simulation softwares are useful and effective, their simulation modules are not specific for the SC domain and are of a low level of abstraction. Hence, numerous simulation bricks need to be combined and customized to simulate a small part of a SC. This makes them time-consuming to use and requires a learning effort.

The solution that we develop in this thesis is to provide a translation guideline that enables the SC practitioner to directly translate its conceptual model into a simulation model expressed



---

using a DES language and to develop a set of SC domain specific simulation modules (or patterns).

The definition of a generic translation guideline consists of specifying for each conceptual construct, the simulations elements translating it and their relations. The translation needs to respect some criteria. In fact, it has to remain faithful to the conceptual constructs. Some of the compromises can be done in order to adapt to the constraints imposed by the selected simulation formalism. Hence, the simulation modules need to share the same properties and relationships as the meta-model library constructs to generate the conceptual model (e.g. the association between the PRODUCE operation construct and the “Resource” construct that expresses how the SCOR sub-process “Produce” uses a set of resources for its execution, needs to be translated and mapped within the simulation modules).

Besides the properties, which give insights on the structure of the SC, the simulation module integrates an algorithm that captures the behavior of the modeled function. To illustrate the translation methodology, we develop a set of simulation modules expressed in the DES formalism using a well-known commercial software. In the next section, we discuss our choices on the DES formalism and the simulation software used in this dissertation.

#### 2.2.1 DES AS SIMULATION FORMALISM FOR TRANSLATING THE CONCEPTUAL CONSTRUCTS

As stated before we choose to translate the meta-model and the libraries using discrete event simulation (DES) formalism. The reasons behind this choice are as follows:

First, DES enables to build models including an extensive level of details if required. Second, it enables to represent different kinds of flows such as information flow, material flow, etc. Third, it enables to analyze both the steady state and the transitional state. As stated by (Van Der Zee & Van Der Vorst 2005), in many cases, DES is a natural approach in studying SCs as they have the ability to capture the complexity of SCs. Furthermore, DES is stated by (Persson et al. 2012) to have the capability of handling the SC stochastic behavior by enabling the evaluating the uncertainty in the SC parameters.

#### 2.2.2 ARENA AS AN EXAMPLE OF A PLATFORM FOR TRANSLATION

We will illustrate the translation approach through developing a set of SC domain specific simulation modules corresponding to the library of operations.

The development of simulation modules includes the development of simulation algorithms, the declaration of the simulation variables, and the development of a human-machine interface to set the parameters values.

To develop these modules we use commercial simulation software that integrates simulation modules development tools and that enables building simulation models using those modules. Hence, when explaining the translation guideline, we will show the correspondence between the building blocks of the adopted simulation software (selected as an example) and the building blocks of the DES simulation software in general. This is to enable the SC practitioner using a different software to develop its own simulation modules.

Many commercial simulation software exists in the market. (Cimino et al. 2010) give a survey on the most used DES simulation software shown in Table 2.1. One hundred simulation practitioners answered the survey. Every participant provided a score between 0 and 10 for every criterion. Every line of Table 2.1 refers to a given criterion. For instance, the three first

lines refer to the suitability of the simulation software to three different domains of application. ARENA seems to be well perceived by the interviewed simulation experts. In fact, it has the best scores for user ability, modular construction, the domains (logistic and manufacturing) and for the user community.

TABLE 2.1: SIMULATION SOFTWARE EVALUATION BY SC PRACTITIONERS (PROPOSED BY (CIMINO ET AL. 2010))

	<i>Anylogic</i>	<i>Arena</i>	<i>AutoMod</i>	<i>Emplant</i>	<i>Promodel</i>	<i>Flexsim</i>	<i>Witness</i>
Logistic	6.5	7.5	7	7.2	6.5	7	7.5
Manufacturing	6.6	7.5	6.5	7.2	6.7	6.7	7.5
3D Virtual Reality	6.6	6.9	7.3	6.8	6.7	7.2	7
Simulation Engine	7	8	7.5	8	7	7.5	8
User Ability	7	8	6	7	9	7.5	8
User Community	6.2	9	6.7	6.5	7.5	6.6	8.5
Simulation Language	6.8	7	6.25	6.5	6.5	6.7	6.5
Runtime	7.5	7	6.5	6.5	7.5	6	7
Analysis tools	6.5	8	6.9	7.1	7.7	6	7.8
Internal Programming	7.2	7	6	7	6.2	7	6.5
Modular Construction	6.1	7	6	6.5	7.5	7	7
Price	7	6	5.6	5.8	7	5.7	6

In general, the discrete event simulation softwares provide similar simulation modules, similar functionalities and similar mechanisms for defining variables; this is why we think that the translation that we provide can be easily transferred to other software.

To introduce ARENA, we report the following information. ARENA is a high level “simulator” developed by “Systems Modeling” and acquired by “Rockwell Automation” in 2000. It is based on the SIMAN simulation language. It proposes a set of simulation modeling constructs (modules) that need to be connected together and customized to build the simulation model. Modules are grouped into panels that compose a template. ARENA integrates Visual Basic for application in order to automate some algorithms. ARENA gives the possibility to design a set of graphical modules using the SIMAN language. These graphical modules permit the SC practitioners to easily build a simulation model through dragging and dropping the patterns (building blocks), connecting them and through their customization.

## CONCLUSION

In this chapter, we explained the methodology adopted for the development of the modeling framework for simulation. The framework aims to assist the SC practitioners in analyzing the risks threatening the SC through providing a set of tools. The provided tools enable modeling the SC and the associated risks, translating the conceptual model into a simulation model, testing risk scenarios and analyzing the results. The choices made for the development of the framework tools are as follows:

- The development of meta-models,
- The usage of SysML as a metamodeling language for expressing the domain of SCs,
- The design of libraries of SC domain specific constructs,

- 
- The usage of the SCOR reference model as a basis for the development of domain-specific constructs,
  - To illustrate the translation to simulation through developing simulation modules in the (DES) formalism using ARENA as an example of a simulation platform.

Each step of our framework as well as the tools developed to support these steps is explained in details in the upcoming chapters.

---

**CHAPTER** 3

**THE MODELING FRAMEWORK:  
CREATING THE CONCEPTUAL  
MODEL**

---

## CHAPTER 3: THE MODELING FRAMEWORK: CREATING THE CONCEPTUAL MODEL

### SUMMARY

This chapter is presenting the conceptual model built to express the SC domain. This meta-model provides the user with the basic components to describe its own SC. These are easily combinable elements with a set of parameters to be specified for describing a SC through its actors, facilities or policies. The meta-model is built around two pillars: SC structure and SC behavior. The SC structure view proposes the elements to describe the static organization of the SC and its assets. Modeling constructs are given to specify the actor network, the exchanged products (including for example their bill of materials), the infrastructure (e.g. factories, stocks) and the transportation network. The behavior pillar is permitting to model the dynamic part of the SC. Modeling constructs are given to specify the flows animating the SC: material flow, financial flow and information flow. The policies defining the collective and individual behavior of actors are defined through the functions the actors execute. The coordination of actors is given through Process modeling constructs. In order to assist the model creation, logical groupings of common behaviors are given through the Role construct definition. The function realized by the actors are modeled through an Operation construct enabling the specification of a behavior with respect to the SC parameters provided in the structural description of the SC. Libraries of common SC Operations are also given to ease the creation of SC models. Finally, the hazards that may disturb the SC are modeled through risks classes. These risks models are modeled to be easily combined with the concepts of the SC meta-model. Risk classes are set up to cover each kind of risk effects that can affect the SC. These classes are defined as: Risks modifying a SC parameter, Risks modifying a behavior, Risks destroying an SC element. The presentation of the meta-model is presented as several complementary views to illustrate each aspects of the concepts. Finally, the use of the meta-model to build a specific SC model is illustrated through an example.

### INTRODUCTION

To analyze the risks threatening the SC, the SC practitioner needs to experiment a set of scenarios and to analyze results. Hence, the first task for the SC practitioner is to create a conceptual model for his/her SC. In this chapter, we present, in details, how to create this conceptual model and the related SC risks. We provide tools to assist the SC practitioner to this end. We recall in Figure 3.1 different phases to go through and the support tools. The framework proposal is as follows: (1) Model the SC structure. The SC structure is formed of the static elements of the SC (e.g., the Resources, the Buffers...). The framework supports this step by providing a structure meta-model. (2) Model the SC behavior. The SC behavior covers the processes and the activities performed within the SC and the exchanged flows. The framework supports this step by providing the SC practitioner with a behavior meta-model and a library. The library is specific to the SC domain and is extracted from the SCOR reference model. (3) Model the SC risks. A risks meta-model is supporting their modeling.

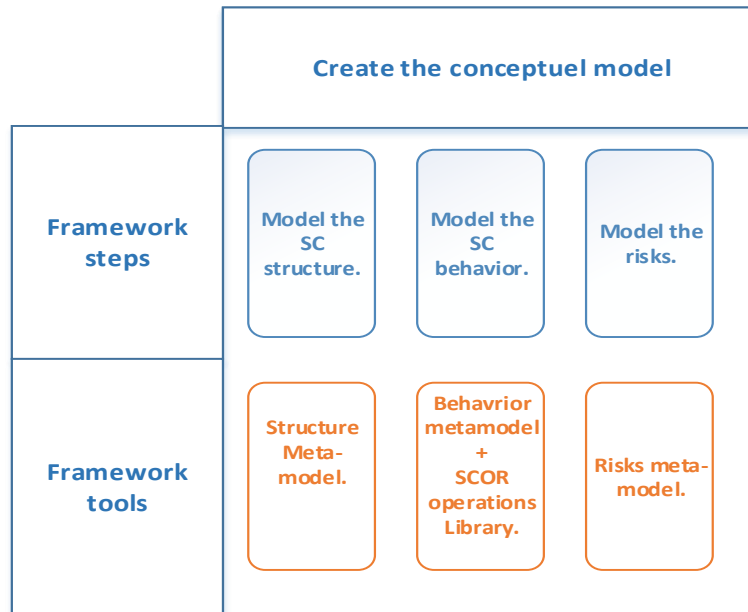


FIGURE 3.1: THE FRAMEWORK TO MODEL THE SC AND THE ASSOCIATED RISKS

In the following sections, we will present each of these 3 phases of the conceptual model.

### 3.1 MODELING SUPPLY CHAIN'S STRUCTURE

In this dissertation, the structure of the SC is understood as the static parts of the supply chain; the actors, the infrastructure (namely, the Facilities, the Buffers and the Resources), the Products, and the transportation within the SC (namely, the Routes, the Paths and the related Transfer or Transportation Resources). In our framework, we provide a meta-model that assists the SC practitioners in modeling these static elements.

Before presenting each of these static elements, we show a global view of the meta-model in figure 3.2. Actor blocks are linked together through the Contract block that defines the terms of the exchange between the Actors of the SC. An Actor holds a set of Facilities. Each Facility may hold Resources and Buffers. TransportationResource and TransferResource are specific types of Resources. The Buffers are linked together through Paths. A Path is associated with TransferResources. This is to express the fact that TransferResource may take a given Path in order to transfer products from a Buffer to another. Hence, these three blocks are used to describe the movement of products within a production site. Facilities are linked together through Routes. A Route is associated with TransportationResources. This is to express the fact that Transportation Resources may take a given route in order to ship products from a Facility to another. Hence, these three blocks are used to describe the movement of products between production sites.

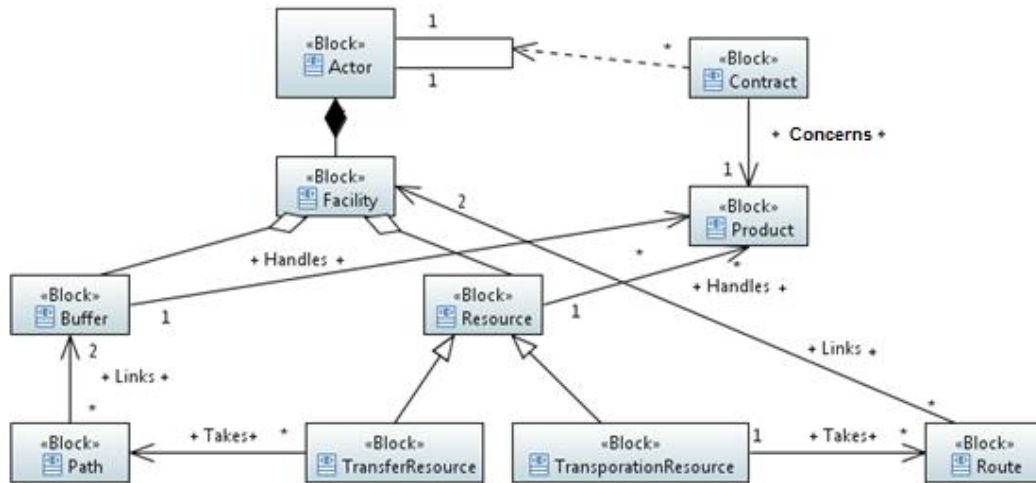


FIGURE 3.2: SC STRUCTURE META-MODEL BLOCKS DEFINITION DIAGRAM

The SC practitioner can model his/her SC structure in the light of the proposed meta-model. He/She needs to pick the meta-model elements that fit with his/her supply chain and then specify the properties' values according to his/her supply chain.

In the remaining of this chapter, we explain the meta-model through various views. Each view regroups a set of elements that either defines a structural level of the SC network (e.g. Actor's network level) or supports a given activity (e.g. transportation). We define four views of the SC structure meta-model: the actor's network view, the product view, the infrastructure view and the transportation network view.

The meta-model elements (also referred as meta-model constructs) are presented in SysML block definition diagrams. Each "Block" is composed of 2 sections:

- A heading describing the type of the element (Actor, Route, etc.),
- A set of properties describing a set of predefined attributes of the element (*Identifier*, *Capacity*, etc.).

In the following, we use italic letters to differentiate the property names from the block names.

### 3.1.1 THE ACTORS' NETWORK VIEW

When modeling the structure of his/her SC, the SC practitioner needs to model the relations that form the network of the SC. To assist the SC practitioner, the actors' network view is used to show the set of Actors involved in the SC. It enables to list them and to describe their links through Contracts. The blocks used in actors' network view are shown in figure 3.3. This view comprises the Actor block and the Contract block.

The Actor refers to a given participant of the SC such as manufacturers, retailers, etc. It has two properties for naming the Actor (*Identifier* and *Designation*). An Actor has a set of facilities specified through the *facility* property and a bank account referenced by the *MoneyAccount* property. An Actor may have relationships with many Actors. A relationship between two Actors is defined through Contract block.



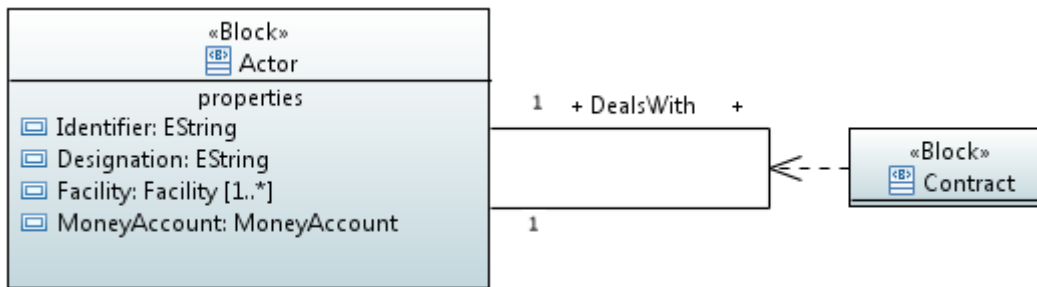


FIGURE 3.3: THE ACTORS' NETWORK VIEW BLOCKS DEFINITION DIAGRAM

The Contract refers to the block used to specify the relationship terms between Actors. There are two kinds of relationships between SC Actors. The first kind is a trading relationship where two parties exchange products and money. The second kind is a transportation relationship where two parties exchange the transportation service and money. To model the possible relationships that can exist between SC Actors, we propose a contract type for each type of relationship. So, the trading relationship terms are specified through the TradingContract block and the transportation relationship terms are specified through a TransportationContract block. The Contracts blocks definition diagram is shown in figure 3.4.

These two Contract types share a set of terms mentioned in the Contract block and inherit the shared properties from the Contract block. The shared properties are the *identifier* that names the contract, the *contractedProduct* that specifies the Product subject to the Contract. The *price* used to specify the traded Product price or the transportation price of the contracted Product. The *minLeadTime* and the *maxLeadTime* properties define the limits of the acceptable lead time for trading or for transporting. The *leadTime* refers to the required time to execute the requested transportation service or to deliver the required Product starting from the reception of the order. The *minPaymentLeadTime* and the *maxPaymentLeadTime* properties define the limits of the acceptable payment time. The *penaltyForDelay* and the *penaltyForPaymentDelay* properties define, respectively, the penalty cost per day of not respecting the delivery and payment delays. The properties *MinQuantity* and *MaxQuantity* define the lower and upper limits of the quantity to be traded or the quantity of the contracted Product to be transported. Besides the shared properties, each Contract block relative to a given type has its own properties. In fact, the TradingContract block defines the *returnPrice* property that specifies the price paid to the buyer for the returned product. Furthermore, the TradingContract block defines two properties to specify the quantity of Product to be reserved for a customer to be delivered in case of emergency. They are the *prioritizedQuantity* property that specifies the quantity reserved for a given customer, for a specific period of time. The period is defined by the second property which is the *priorityTime*. The TransportationContract defines the property *contractedRoute* that specifies the route used for transportation.

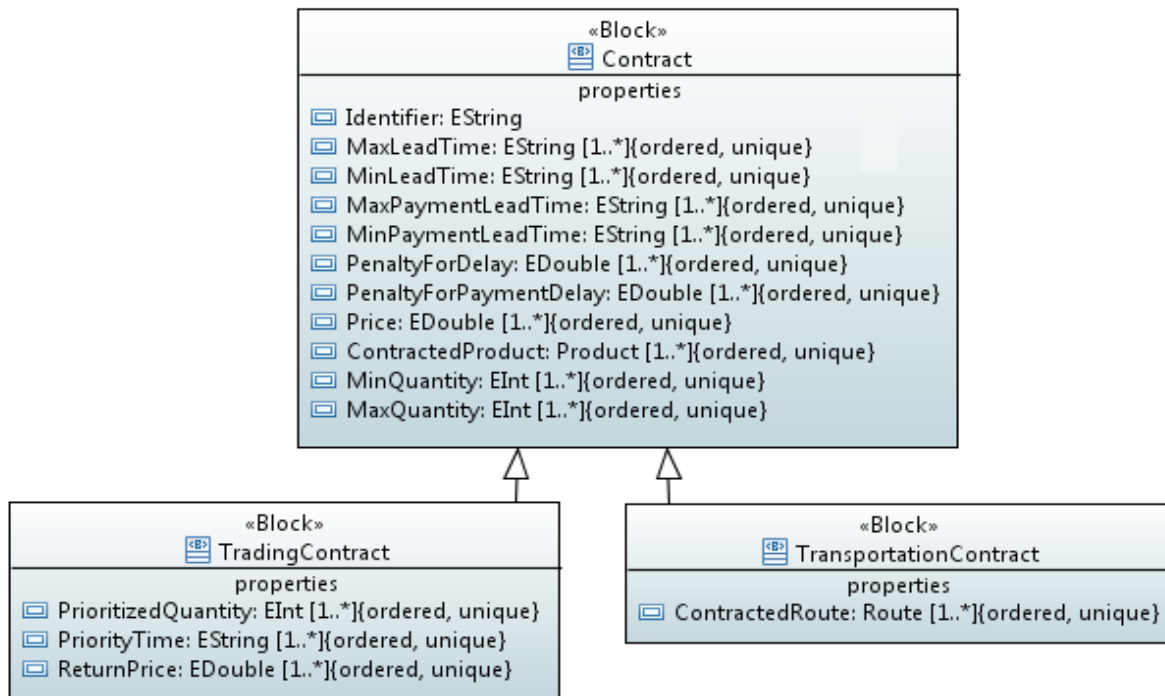


FIGURE 3.4: THE CONTRACT BLOCKS DEFINITION DIAGRAM

### 3.1.2 THE PRODUCT VIEW

When modeling the structure of his/her SC, the SC practitioner needs to model the products flowing in the SC.

The Product View refers to the block used to describe the products manipulated by the SC Actors. A Product is usually manufactured using other products (components) (see figure 3.5). The two first properties of the Product block are used for naming, which is the *identifier* and the *designation*. The property *shelfLife* indicates the conservation time, useful for perishable items. The *listOfComponents*, which is a vector of Products, specifies the list of Products or components used to manufacture the Product. The *billOfMaterials*, which is an integer vector, defines the coefficient or the required number of units of a component necessary to manufacture the main product. The *billOfMaterials* vector lists the coefficient values in the same order as the order of listing of the components in the *listOfComponents* property. For instance, to produce a Shatterproof Glass Water Bottle, we need one glass insert, one outer shell, one flip cape and one base. The value of the *listOfComponents* is [GlassInsert, outerShell, flipCape, Base] and the value of the resulting *billOfMaterials* is [1,1,1,1]. The *length*, the *width*, the *height* and the *weight* specify the dimensions of the Product useful for calculating the transportation loads.

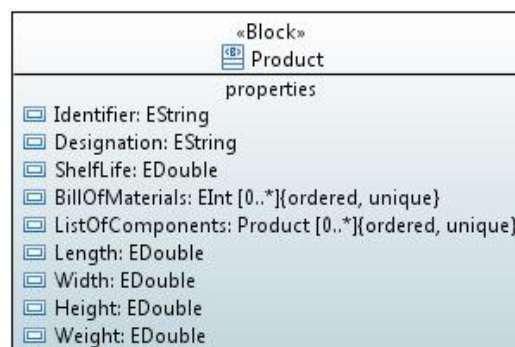


FIGURE 3.5: THE PRODUCT VIEW BLOCK DEFINITION DIAGRAM

### 3.1.3 THE INFRASTRUCTURE VIEW

When modeling the structure of his/her SC, the SC practitioner needs to model the infrastructure of the SC

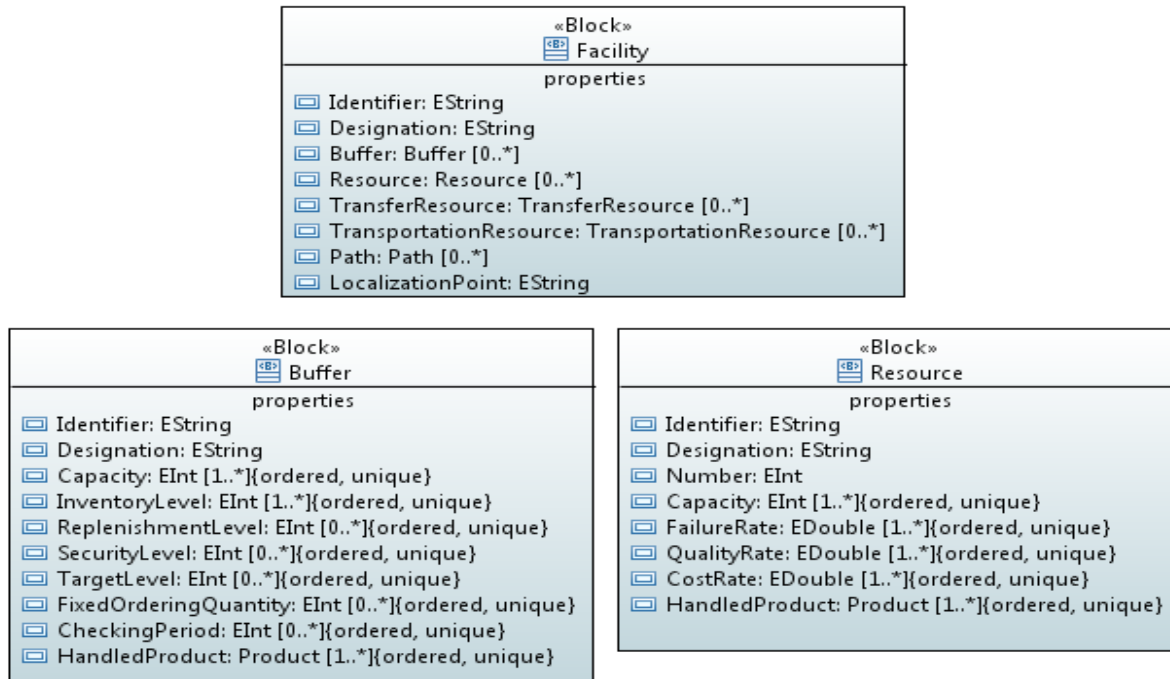


FIGURE 3.6: THE INFRASTRUCTURE VIEW BLOCK DEFINITION DIAGRAM

In our meta-model, we consider the Facilities, the Resources and the Buffers as the elements of the infrastructure (see figure 3.6.) .

The Resource block refers to the physical element required to perform one or several functions (e.g. a machine, an operator, etc.). The Resource block has an *Identifier* and a *Designation* for naming purposes. The Products handled by the Resource are specified through *HandledProduct*. The *HandledProduct* is a vector of products on which the Resource acts. The maximum capacity of the Resource defined for each treated Product is specified through the *capacity*. The *capacity* is a vector of integers. The number of replications of a resource belonging to a facility is specified through the *number* property. The *cycleTime* defines the required time to treat a given product. The *failureRate*, which is a vector of real numbers, is used to define the mean quantity of the handled products to generate a failure of the resource. The *qualityRate* that refers to the mean quantity of defective products generated by the resource per time and the *costRate* is a vector of real numbers that refers to the mean cost of treating one Product unit by the Resource.

The Buffer block refers to the location where Products are stored. It possesses the following set of properties: The *identifier* and the *designation* properties are used to name the Buffer, the *handedProducts* refers to the list of Products stored in the Buffer, the *capacity* that is a vector of integers refers to the maximum stored quantity for each handled Product, the *inventory level* which is a vector of integers refers to the quantities of stored Products. The Buffer has also a set of properties that are relative to the inventory management policy: The

---

*replenishmentlevel* which is a vector of integers refers to the inventory level that if reached the concerned Product has to be replenished, the *securitylevel* which is a vector of integers refers to the inventory level that has to be available in all situations and the *targetReplenishmentlevel* which are a vector of integer is used in case of adopting the “order-up-to-level” replenishment policy. This one refers to the level of products stored in the Buffer that has to be filled by the replenished quantity. The *CheckingPeriod* refers to the periodicity, by which the current inventory level has to be checked and finally the *OrderingQuantity*, which is a vector of integers, refers to the quantities to be ordered when editing a replenishment order.

The Facility block refers to the location of a set of physical entities (such as Buffers or resources...) that are grouped for production, storage or transportation purposes. Like the other structural elements of the SC, the Facility has two properties used for naming (*identifier* and *designation*) and a property used to define its localization, named *LocalizationPoint*. The Facility block regroups a set of Buffers, a set of Paths that link the Buffers, a set of Resources, a set of TransportationResources and a set of TransferResources.

#### 3.1.4. THE TRANSPORTATION NETWORK VIEW

The structure of the SC includes the transportation network. The Transportation Network View refers to the static elements used to define the possible movement of Products from one location to another. The blocks that form the transportation network view are shown in figure 3.7. They are as follows:

The Route block refers to the physical link that connects two geographical points where Facilities are located. It has two properties for naming (*identifier* and *designation*). The linked geographical points are specified respectively through two properties: the *startingPoint* and the *endingPoint*. The type of the Route (such as road or railway) is defined through *type*. The length of the Route is defined through the *length*.

The TransportationResource block refers to the physical transportation mean used to transport products from one geographical location to another (e.g. a truck). It inherits a set of properties from the Resource block (*Number*, *Capacity*, *HandledProduct*) but it also has its own properties: The *transportedLoad* which refers to the quantity of products transported within the vehicle (a value is specified for each product.) and the *tripTime*, that refers to the required time to travel through a given route (A value is specified for each route). The Routes used by the TransportationResource are specified in a list. We note that the *capacity* is valued for each transported product.

The Path block is similar to a route and refers to the physical link that connects two Buffers within a given Facility (e.g. the aisles in a warehouse). The Path is characterized by a set of properties, which are: the *identifier* used to name the Path, the *startBuffer* and the *endBuffer* refer to the linked Buffers. The *length* property is used to specify the distance between the linked Buffers.

The TransferResource block is analogous to TransportationResource, but is defined for transportation between Buffers within a facility. It, hence, refers to the physical transportation mean used to move products from a Buffer to another by following a Path (e.g. a forklift). The TransferResource block inherits a set of properties from the Resource block (*Number*, *Capacity*, *HandledProduct*) and has its specific properties. They are as follow: the

*transferredLoad* that refers to the quantity of Product transferred by the Resource and the *moveTime* which refers to the time spend to move the Products through a given Path (A value is specified for each Path).

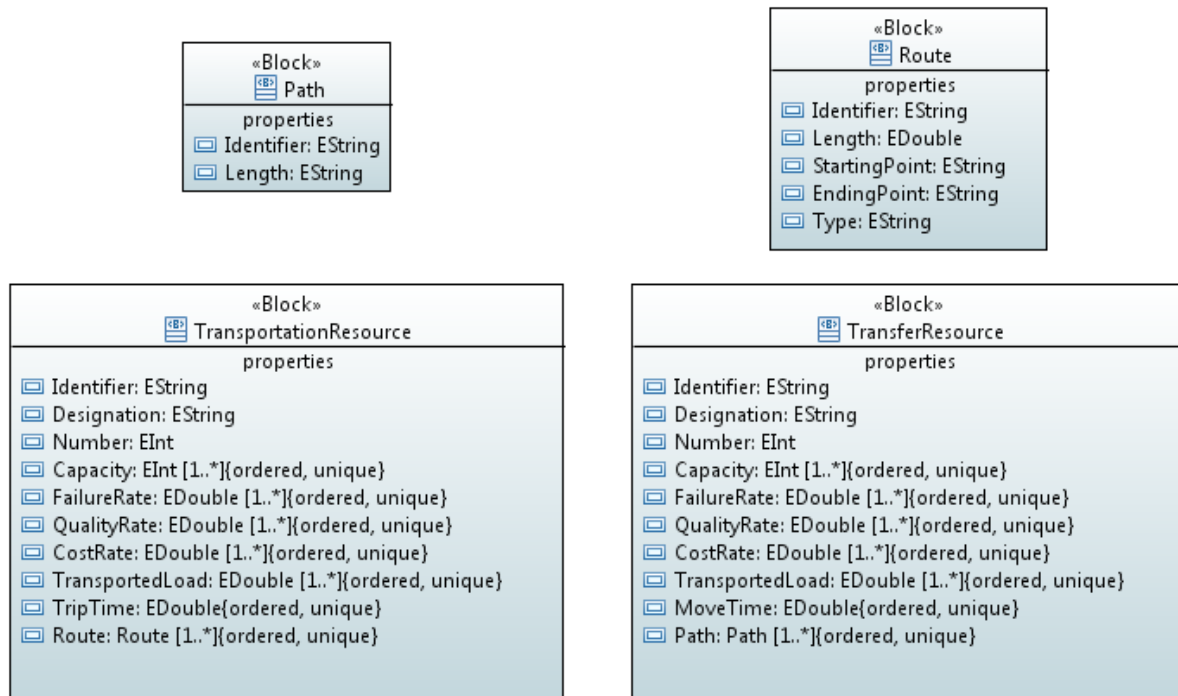


FIGURE 3.7: TRANSPORTATION NETWORK VIEW BLOCKS DEFINITION DIAGRAM

### 3.2 MODELING THE SUPPLY CHAIN'S BEHAVIOR

After modeling the SC structure, the second step is to model the SC behavior. The SC behavior describes the functionalities (e.g., the production functionality) performed within the SC using the SC structure elements. In this section, we provide a behavior meta-model that organizes a set of general modeling constructs. We introduce also the library of the SC domain specific constructs that provide more facilities for SC practitioners. In this section, we introduce the behavior meta-model, the library of domain specific constructs and the modeling approach.

The behavior model is composed of the following general modeling elements: Processes, Operations and their Interactions, Flows, and Roles. A global view of the behavior meta-model is shown in figure 3.8. This figure highlights the relations that exist between the behavior modeling blocks. An Operation block defines an activity performed within the SC. The Operation acts on variables. The variables are either the properties of a flow (Such as the *requiredQuantity*) or the properties of an Actor or of one of its component (Such as a *Buffer inventory level*). The interaction between the Operations is modeled through the Process and the OperationsInteraction blocks. The OperationInteraction is used to specify the connected Operations and the transferred Flows. The Flows are the information transferred between Operations.

A SC Actor may play one or many roles within the SC. For instance, he could be a manufacturer but also a supplier to another manufacturer. The Actor block is, hence, linked to

the Role block. The Role is linked to Operation in order to specify the set of Operations performed by Actors.

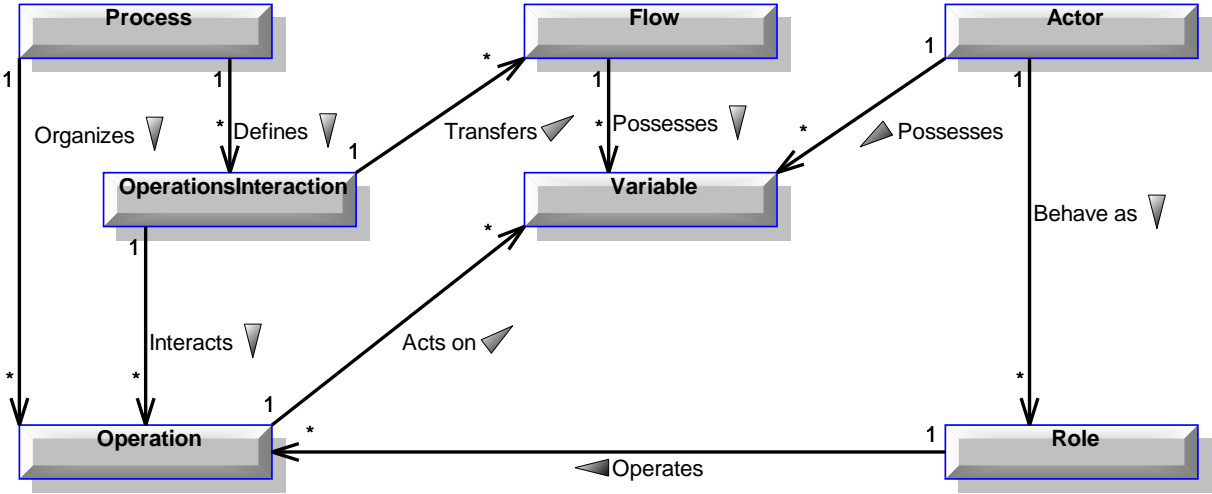


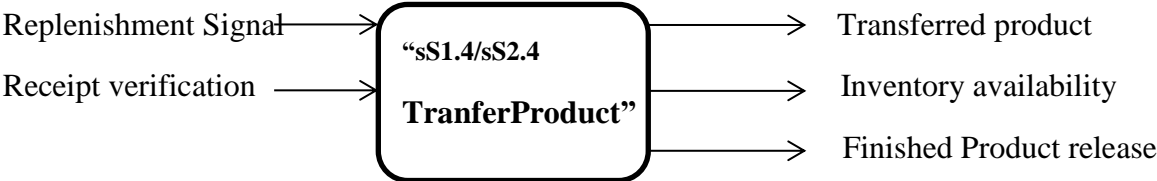
FIGURE 3.8: THE BEHAVIOR META-MODEL BLOCK DEFINITION DIAGRAM

Besides these general blocks, we propose also a library of domain specific constructs. In our case, this library is a collection of SC Operations. The Operations in the library are designed based on SCOR model. In order to avoid the confusion between the original SCOR Process elements and the Operations proposed in this dissertation, the names of the SCOR Process elements are given in bold characters between inverted commas while the names of the proposed Operations are in uppercase.

To explain the proposed constructs, we first start by presenting the constructs proposed to represent flows. Since the flows are edited and modified by Operations we present the Operation blocks in the second place. Since the Operations are interacting within a Process, we represent the Process block in the third place. Finally, we represent the Role block that gives access to Operations.

3.2.1 REPRESENTING FLOWS

One of the main questions that has to be answered when proposing modeling constructs for behavior is “what are the inputs and the outputs of the SC functions that have to be represented and how to represent them?” To well cover the flows that exist in the SC domain, we use the inputs and outputs described by SCOR as a basis for the definition of the constructs that model those flows. SCOR defines three categories of inputs and outputs: the material flow, the financial flow, and the information flow. In the following sections, we explain our proposal to model flows. “sS1.4/sS2.4**TransferProduct**” and “sD1.15/sD2.15**Invoice**” processes of SCOR (see figure 3.9) will be used as working examples to illustrate our proposition.





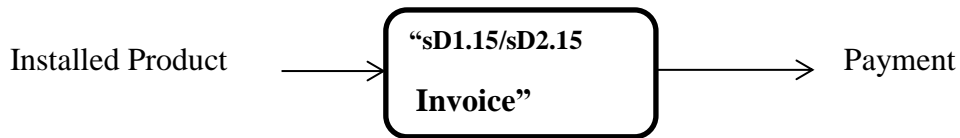


FIGURE 3.9: THE FLOWS OF THE SCOR PROCESS ELEMENTS “sS1.4/sS2.4 TRANFERPRODUCT” AND “sD1.15/sD2.15 INVOICE”

### 3.2.1.1 THE MATERIAL FLOWS

They are the inputs and the outputs of the SCOR Process elements of material type. They are the Product units on which the physical SC functions act (e.g. as shown in figure 3.9, the “transferred products” is the output material flow defined by SCOR for the Source Process element “sS1.4/sS2.4 transfer product”).

In our meta-model, the inputs and the outputs of material are modeled as a modification of the values of the Buffer’s *inventoryLevel* and of *transferdLoad* and *transportedLoad* for respectively the TransportationResource and the TransferResource. The modification of the *inventory level*, the *transferredLoad* and *transportedLoad* express respectively a modification of the physical presence of products in a Buffer, in a TransportationResource, and in Transfer Resource.

### 3.2.1.2 THE FINANCIAL FLOWS

They are the inputs and outputs of the SCOR Process elements of type money (or equivalent). They are the money handled by SC functions (e.g. as shown in figure 3.9, the Payment is defined by SCOR as the output financial flow of the Deliver SCOR Process element “sD1.15/sD2.15 Invoice”). In our meta-model, the financial flows are a modification of the values of the *MoneyAccount*. The modification of the *moneyAmount* expresses a modification of the numeric presence of money in the account.

### 3.2.1.3 THE INFORMATION FLOWS

They are the inputs and the outputs of the SCOR Process elements of type data which aims either to order or to inform (e.g. as shown in figure 3.9, the Replenishment Signal is defined by SCOR as an input information flow of the source Process element “sS1.4/sS 2.4 transfer Product”). Unlike the other flows, we define a set of information flow blocks to represent the inputs and outputs of type data provided by SCOR.

Each information flow aims to transfer a given message. We define three categories of blocks, based on the subject of the message transferred by the information flows: the Order, the Notification and the Program.

#### THE ORDER RELATED INFORMATION FLOW

It is a type of information flow exchanged between Actors to express a request for products or a request for payment. The Order related information flows have four properties: The *identifier* that names the order, the *edition date* that defines the date when the Order is generated, the *transmitter* that specifies the name of the sender and the *consignee* that specifies the name of the order receiver. The information flows of type Order are shown in the blocks definition diagram of figure 3.10. They are as follow:



---

The ProductionOrder is used to define the details of the production request such as the required Product and the required quantity to produce. Besides the properties inherited from the Order block, the ProductionOrder has: the *dueDate* property that defines the expected date when the manufactured products have to be ready; the *status* that specifies the state of evolution of the execution of the ProductionOrder, the *requiredProduct* that specifies the name of the Product to be manufactured and the *requiredQuantity* that specifies the quantity to be manufactured.

The PurchaseOrder is used to define the details of the purchase request. Besides the properties inherited from the Order block, the ProductionOrder has: the *status* property which specifies if the order is validated or to be changed, the *dueDate* that specifies the date when the products must be delivered, the *requiredProduct* and the *requiredQuantity* properties, which specify the customer requirements in terms of Products and quantities.

The ReplenishmentOrder is used to define the replenishment request details such as the products, the relative quantities to replenish and the Buffer of reception. Beside the properties inherited from the Order block, the ProductionOrder has the following properties: The *requiredProduct* and the *requiredQuantity* that specify the replenishment requirements in terms of products and quantities and the *receptionBuffer* which specifies the Buffer of reception where products are to be put.

The DeliveryOrder is used to define the general details of delivery such as the delivery date and what to be delivered. Besides the properties inherited from the Order block, the DeliveryOrder block has the following set of properties: The *deliveryDate* that specifies the date when Products have to be delivered, the *receiver* that specifies the delivery Actor, the *requiredProduct* and the *requiredQuantity* that specify the delivery requirements in terms of products and quantities.

The ShippingOrder is used to specify the shipping details such as who is in charge of delivering the products. Besides the properties inherited from the Order block, the ShippingOrder block has: The *carrier* that specifies the Actor in charge of delivering the products, the *shippingResource* that specifies the transportation resource used to ship products and the *shippingRoute* that refers to the route to be taken to deliver products.

The Invoice is used to specify the details of the payment request in exchange of the delivered or returned products or for paying a penalty. In addition to the properties inherited from the Order block, the Invoice block has the following properties: The *dueDate* that specifies the date when the requested amount is expected to be received, the *requiredAmount* that specifies the amount of the requested money, the *receptionMoneyAccount* that specifies the account where the requested money has to be received.

The DispositionOrder is used to specify the disposition request. In addition to the properties inherited from the Order block, the DispositionOrder block has: The *receiver* that refers to the facility where the products need to be returned, the *returnDate* property that refers to the requested date for returning the Products, the *requiredProduct* and the *requiredQuantity* properties that specify what to return and its quantity, respectively.

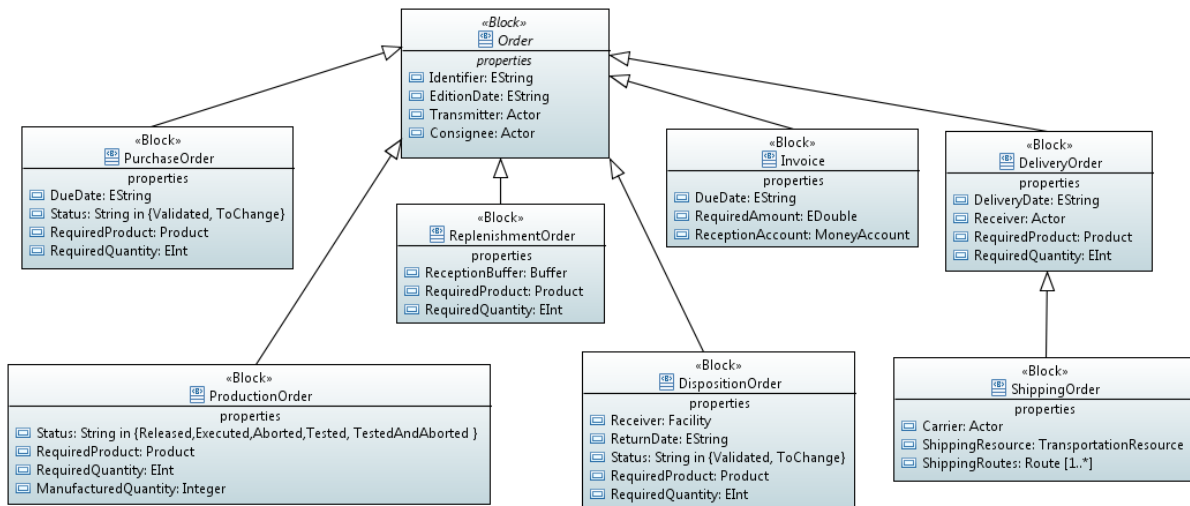


FIGURE 3.10: BLOCK DEFINITION DIAGRAM OF INFORMATION FLOWS OF TYPE “ORDER”

### NOTIFICATION INFORMATION FLOW

This type of information flow expresses a notification about a given state of inventory or a given state of execution. The information flows of type Notification are shown in the block definition diagram of figure 3.11. The information flows of this type share a set of properties, namely, the *identifier* used to name the notification object and the *editionTime* used to specify the date when the notification is generated.

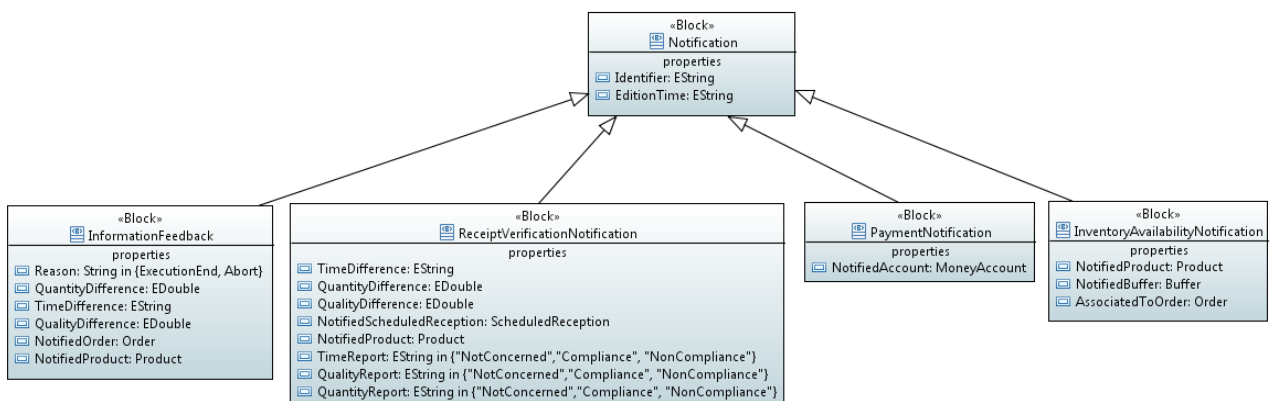


FIGURE 3.11: NOTIFICATION BLOCK DEFINITION DIAGRAM

The information flows of type Notification are as follow:

The InformationFeedback is used to inform about the execution’s state of Operations through reporting the found results. A set of specific properties is used for this purpose besides the properties inherited from the Notification block. They are as follow: the *Reason* that defines the cause of the notification edition, the *TimeDifference*, the *QuantityDifference* and the *QualityDifference* that are used to report about the difference between the expected results and the realized results in terms of time, quality and quantity, respectively. Furthermore, the InformationFeedback specifies the elements object of the notification, i.e. the Buffer, the order, or the product, through the properties *notifiedOrder*, *NotifiedProduct*, and the *NotifiedBuffer*.

The PaymentNotification is used to inform that the payment was executed. Besides the properties inherited from the Notification block, the PaymentNotification block specifies the reception account through the property *NotifiedAccount*.

### 3.2.2 REPRESENTING THE SC FUNCTIONS

One of the main questions that the SC practitioner asks when modeling his/her SC is “how to represent the functions performed within the SC?”. In order to assist the SC practitioner in answering this question, we introduce in this section the constructs proposed to model these functions. First, we describe the construct defined to capture the SC functions which are the Operation block and then we describe the library that defines a set of Operation blocks capturing the SC functions listed in SCOR.

#### 3.2.2.1 THE OPERATION BLOCK

It is a block used to describe the functions performed within the SC that transform input variables into output variables. The Operation captures the functions through an algorithm that acts on the variables of both the SC structure elements and the input and the output Flows.

The Actor's structure elements are the static parts of the SC (such as Resources, Buffers, MoneyAccount, Contracts...) described within the meta-model mapped in figure 3.2. As shown in figure 3.12, the Operation receives a set of flows (e.g. PurchaseOrder) that are transformed into output flows (e.g. DeliveryOrder). The Operation uses the structure elements to perform the modeled function (e.g. A TransportationResource is used for shipping...).

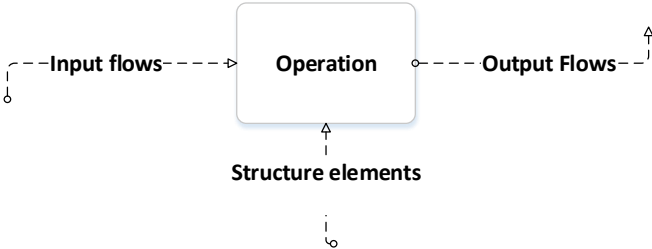


FIGURE 3.12: THE OPERATION INTERACTION

The Operation may behave in various ways, which is captured through the OperationMode block. The OperationMode block is used to model the alternative ways of functioning through describing alternative algorithms (see figure 3.13). The OperationMode has the possibility to call another OperationMode when its conditions of activation are met. For example, if there is a severe delay in a delivery process, a degraded Operation Mode can be triggered for a given Operation.

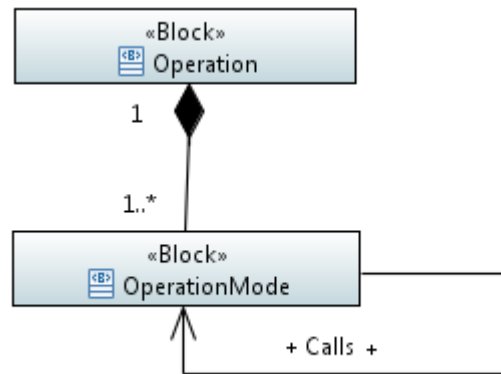


FIGURE 3.13: THE OPERATIONMODE BLOCK DEFINITION DIAGRAM

To define an Operation that models a SC function, we start by setting the modeling assumptions. Then, we define the parameters and the information flows on which the Operation acts. After that, we define the algorithm modeling the functioning, we define the associated methods and finally, we define the used internal variables.

### 3.2.2.2 THE LIBRARY OF SUPPLY CHAIN OPERATIONS

When modeling the Operations of his/her SC the SC practitioner needs to answer the following questions: “What are the parameters of the modeled functions, what are the received and edited information flows, what are the structure elements (Such as Buffers, Resources, and MoneyAccount) used for performing the modeled function ? What is the algorithm that captures the modeled function?”

In order to assist the SC practitioner answering these questions, we provide a library of supply chain Operations. The Operations are defined based on the SCOR reference model. Each Operation is defined based on one or several SCOR Process elements. As stated by the SCC (Supply Chain Council) the SCOR model provides a unified terminology and standard descriptions of Processes that can be used to describe SCs that are very simple or very complex.

SCOR provides a set of processes in four levels of hierarchy that helps to describe an SC from supplier’s supplier to customer ‘customer’, the fourth level which is the implementation level that decomposes Process elements is out of the boundary of the SCOR model. SCC states that it is up to the company to define their specific decomposition of the process elements. SCOR model provides five key processes: Plan, source, make, deliver and return. For each process, a set of categories is provided. For instance, for the process Make, SCOR proposes the categories: make to order, make to stock and engineer to order. Each process is composed of a set of process elements. For instance, the Make process includes the process elements ( “sM1.1/sM2.1 **scheduleProductionActivities**”, “sM1.2/sM2.2 **issue material**”, and “sM1.3/sM2.3 **produceAndTest**”...). The SCOR description of processes is used to define the libraries of SC domain specific constructs.

In order to propose simpler simulation blocks providing a good level of granularity that better cover the Operations performed within the SC, we decide to rearrange the Process elements proposed by SCOR into more convenient Operations blocks (e.g. regrouping a set of SCOR process elements into one Operation block or splitting a SCOR Process element into several

---

Operation blocks) When a function is described with more than one process element and when it can be represented with a single construct we choose to use a single construct. Furthermore, when more than one function is described within a single SCOR process element we choose to separate them into a set of constructs for providing finer granularity for users. For instance, the process element “**sM1.3/SM2.3 ProduceAndTest**” is separated into two constructs PRODUCE and TEST.

In order to keep the traceability between the constructs that we propose and the process elements defined by SCOR, we propose a naming convention issued from SCOR.

SCOR proposes the following symbol to identify the Process elements: [“Process type”, “Policy type”, “Process Element”]. To give an example, the Process element “**sM1.2 Issue Material**” that belongs to the Process “Make” and that follows the policy “Make to stock” is identified with the symbol “**sM1.2**” where “**sM**” refers to the Process type, “1” refers to the policy type and “.2” refers to the Process element.

The adopted naming conventions are as follow:

- The “composition” consists of assembling consecutive Process elements belonging to the same SCOR Process into one Operation. This decision is taken for the case where only the outputs of the last Process element is interesting for imulation and for the case when what is exchanged between the consecutive Process elements is not interesting (e.g. since we are not interested in the output of the Process element “**sD 1.9/sD2.9 Pick Product**” that is transferred to the consecutive Process element “**sD 1.10/sD2.10 Pack Product**”, we decide to combine them into one Operation). The symbol used for this arrangement is defined as follows: First, we put the character “C.” that refers to the word “Composed”. Second, we add the symbols of the combined Process elements successively separated by a hyphen. For instance, we regroup the following Process elements “**sD 1.9/sD2.9 Pick Product**” and “**sD 1.10/sD2.10 Pack Product**”. The resulting Operation is named (C.sD i.9-sD i.10) PICKANDPACK Operation”.
- The “splitting” consists of splitting one Process element into many Operations. This decision is taken in the case where the Process element that describes more than one function, uses different resources and provides different outputs ( e.g. the “**sM1.3/sM2.3 Produce And Test**” Process element describes both the produce function and the testing function ). The symbol used for this arrangement is defined as follows: we just add a number that refers to the rank of the split part at the end of the Operation symbol. For instance, the Process element “**sM1.3/sM2.3 produce and test**” that belongs to the Process “Make” is split into two Operations. The resulting Operations are called, respectively, (sMi.3.1) PRODUCE Operation and (sMi.3.2) TESTOperation.
- The “integration” consists of putting together more than one Process element that shares the same functionality into one Operation. The symbol used for this arrangement is defined as follows: We just separate the symbols of the Process elements by the characters “+A+” which refers to the word assimilation. For instance, we integrate the Process element “**sDRi.4 Transfer defective/ MRO return/ Excess product**” into the Process element “**sS1.4/sS2.4 transfer product**”. The resulting Operation is called (sSi.4+A+sDRi.4) TRANSFER Operation.

The library of Operations is provided in Annex A1 of this dissertation. We made the choice of describing only the Operations that physically handle the products, since they are the most

common Operations in a production process and can be used as a proof of concept for our method. The Operations controlling the physical handling of products are not considered in the provided library for the time being but they can be constructed using our methodology. We use the SCOR description as a basis for the definition of the algorithm of each Operation block. Also, we use the inputs and the outputs defined by SCOR for process elements as a basis for defining the flows of the Operation constructs.

Hence, the current list of Operations provided in the library is shown in Table 3.1.

TABLE.3.1: THE LIBRARY OF OPERATIONS

<b>The Operations library</b>
PRODUCE (sMi.3.1)
TEST (sMi.3.2)
ISSUE MATERIAL (sMi.3.3)
PICKANDPACK (C.sDi.9-sDi.10)
LOADVEHICLE (sDi.11)
SHIPPRODUCT (sDi.12+A+sSRi.5)
RECEIVE (sSi.2+A+sDi.13+A+sDRi.3)
VERIFY (sSi.3+A+sDi.13+A+sDRi.3)
TRANSFER (sSi.4+A+sDRi.4+A+ sD1.8)

In the next section, we present the example of the PRODUCE Operation (sMi.3.1) .

**THE EXAMPLE OF THEPRODUCE OPERATION**

**Definition**

This Operation is responsible for products manufacturing based on a ProductionOrder. It generates an information feedback for the scheduling Operation and modifies the status of the received production order.

**Inputs and Outputs from the SCOR model**

This produce Operation is defined in SCOR as “**sM1.3/ sM2.3 produce and test**” Process element. In our work, we divided this Process element into two Operations: sMi.3 PRODUCE Operation and sMi.3 TEST Operation to provide a finer granularity.

The SCOR model specifies the workflow as input for this Process element, the workflow is received from the previous Process element “**sM1.2/ sM2.2 Issue material**”. In our model, we assume that the information included in the workflow can be described by a production order. The ProductionOrder does not only define the quantity to be produced but also the Buffer of the issued products and the required resources (see figure 3.16). Furthermore, we consider that the workflow contains the information about the resource to be used for production. The SCOR model specifies an output, which is the information feedback, to notify the current state of production. Furthermore, the SCOR model specifies the produced wastes and the workflow as outputs. The produced waste will be considered as an output for the TEST Operation (sMi.3.2) rather than being considered as an output for the PRODUCE Operation (sMi.3.1). This is done since we consider that it is detected when the TEST

Operation is executed. In place of the workflow, the ProductionOrder is used. The ProductionOrder is sent to the next Operation as the output workflow of the PRODUCE Operation after adjusting its status.

Table 3.2 summarizes the inputs and the outputs for the PRODUCE Operation retained from the SCOR model and the variable names that will be used to represent them in the model. Figure 3.14 describes the relations between blocks of the variables and the PRODUCE Operation. The figure 3.15 gives the elements of our meta-model related to the inputs and outputs of the PRODUCE Operation.

TABLE 3.2 RETAINED INPUTS AND OUPUTS FROM SCOR FOR THE PRODUCE OPERATION

SCOR inputs	Retained Inputs and outputs:	Designations
	<b>Inputs</b>	
Workflow	pO [1..*]: ProductionOrder [1..*]	The received production order that informs about what to produce and the required quantity.
	rP: Product	The Product to be produced.
	iB:Buffer [1..*]	The input Buffers where components are taken.
	oB: Buffer	The Buffer where manufactured products are put.
	uR:Resource	The resource used for manufacturing.
<b>SCOR outputs</b>	<b>Outputs</b>	
Information feedback, Waste produced, Workflow.	iFd : Informationfeedback [1..*]	The notification about the execution state.
	pO [1..*]: ProductionOrder [1..*]	The production order with a modified status.

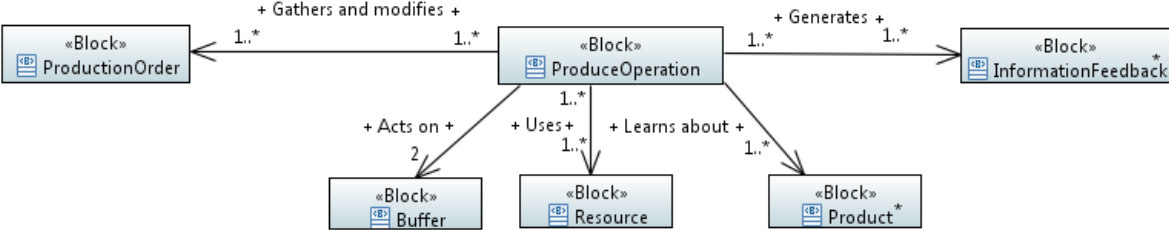


FIGURE 3.14: THE PRODUCE OPERATION BLOCK DIAGRAM



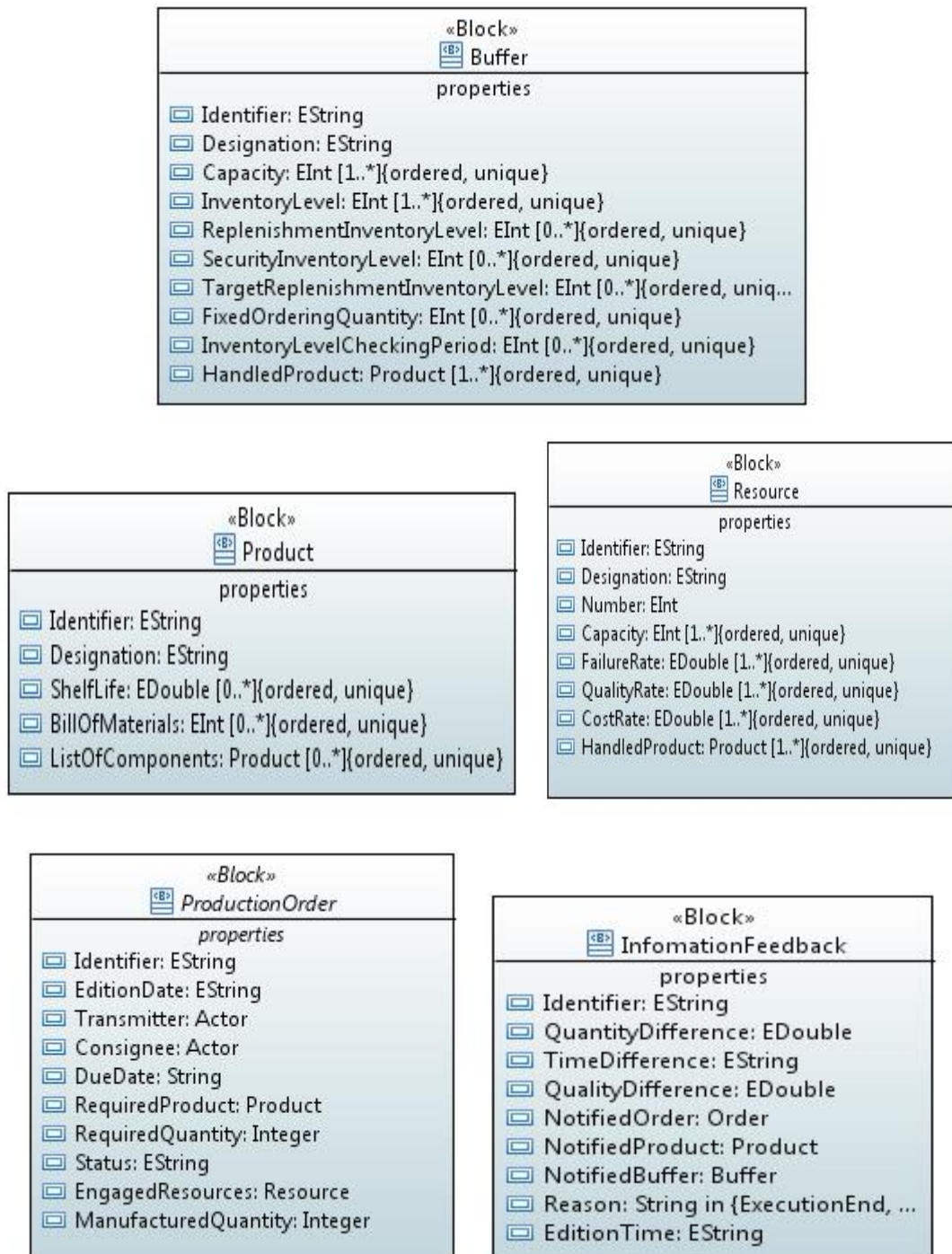


FIGURE 3.15: DETAILS OF THE USED INPUTS AND OUTPUTS FOR THE PRODUCE OPERATION

## Assumptions

For the PRODUCE Operation we assume the following:

- The ProductionOrders are executed by one with respect to the first in first out rule.
- The Operation may abort the manufacturing of a Product when the inventory of the required components is not available. In this case, the Operation sends an InformationFeedback to the scheduling Operation and resumes the production of other products.

- The production capacity is defined by the capacity of the main resource.

### The operation algorithm

The operation in its standard mode receives a set of ProductionOrders. The execution starts when a ProductionOrder is received. If there is more than one order, the orders are released by following the first in first out rule. The quantity to be manufactured is divided into a set of smaller quantities that respects the production resource capacity. Using the bill of materials, the availability of components for the released quantity to produce is checked. If the components are available the production resource is reserved. The production is executed using the available components then the production resource is released. When finishing manufacturing, an InformationFeedback is generated. The InformationFeedback states about the execution end and about the non-achievement of manufacturing in the case of non-availability of components. The algorithm of the Operation is illustrated in the state machine shown in figure 3.16.

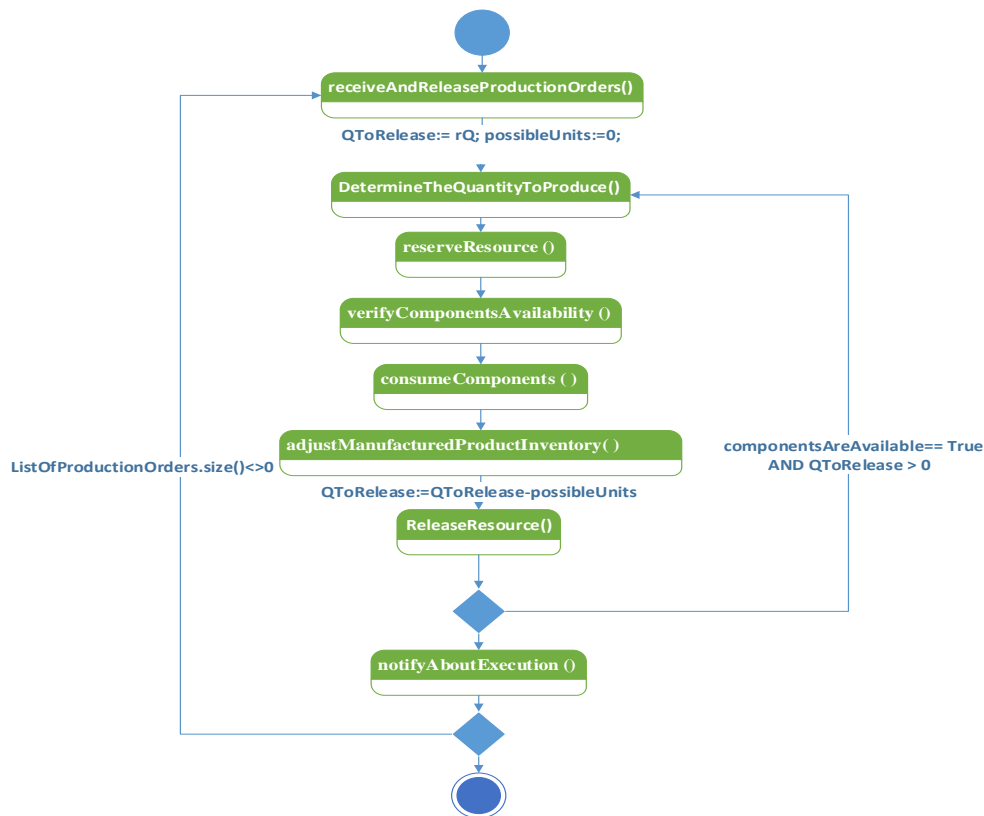


FIGURE 3.16: THE PRODUCE OPERATION STATE MACHINE

### Internal variables

Aside from the variables already mentioned in Table 3.2, we need some internal variables for the algorithm of the PRODUCE Operation. In Table 3.3 we summarize those variables.

TABLE 3.3: INTERNAL VARIABLES USED IN THE PRODUCE OPERATION ALGORITHM

Internal variables	Designations
componentsArAvailable	A Boolean variable which states the availability of components in the input Buffer.
CurrentComponent	An indicator variable that refers to the current checked component.
mQ	Manufactured Quantity
ResourcesAreAvailableAndAll ocable	A Boolean variable that takes “true” if the resources are available.
QToRelease	The quantity to be manufactured by considering the production capacity constraint.
possibleUnits	The quantity to be manufactured by considering the components availability.

### Methods

In the following, we provide the pseudo codes of the procedures (methods) that are used in the algorithm of the PRODUCE Operation (see figure 3.16). They are shown in tables 3.4 to 3.11.

TABLE 3.4 : THE RECEIVEANDRELEASEPRODUCTIONORDERS METHOD

Method 1: Public void receiveAndReleaseProductionOrders ()
<p><b>Description:</b>            This method is responsible for receiving new production orders (ReceivedProductionOrder) and adding them to the list of received orders (ListOfProductionOrders). The received production orders (ListOfProductionOrders) are held. A production order PO is released (based on the first in first out rule) only when the current production order is executed (pO.Status==”executed”).            When the Product order is released (pO:=ListOfProductionOrders.Next() ) its status is set to ( pO.status:= “Released”) .</p>
<p><b>Algorithm:</b>            Public Void receiveAndReleaseProductionOrders () {              Gather();              If (ProductionOrderIsReceived==true) then{                ListOfProductionOrders.Add(ReceivedProductionOrder);                If ( ListOfProductionOrders.size()==1&amp;&amp; ListOfProductionOrders[1].Status &lt;&gt; “Produced” &amp;&amp; ListOfProductionOrders[1].Status &lt;&gt; “ProductionAborted”) then                  {pO:= ListOfProductionOrders [1] ;                } EndIF              } EndIf              / Hold Production orders until executing the current one/              Do { wait ; }              while (pO.Status!= “Produced” &amp;&amp; pO.Status!=“ProductionAborted”)              EndWhile              /Release a new production order/</p>

```

If ( pO.Status=="Produced") then {
    pO:=ListOfProductionOrders.Next();
    pO.Status:="ReleasedForProduction";
    rP := pO.requiredProduct;
    rQ := pO.requiredQuantity;
    uR := pO.engagedResources[1];
    eCT := uR.CycleTime[rP];
}EndIf }

```

TABLE 3.5: THE DETERMINE THE QUANTITY TO PRODUCE METHOD

<b>Method 2:</b> Public void determineTheQuantityToProduce ()
<p><b>Description:</b>  This method is responsible for specifying the required quantity to produce. The Product units are released by batch (releasedProductUnits ) with a size equals (or less) to the capacity of the used resource (uR.Capacity).</p>
<p><b>Algorithm:</b>  Public void releaseTheProductionBatch ()  / For every Units batch of the quantity to be manufactured do  If (QToRelease &lt; uR.Capacity × uR.Number) then {      releasedProductUnits:= QToRelease;  Else { releasedProductUnits:= uR.Capacity × uR.Number }  } EndIf}</p>

TABLE 3.6 : THE RESERVE RESOURCE METHOD

<b>Method 3:</b> Public void reserveResource ()
<p><b>Description:</b>  This method checks the availability of the production resource and its allocability (uR.Available==false Or uR.Allocated==true).  The method allocates the main resource (uR.Allocated==true), if it is available and allocable otherwise, it waits for the resource availability and allocability.</p>
<p><b>Algorithm:</b>  Public Void reserveResource () {  Do {      ResourcesAreAvailableAndAlocable := true;      If (uR.Available==false Or uR.Allocated==true) then {          ResourcesAreAvailableAndAllocable := false; wait();      } EndIF  }while (ResourcesAreAvailableAndAllocable == false) EndWhile  If (ResourcesAreAvailableAndAllocable == true) then {      uR.Allocated:=true ;  } EndIf;}</p>

TABLE 3.7: THE VERIFYCOMPONENTSAVAILABILITY METHOD

<b>Method 4: Public void verifyComponentsAvailability ()</b>
<p><b>Description:</b>            This method checks if there is an available inventory of components (iB.InventoryLevel [currentComponent] ) to cover the production of the releasedProductUnits. The inventory level is compared to the bill of materials coefficient rP.BillOfMaterials[ currentComponent ] ×releasedProductUnits. If the inventory is not available the variable (componentsAreAvailable) is set to false and the variable possibleUnits is set to the minimum possible.</p>
<p><b>Algorithm:</b>            Public void verifyComponentsAvailability () {                componentsAreAvailable:= true;                possibleUnits= releasedProductUnits;                For i from 1 to rP.ComponentsNumber do {                    currentComponent: = RP. ListOfComponents[i]                    If (iB.InventoryLevel[ currentComponent] &lt; rP.BillOfMaterial[ currentComponent ] ×releasedProductUnits) then {                        componentsAreAvailable:=False;                        If (possibleUnits &gt; iB.InventoryLevel[ currentComponent] / releasedProductUnits)                        then {                            possibleUnits := iB.InventoryLevel[ currentComponent] / releasedProductUnits;                        } EndIF                    }EndIF                } EndFor</p>

TABLE 3.8: THE CONSUMECOMPONENTS METHOD

<b>Method 5: Public void consumeComponents ()</b>
<p><b>Description:</b>            This method consumes the inventory of components (iB.InventoryLevel[currentComponent]). The inventory level is reduced by the value of : rP.BillOfMaterials[ currentComponent ] ×releasedProductUnits.</p>
<p><b>Algorithm:</b>            Public void consumeComponents () {                For i from 1 to rP.ComponentsNumber do {                    currentComponent: = RP. ListOfComponents[i]                    iB.InventoryLevel[ currentComponent] :=                    iB.InventoryLevel[currentComponent] - rP.BillOfMaterial[currentComponent ] ×                    possibleUnits;                } EndFor }</p>

TABLE 3.9: THE ADJUSTTHEMANUFACTUREDPRODUCTINVENTORY METHOD

<b>Method 6:</b> Public void adjustTheManufacturedProductInventory()
<p><b>Description:</b>  This method adjusts the inventory level ( oB.inventoryLevel[ rP ]) of the manufactured product. In fact, the method increases the inventory by the possible quantity to produce. Furthermore, the method adjusts the simulation time by adding the execution cycle time to the current simulation time.</p>
<p><b>Algorithm:</b>  Public void adjustTheManufacturedProductInventory( ) {  /Add a delay./  Simulation.currentTime:= Simulation.currentTime+ eCT ;  /Increase the inventory level of the manufactured products./  oB.InventoryLevel[ rP ] := oB.InventoryLevel[ rP ] + possibleUnits;  mQ:= mQ+ possibleUnits; // Increase manufactured quantity }</p>

TABLE 3.10: THE RELEASERESOURCE METHOD

<b>Method 7:</b> Public void releaseResource()
<p><b>Description:</b>  This method is releasing the reserved resource for production.</p>
<p><b>Algorithm:</b>  Public void releaseResource ( ) {  / For every Units batch of the quantity to be manufactured do  uR.Available = true ; }</p>

TABLE 3.11: THE NOTIFYABOUTEXECUTION METHOD

<b>Method 8: Public void notifyAboutExecution ()</b>
<p><b>Description:</b> This method edits an information feedback about the final state of production and modifies the status of the production order (pO.Status).</p>
<p><b>Algorithm:</b>  <pre> Public Void notifyAboutExecution () { iF.Generate(); iF.Identifier:= iF.GenerateId(); iF.EditionTime=currentTime, iF.QuantityDifference=rQ-mQ; iF.TimeDifference=pO.duteDate-currentTime; iF.NotifiedOrder=pO; iF.NotifiedProduct=rP iF.NotifiedBuffer=oB If (mQ &lt;rQ) then {     pO.status="ProductionAborted"; iF.Reason="Abort"; } EndIf If (mQ =rQ) then {     pO.status="Produced"; iF.Reason="ExecutionEnd"; } End if }</pre> </p>

### 3.2.3 MODELING THROUGH ROLES

When modeling the functions of his/her SC, the SC practitioner asks the question: "What operations to use in order to model the functions of the SC?". To assist the SC practitioner in finding a quick answer to this question, we propose to filter the operations of the library based on the capabilities of the SC companies. The construct proposed for this purpose is the Role . Besides the general Role block definition, we provide a library of domain specific roles.

#### 3.2.3.1 THE ROLE BLOCK

A Role defines a logical grouping of Operations according to a kind of activity (e.g. store, make). The grouped operations define a capability to provide services to SC. It is used to get a filtered set of operations that fits with what the Actor does.

#### 3.2.3.2 THE ROLES LIBRARY

To facilitate the modeling of the SC processes, we assist the SC practitioner by providing a set of domain specific roles that assist him in picking the operations that fits best with his processes.

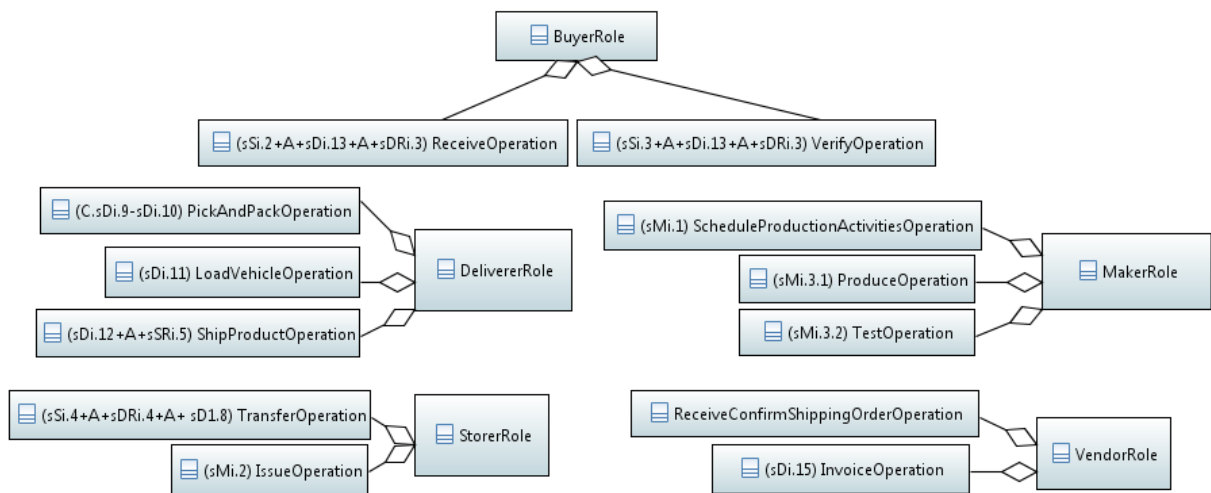
By studying the SC processes described in SCOR, we identify five capabilities of the SC Actors that deserve to be presented with Roles.

- The Storer Role: It regroups the operations related to managing and transferring the products inventories within a factory.
- The Vendor Role: It regroups the operations related to managing and executing the commercial activities to sell products and to return non-conforming products.



- The Buyer Role: It regroups the operations related to managing and executing the commercial activities to source products, to receive and to verify them and to collect returned non-conforming products.
- The Maker Role: It regroups the operations related to managing and executing the activities of products manufacturing.
- The Deliverer Role: It regroups the operations related to managing and executing the products delivery to their destinations, including returning the non-conforming products. The operations attached to each Role are shown in the block definition diagram of figure 3.17. In this figure, only the operations which are currently available in our Operations Library

(Annex A1) are represented. It is clear that other operations (e.g. to represent Planning and/or Scheduling activities) can be added for each Role, when the Operations Library is enriched by such operations. Indeed, for each Role, we can express the correspondence between operations and their relative SCOR Process elements. The correspondence between SCOR Process elements and the maker role is shown in Table 3.12, as an example. Some of the



process elements are already used to define the library of Operations (such as “sM1.3/ sM2.3: Produce and Test” used to define the Operation PRODUCE and the Operation TEST). Other process elements are still to be explored to extend the current Operations library. We refer the readers to Annex A2 for the correspondences for the remaining Roles.

FIGURE 3.17: SC ROLES BLOCK DEFINITION DIAGRAM

TABLE 3.12: CORRESPONDENCE BETWEEN THE MAKER ROLE AND THE SCOR PROCESS ELEMENTS

Maker role Process elements	Process categories
“sM1.1/ sM2.1: Schedule Production Activities”	Make Process (sM)
“sM1.3/ sM2.3: Produce and Test”	
“sM1.4 /sM2.4: Package”	
“sM1.7/ sM2.7: Waste Disposal”	
“SP1.1: Identify, Prioritize and Aggregate Supply Chain Requirements”	Plan supply chain Process (sP1)
“SP1.2: Identify, Prioritize and Aggregate Supply-Chain	

<b>Resources”</b>	
<b>“SP1.3: Balance Supply Chain Resources with SC Requirements”</b>	
<b>“SP1.4: Establish &amp; Communicate Supply-Chain Plans”</b>	
<b>“SP3.1: Identify, Prioritize and Aggregate Production Requirements”</b>	Plan make Process (sP3)
<b>“SP3.2: Identify, Assess and Aggregate Production Resources”</b>	
<b>“SP3.3: Balance Production Resources with Production Requirements”</b>	
<b>“SP3.4 Establish Production Plans”</b>	

### 3.2.4 MODELING THE SC PROCESSES

Another question that needs to be answered is “how to connect operations to model the Processes of its SC?”. To assist the SC practitioner finding a quick answer to this question, we propose to form the Processes through specifying the Operations to be connected together and the flows to be transferred between the connected Operations. We propose the **Process** block for this purpose.

The **Process** organizes a set of operations via the **OperationsInteraction** that specifies the connection details. The block definition diagram of the Process is shown in figure 3.18. The Process block has two properties for naming (*identifier* and *designation*) and a property that defines the list of organized operations. The **OperationsInteraction** defines the connected operations through the properties *OperationConnexionIn* and *OperationConnexionOut* and the transferred information flows through the property *transferredFlow*.

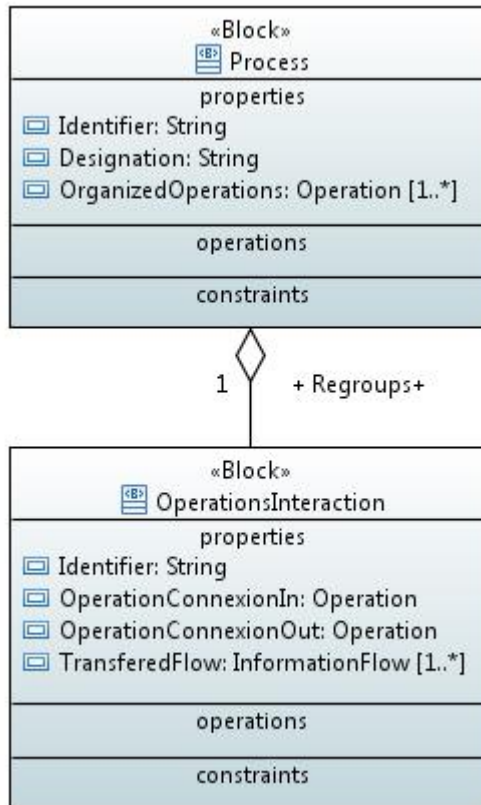


FIGURE 3.18: THE PROCESS BLOCK DEFINITION DIAGRAM

### 3.2.5 ILLUSTRATIVE EXAMPLE

We illustrate the approach through the example of an automotive parts supply chain. The automotive parts SC works as follows: the focal company is a damper manufacturer (D) that sells dampers to an automotive wholesaler (W) and for a car manufacturer (C). The wholesaler (W) sells them to retailers (R). End customers (E) buy those parts from retailers to repair vehicles. We model each SC Actor’s behavior by selecting a set of roles from the domain specific roles library. The manufacturer is responsible for producing automotive parts in order to satisfy customers’ demand. This functionality is modeled through the Role Maker. The manufacturer sources materials for production, which is modeled through the Role Buyer. He is responsible for his own inventory management, deliveries and sales; therefore we assign the Roles Storer, Deliverer and Vendor, as well. The other Actors are defined similarly (see, Table 3.13).

TABLE 3.13: ACTORS’ ROLES’ CONFIGURATION

	Buyer	Vendor	Deliverer	Storer	Maker
Manufacturer (D)	X	X	X	X	X
Cars manuf. (C)	X	X	X	X	X
Wholesaler (W)	X	X	X	X	
Retailers (R)	X	X		X	
End-customers (E)	X				

When selected, each Role gives the Actor access to a set of Operations, since the Roles specify the Actor capabilities. An example of the Operations accessible for the Vendor Role of the damper manufacturer (D) is given in Table 3.14.

TABLE 3.14: EXCERPT OF DELIVER OPERATIONS RELATIVE TO THE MANUFACTURER (D)

Instantiated operations
The (sDi.11) LOADVEHICLE Operation
The (sDi.12+A+sSRi.5) SHIPPRODUCT Operation
The (C.sDi.9-sDi.10) PICKANDPACK Operation

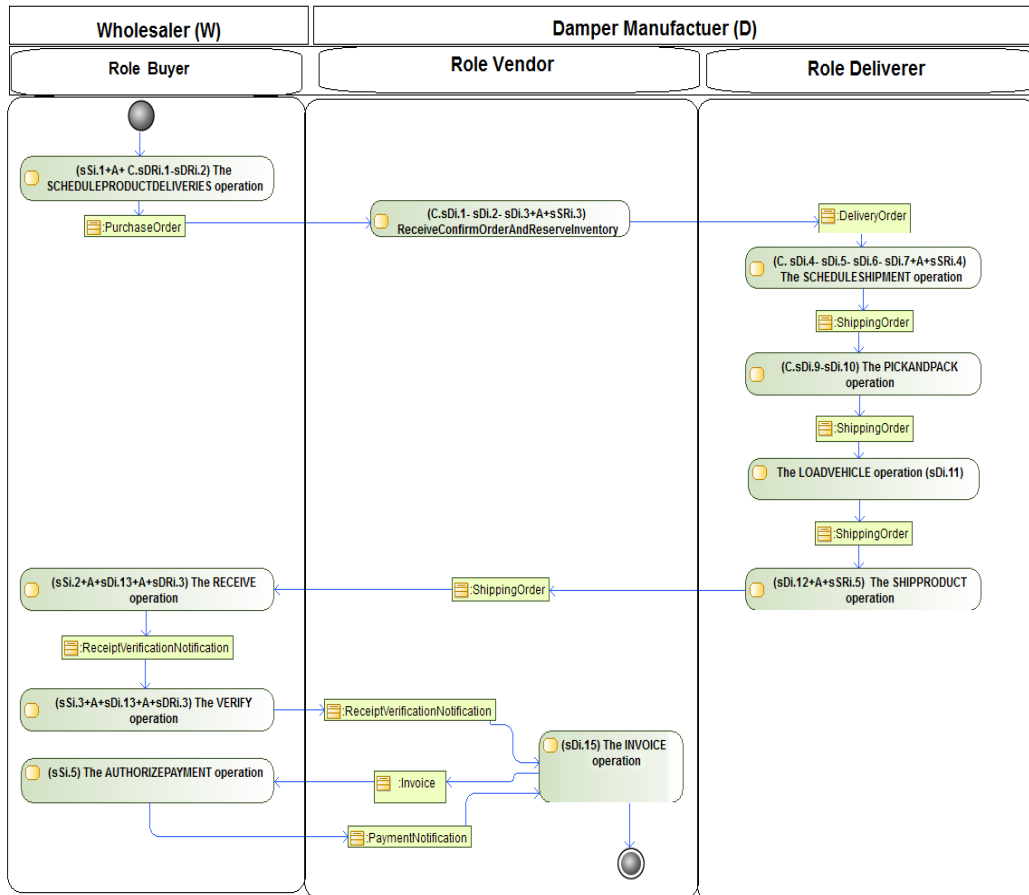


FIGURE 3.19: TRADING GOODS PROCESS

The next step is to model the SC processes. To do that, the Operations available in the selected Roles are connected together. The Operations' instances are then customized. This is through setting their properties values and through linking them with the information flows.

To provide an example of a process that is modeled through the provided Operations, we present the example of the trading goods process. The modeled process is shown in figure 3.19 where the wholesaler buys dampers from the damper manufacturer. Indeed, figure 3.19 shows an activity diagram where each activity represents an Operation. The object nodes shared between activities represent the information flows. The Process starts with an object node that is sent to the damper manufacturer (via the role Vendor) by the wholesaler (via the role Buyer). The object node is a PurchaseOrder that specifies the details of what is requested (product, quantity, leadTime...). After the reception of the PurchaseOrder the damper

---

manufacturer prepares the requested command and ships the products through the Operation SHIPPRODUCT (sDi.12+A+sSRi.5) (available in the role Deliverer). The products are received by the wholesaler; through the Operation RECEIVE (sSi.2+A+sDi.13+A+sDRi.3). The wholesaler confirms the reception of the products after verification and proceeds to pay the Invoice issued by the damper manufacturer.

### 3.3 MODELING THE SC RISKS

After modeling the structure and the behavior of the SC, the SC practitioner needs to model the risks capable of threatening his/her SC. To model the risks the SC practitioners need to find an answer to the following questions: “How to model the numerous risks capable of threatening the SC?”

In order to assist the SC practitioner answering these questions, we provide a meta-model of risks. This meta-model describes a set of easily customizable SC risk modeling constructs and how they can be connected with the constructs of the SC meta-model.

As mentioned in chapter 1, SC risk is defined as follows: “SC risk is a scenario originating from a fault (internal or external to SC) which incurs negative effects on the objective of more than one element of the SC. The realization of the scenario depends on both the fault realization and the current SC states”.

As stated in chapter 2, there are numerous risks threatening SCs. In the simulation based risk literature, most of the studied risks are case specific. To overcome this gap, we will provide a generic modeling constructs for each group of risks having similar features. To identify the groups of SC risks having common features we adopt the categorization by (Saleh Ebrahimi et al (2012)) that classifies risks based on their impacts on SC models. For each risk category we define related parameters and its relation with the rest of the SC. The adopted categories are as follow:

- **Operation Mode risks**

They are the risks which redefine the functioning of an Operation. They act by activating a degraded functioning Mode of the Operation. For instance, a supply delay due to shipping dysfunction is a functional risk which changes the mode of the SHIPPRODUCT Operation to a degraded Mode where an additional delay is added to generated transportation time.

- **Property change risks**

They are the risks which act on a property of an object (Flow, Resource, Actor...). The value of the property changes when the risk occurs. For instance, Decrease Of Production Capacity is a PropertyChangeRisk which modifies the value of the attribute *Resource.Capacity*.

- **Object destruction risks**

They are the risks which eliminate an object such as (resource, Actor...). When the risk occurs the concerned object is deleted. For instance, Supply Cease is an object destruction risk which deletes the supplier Actor.

(Saleh Ebrahimi et al (2012)) also provide a crosscheck with the risks literature as shown in Table 3.15 to verify how well the proposed classes cover the risks mentioned in the literature.

TABLE 3.15: SC RISKS LITERATURE CROSSCHECKED WITH THE PROPOSED RISK CATEGORIES (SALEH EBRAHIMI ET AL.(2012))

Types	Risks
Operation mode risks	Quality errors discovered internally
	Capacity issues internally (manufacturer)
	Forecast errors
	Delayed production (internal)
	Delayed shipment (sending)
	Quality errors from supplier when delivered
	Quality errors due to transit damage/excessive handling
	Material shortage
	Capacity issues at suppliers
	Lost goods while shipping
	Missing parts at delivery
	Down-prioritization
	Demand volatility
	Financial stability of partners (customers)
Property change risks	Accidents (internal) (fire/machine breakdown, etc.)
	Cycle time volatility
	Labor disputes (internal)
	Overstocking
	Material cost increase
	Added or raised taxes/tolls
	Exchange rate volatility
	Competition causing force to decrease prices
Object destruction risks	Supplier bankruptcy
	Customer bankruptcy
	Natural disasters
	Route blockades
Covered by other risks	Product obsolescence (covered by forecast errors)
	Legal liabilities (covered by other risks (delay, financial stability ...))

Based on this categorization, we propose SC Risk constructs are shown in figure 3.20. The block definition diagram shows how the Risk blocks impact the SC elements. The objectDestructionRisk acts on the objects (MoneyAccount, Buffer, Facility, Actor, InformationFlow, Resource, and Route). For example, the Supplier Bankruptcy is a type of ObjectDestructionRisk. When it occurs, the related supplier will be removed (or inactivated) from the SC model. The PropertyChangeRisk acts on the properties of those impacted elements. For example, the exchange rate volatility or internal labor disputes are some PropertyChangeRisks, which can modify the value of a property when they occur. For instance, exchange rate volatility may impact the *Price* property of a contract. Similarly, internal labor disputes (or strikes) may impact the *Capacity* of Human Resources. Finally, the OperationModeRisk defines another functioning for the SC operations. For instance, when

there is a capacity issue at a supplier, the PRODUCE Operation related to this supplier would switch to a degraded Operation Mode.

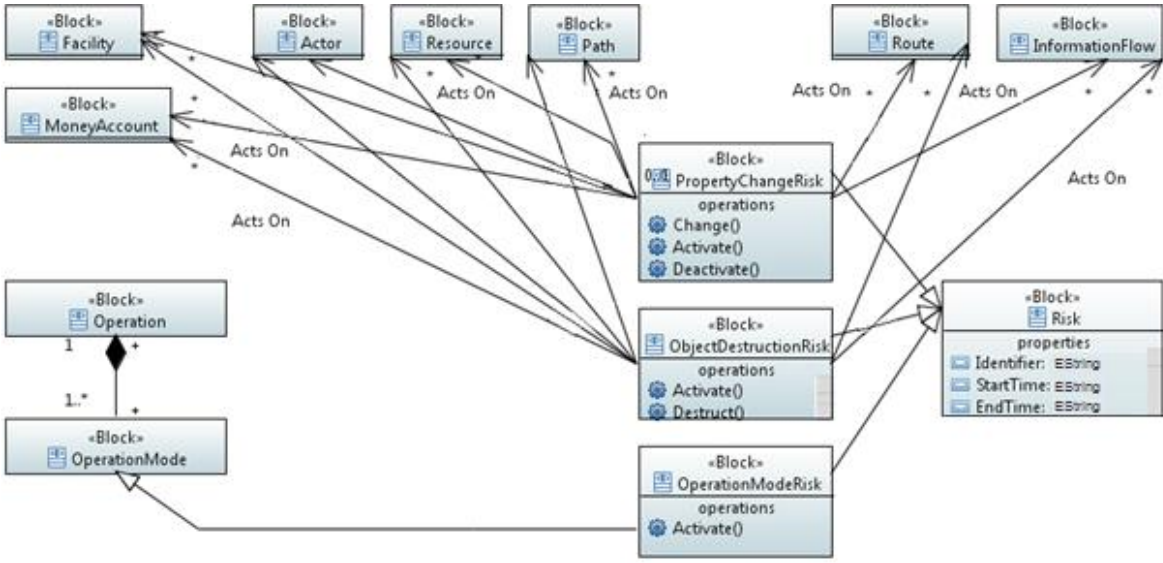


FIGURE 3.20: SC RISK META-MODEL BLOCK DEFINITION DIAGRAM

## CONCLUSION

In this chapter, we present the developed modeling framework for simulation that assists the SC practitioners in modeling their SCs and the risks threatening it.

The modeling frameworks developed in the literature are not always loyal to the SC domain. We, therefore, propose a metamodel and a library of building blocks specifically designed for modeling SCs and the inherent risks, by relying on the SCOR reference model. More precisely, we propose a library of Operations based on the SCOR process elements. We propose a metamodel for the structure of the SC based on the structure elements on which the SCOR process elements act and we propose a full description of the SC flows exchanged between the SCOR process elements defined as a set of modeling building blocks.

We use SysML metamodeling language to express the proposed metamodel. This is to overcome the communication problems and the lack of expressiveness encountered in the literature frameworks. To this end, a profile SysML is created to get advantage of its object oriented paradigm.

Furthermore, we define a metamodel for risks. This is to assist the SC practitioners in analyzing their risks and to deal with the weak integration of risks within the frameworks proposed in the literature. Namely, we define Risk modeling constructs for a set of generic risks representing groups of specific risks. The risk groups are defined based on the categorization of (Saleh Ebrahimi et al. 2012).



---

**CHAPTER** **4**

**A SIMULATION FRAMEWORK:  
CREATING AND EXPERIMENTING  
THE SIMULATION MODELS**

---

## CHAPTER 4: A SIMULATION FRAMEWORK: CREATING AND EXPERIMENTING THE SIMULATION MODELS

### SUMMARY

In this chapter, we present the simulation framework proposed to assist the SC practitioner in creating a simulation model enabling performance analysis. The framework provides the user with a translation guideline that explains how to translate the different elements of the conceptual model into simulation elements (both for structural and behavioral elements) and how to modify the simulation model to experiment risk scenarios. The translation is illustrated with examples based on ARENA. For assuring the genericity of the translation, we provide the correspondance between the ARENA elementary modules and a set of general modules found in most of the common simulation software.

We first provide the guidelines for the translation of the structure elements into simulation variables and into a set of predefined simulation software modules. This translation is illustrated with the examples of the Buffer and the Resource constructs. Second, we explain the translation of Operations and SC Flows into flow charts of elementary simulation modules connected together and flowing entities, respectively. The translation of the Operation construct is illustrated through the example of the Produce Operation in ARENA. Third, we explain the translation of the risk modeling constructs into a set of simulation modules. Finally, we provide guidelines for conducting experimentation for risk analysis, namely, the modifications to implement for each situation (e.g, a new policy, a different SC structure, a specific risk...).

### INTRODUCTION

When built by the SC practitioner, the conceptual model provides a complete description of the SC. Therefore, it is a precious help in providing a structured walk-through for the SC practitioner. However, the conceptual models cannot be used directly for performances analysis since they are static in nature and do not calculate the dynamic evolution of the system states' variables. So, the conceptual model needs to be translated into an executable model (e.g. a simulation model) for performance analysis. However, making this move is not straightforward. This requires an expertise in simulation formalisms, hardly possessed by SC practitioners. The aim of this chapter is to provide assistance to the SC practitioners in executing this task.

In this chapter, we present in detail how to translate the conceptual model into a simulation model and how to experiment it. We provide tools that assist the SC practitioner to this end. We recall, in figure 4.1 the different steps to go through and the provided tools. The framework proposal is as follows:

- (1) Create the simulation model by setting the values of the simulation variables, by instantiating the simulation modules, by connecting them and by parameterizing them.

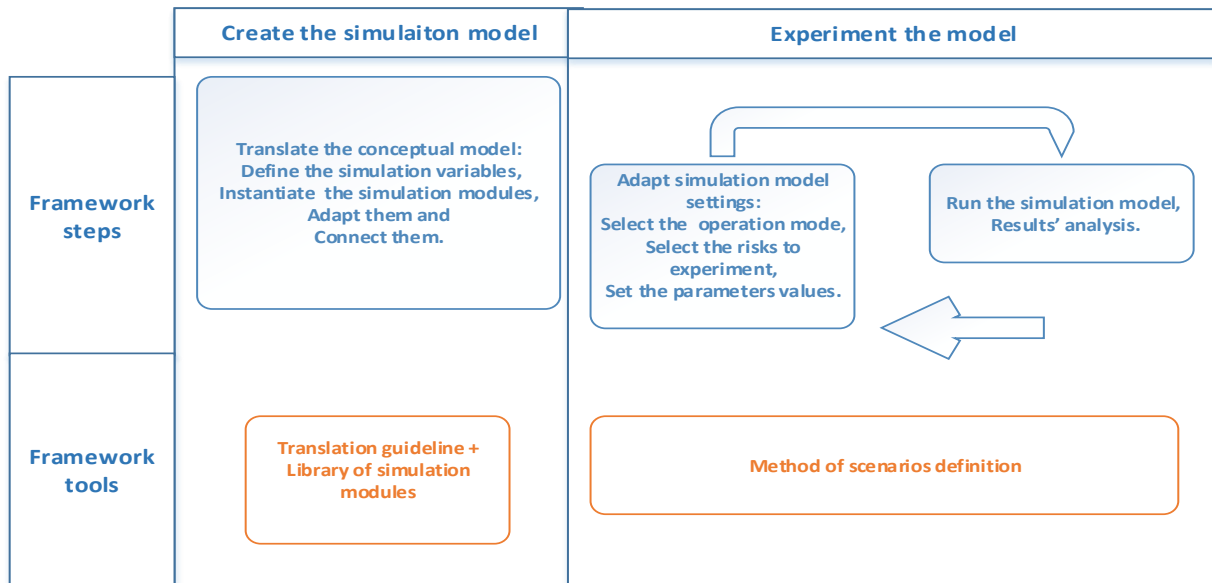


FIGURE 4.1: THE FRAMEWORK SUPPORT TO SIMULATE THE SC AND THE ASSOCIATED RISKS

The framework supports this step by providing a generic translation guideline for creating the simulation variables for the SC structure and for creating simulation modules and simulation flow entities for the SC behavior and for the SC risks.

(2) Experiment the simulation model. We explain how to define a scenario and to drive an experiment on the created model. The framework supports this step by describing the adaptations to be done to represent a given scenario (e.g., a risk scenario) and for monitoring performances.

In this chapter, we present each step leading to the simulation model. We successively present the translation of the SC structure, of the SC behavior and of the risks to be simulated. We finally comment on the experiment to be conducted with the generated model.

## 4.1 THE CREATION OF THE SIMULATION MODELS

Usually, in this step, the SC practitioner uses the modules provided by the simulation software for representing each of the elements of its real SC. We have shown in chapter 2 that DES are good candidate techniques to express SC behavior. Nevertheless, creating the simulation model using the syntax of the current simulation software requires extensive effort. Even though very useful and user-friendly compared to programming languages (e.g. Java, C++,...), the simulation softwares still require modeling know-how and an effort to learn the syntax and the semantics of the chosen DES tool. Moreover, the building of a model from basic construction elements provided in software requires also abstraction skills and quite a long time to build a full model. The reason for this is that the software constructs are of low level: we mean by this that they do not integrate an abstraction of the domain of interest. For the same reason, the SC practitioner needs to instantiate numerous constructs for building its simulation model. The practitioner may, therefore, face both a skill and required effort problem to deploy simulation. In order to assist the SC practitioner for rapidly creating simulation models, we provide a translation guideline that permits converting a conceptual model into a simulation model using discrete event simulation software. The translation guideline is explained through the example of ARENA simulation patterns.

---

The translation guideline defines the correspondence between the modeling constructs and the simulation elements that translate them

First, we explain the translation of the conceptual model of the SC structure into simulation variables and then we explain the translation of the conceptual model of the SC behavior into simulation modules and their relative variables.

#### 4.1.1 TRANSLATION OF THE SC STRUCTURE

There are two methods to translate a structure construct.

- Using the simulation variables

Each property of the SC structure construct is translated into a simulation array variable. The first column of the array variable is used to identify the construct in question. The other columns are used for referring to the other constructs in relation. For instance, the *cycleTime* property of the Resource construct is translated into an array variable of two dimensions where the first column contains the identifier of the related resource and the second contains the identifier of the products handled by this resource.

- Using the predefined simulation modules

Some of the simulation software provides specific simulation modules to present physical constructs. When predefined simulation modules exist it is better to use them to get advantage of the predefined variables and functions. For example, ARENA provides a simulation pattern for defining the Resource construct.

We illustrate the translation method through presenting the translation of two SC structure elements which are the Buffer and the Resource constructs. Each structure of the SC Actor can be modeled as a set of Buffers and a set of Resources. The translation can be expanded easily to translate other elements of the SC structure.

##### 4.1.1.1 BUFFER CONSTRUCT TRANSLATION EXAMPLE

The Buffer construct to be translated is shown in figure 4.2. For each property of the Buffer, we declare a simulation variable. The declared simulation variables are as follow:

- Capacity (BufferIdentifier, ProductIdentifier) : Array of integers of 2 dimensions.
- InventoryLevel (BufferIdentifier, ProductIdentifier): Array of integers of 2 dimensions.
- ReplenishmentLevel (BufferIdentifier, ProductIdentifier): Array of integers of 2 dimensions.
- SecurityLevel (BufferIdentifier, ProductIdentifier): Array of integers of 2 dimensions.
- TargetLevel (BufferIdentifier, ProductIdentifier): Array of integers of 2 dimensions
- FixedOrderingQuantity (BufferIdentifier, ProductIdentifier): Array of integers of 2 dimensions.
- CheckingPeriod (BufferIdentifier, ProductIdentifier): Array of integers of 2 dimensions.
- HandledProductsNumber (BufferIdentifier): Array of integers of 1 dimension.
- Active (BufferIdentifier) : Array of integers of 1 dimension (This additional variable is used to set the activation state of the Buffer instance).

The second step is to declare the instances of the studied SC model. Hence, we start by specifying the identifier of the Buffer's instance and then for each value of the Buffer instance we specify a value in the correspondent case of the array variable.

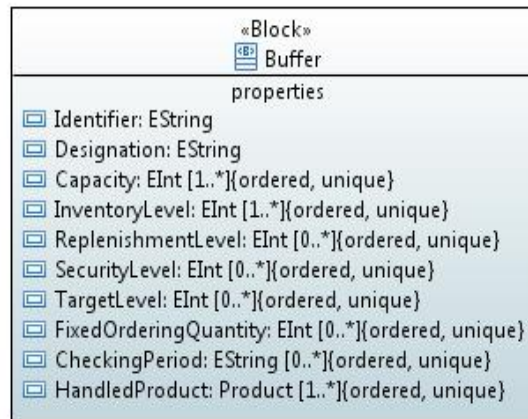


FIGURE 4.2: THE TRANSLATED BUFFER CONSTRUCT

#### 4.1.1.2 THE RESOURCE CONSTRUCT TRANSLATION EXAMPLE

The Resource construct to be translated is shown in figure 4.3. Here, we use a predefined simulation module of the ARENA simulation software for declaring the Resource. We use the predefined attributes provided by the ARENA Resource module for setting the values of some of the properties of our Resource construct.

To translate the *identifier* property and the *number* property of the Resource instance we assign their values to respectively the values of the predefined attributes “name” and “capacity” of the ARENA Resource module. Since the values of the predefined capacity *attribute* of the ARENA resource module can not be set by Product we choose to use this attribute to declare the *number* property of the Resource instance.

Furthermore, like for the Buffer construct, we start by translating the variables relative to the construct of interest. Hence, for each property of the Resource construct, we declare a simulation variable. The declared simulation variables are as follow:

- Capacity: Array of integers of 2 dimensions (ResourceIdentifier, ProductIdentifier).
- CycleTime: Array of reals of 2 dimensions (ResourceIdentifier, ProductIdentifier).
- FailureRate: Array of reals of 2 dimensions (ResourceIdentifier, ProductIdentifier).
- QualityRate: Array of reals of 2 dimensions (ResourceIdentifier, ProductIdentifier).
- CostRate: Array of reals of 2 dimensions (ResourceIdentifier, ProductIdentifier).
- HandledProductsNumber: Array of integers of 1 dimension (ResourceIdentifier).
- Active (ResourceIdentifier) : Array of integers of 1 dimension (This additional variable is used to set the activation state of the Buffer instance).

Then we assign the values of the properties of the Resource instance to the corresponding cases of the declared array simulation variables.

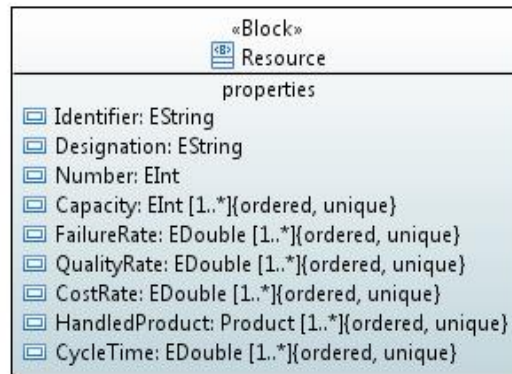


FIGURE 4.3: THE TRANSLATED RESOURCE MODELING CONSTRUCT

#### 4.1.2 TRANSLATION OF THE SC BEHAVIOR

To complete the translation of its SC model, the SC practitioner needs to translate its SC behavior model into simulation modules. Hence, we translate the behavior constructs listed in Table 4.1.

TABLE 4.1: THE TRANSLATION OF THE BEHAVIOR MODELING CONSTRUCTS

Behavior modeling constructs	Simulation modules
Operation modeling construct	An Operation simulation module.
Flow modeling construct	A set of flowing simulation entities.
Process modeling construct	A simulation flowchart

In next paragraphs, we describe the translation of each of the behavior modeling constructs.

##### 4.1.2.1 TRANSLATION OF THE OPERATION MODELING CONSTRUCTS

To simulate the functions of its SC, the SC practitioner needs to translate the instantiated Operation constructs. Here, we explain the translation through providing the translation relative to ARENA simulation software. For the elements belonging to the operations library we developed, an ARENA pattern is provided. For newly developed Operations, sub model shall be constructed in analogy to the logic we used for the library elements. Other translation may be found for other DES simulation technologies.

We illustrate the translation of the Operation modeling constructs into ARENA simulation modules through the example of the PRODUCE Operation construct. The modeling construct PRODUCE operation is explained in section 3.2.2.2.

The Operation construct is translated into a simulation module. The variables defining the characteristics of the Operation construct are declared as parameters. Those parameters are declared as global variables using an ARENA dialog box.

We use the ARENA platform for modules developpement to create a dialog box for the PRODUCE Operation. The dialog box is shown in figure 4.4.

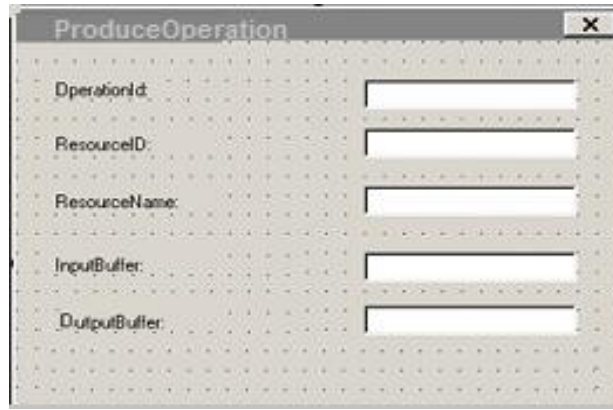


FIGURE 4.4: THE ARENA DIALOG WINDOW FOR PARAMETRIZING THE SMI.3.1 PRODUCE

For the PRODUCE Operation, the dialog box includes the following variables: The *identifier* of the Produce Operation, the *name* and the *identifier* of the manufacturing resource and finally the input and the output Buffers. These are the input variables mentioned in the PRODUCE Operation definition made in section 3.2.2.2. The users need to instantiate the PRODUCE Operation construct.

The second step is to translate the algorithm of the operation. We set the internal variables of the Operation. The declared variables are shown in Table 4.2.

TABLE 4.2: DECLARATION OF A PART OF THE INTERNAL VARIABLES OF THE PRODUCE OPERATION

Internal variables	Designations
componentsAreAvailable : Boolean Initialization: True	It states about the availability of components in the input Buffer.
CurrentComponent : integer Initialization: 1	It refers to the current checked component.
mQ : int Initialization: 1	It calculates the Manufactured Quantity.
ResourcesAreAvailableAndAllocable : Boolean Initialization: True	It takes “true” if the resources are available and allocable.
QToRelease: integer Initialization : PO.requiredQuantity(ProductIdentifier)	The quantity to be manufactured by considering the production capacity constraint.
PossibleUnits: integer Initialization : QToRelease	The quantity to be manufactured by considering the components inventory availability.

Then, we translate the algorithm of the PRODUCE Operation into a simulation flowchart using the ARENA flow modules. We start by declaring the internal variables as a set of ARENA simulation variables, and then we create a flowchart (called a logic diagram in ARENA) that connects a set of ARENA flow modules and submodels. Those modules and submodels are programmed with the algorithms of the PRODUCE operation. The resulting ARENA logic diagram is shown in Figure 4.5.



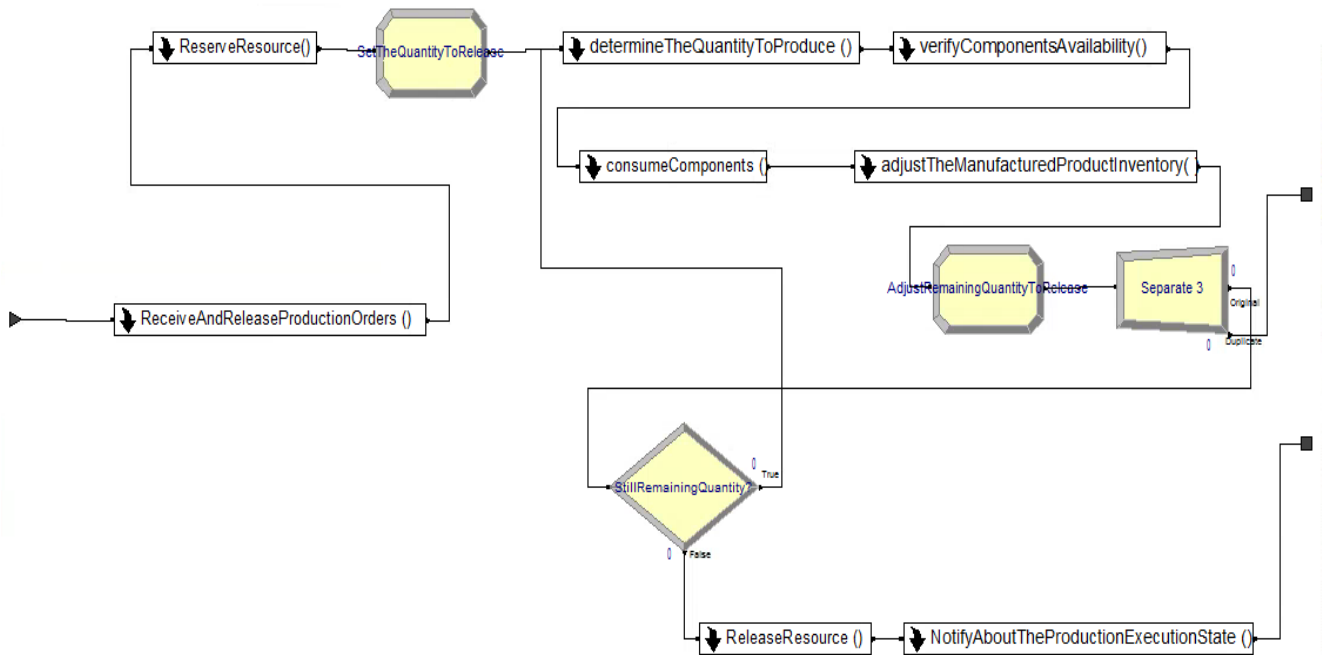


FIGURE 4.5: FLOWCHART OF THE SM1.3.1PRODUCE OPERATION ARENA SIMULATION MODULE

The logic diagram is formed of a set of sub-models that translates the algorithm's methods. The first sub-model <ReceiveAndReleaseProductionOrders> holds received production orders and release them by following first in the first out rule, the released order is received by the sub-model <ReserveResource> that seizes the production resource, then the sub-model <DetermineTheQuantityToProduce> separates the requested quantity into a set of lots respecting the resource capacity. The availability of a sufficient quantity of components is then checked through the sub-model <VerifyComponentsAvailability>. The available quantity of the components is consumed through the sub-model <consumeComponents> and the inventory of the manufactured products is adjusted through the sub-model <adjustTheManufacturedProductInventory>. This cycle is repeated until manufacturing all the requested quantity or consuming the entire available components' inventory. The seized resource is then released through the sub-model <ReleaseResource> and a notification is edited and sent to the other Operation instances.

To give an example of the simulation flowchart of the sub-model translating a method of the algorithm, we describe the sub-model of the method consumeComponents (). The sub-model flowchart is shown in figure 4.6.

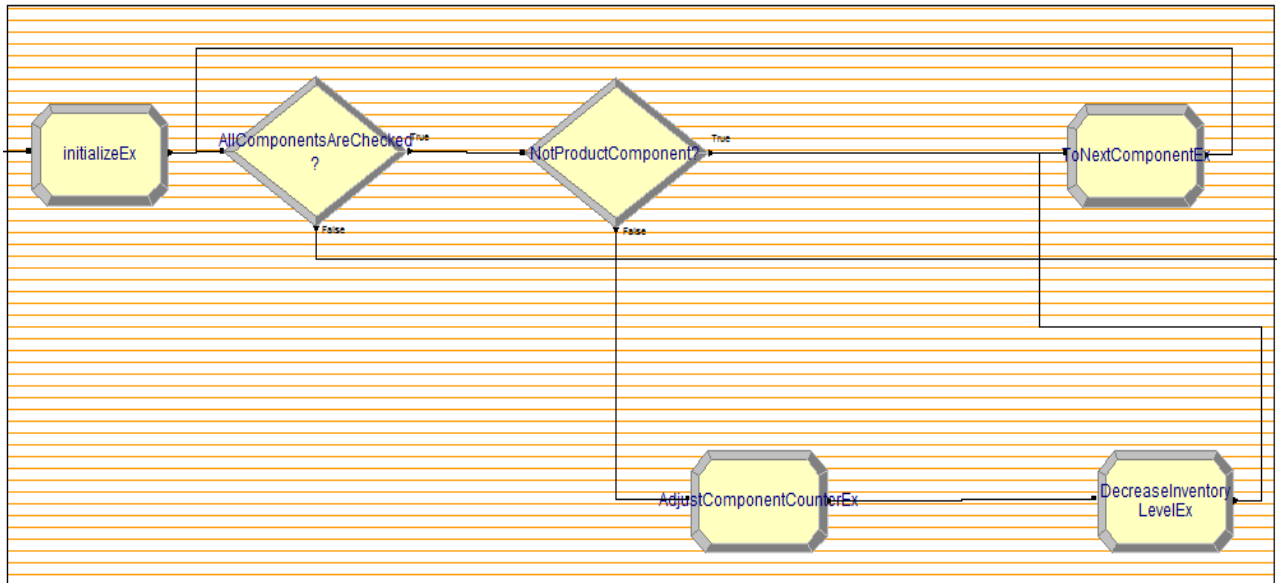


FIGURE 4.6: FLOWCHART OF THE CONSUMECOMPONENTS () METHOD ARENA SUBMODEL

Using the variable *billOfMaterial*, the sub-model selects components one by one from the list of product's components and then decreases the level of inventory for each selected component. To this end, a set of ARENA modules is used. For example, the first <Assign> module is used to initialize the variables that monitor each component's inventory. It is also used to initialize the variable that informs about the selected component by setting it to the first component. Another <Assign> module is used to decrease the inventory of components named <DecreaseInventoryLevelEx>.

For the other operations, we provide the SC practitioner with a library of Operation simulation modules called template in ARENA. An overview of the proposed templates is shown in figure 4.7.

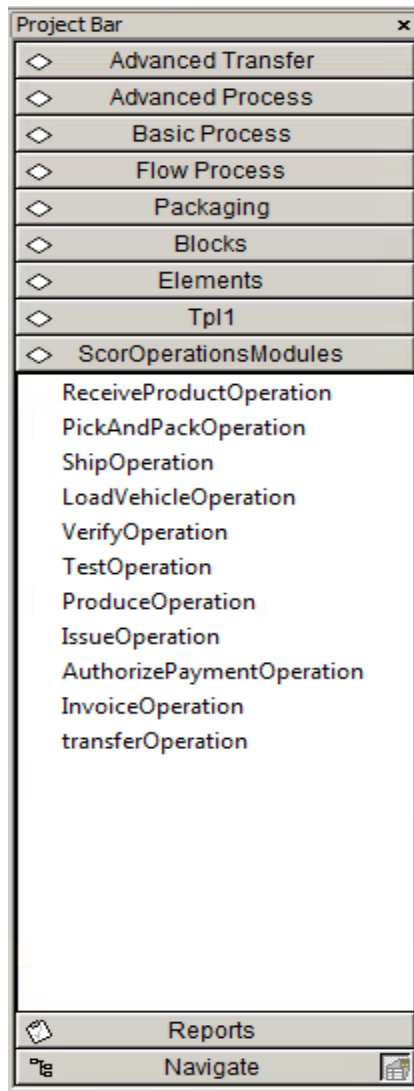


FIGURE 4.7: THE TEMPLATE OF THE SCOR OPERATION SIMULATION MODULES

#### 4.1.2.2 TRANSLATION OF THE FLOW MODELING CONSTRUCTS

After the translation of the Operations, the SC practitioner needs to translate the flow objects that are exchanged between Operations. The flow objects are translated as a set of simulation flow entities. Most of the current DES simulation software include the flow entity construct that needs to be customized by the user.

Hence, each property of the flow construct is translated into an attribute of the simulation flow entities. We illustrate the translation by the example of the PurchaseOrder flow object in ARENA. The PurchaseOrder Block is given in figure 4.8.

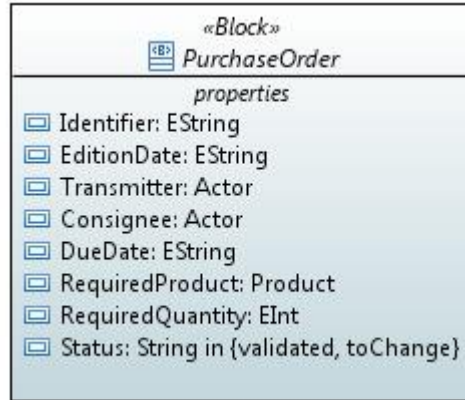


FIGURE 4.8: THE PURCHASEORDER

We translate each property of the PurchaseOrder into a flowing entity attribute. Some simulation software (such as ARENA) do not give the possibility to declare a flowing entity attribute of type array. Hence, the solution is to use a simulation variable of type array where we reserve the first case for the identifier of the flowing entity in question.

The properties of the PurchaseOrder are translated into the following set of ARENA flowing entity attributes:

- “*Identifier*”: Attribute of type String,
- “*EditionDate*”: Attribute of type String,
- “*Transmitter*”: Attribute of type String,
- “*Consignee*”: Attribute of type String,
- “*RequiredProduct*”: Attribute of type String,
- “*RequiredQuantity*”: Attribute of type Integer,
- “*Status*”: Attribute of type String.

#### 4.1.2.3 TRANSLATION OF THE PROCESS MODELING CONSTRUCTS

To translate the process model into a simulation model, we rely on the connectors that are provided by most of the DES simulation software. The connectors define an interaction between two simulation modules that transfers flows.

As an example, we provide the ARENA model of an automotive SC process presented in section 3.2.5. This one is presented in figure 4.9. The figure shows the flowchart of the process made up of connected Operation modules.

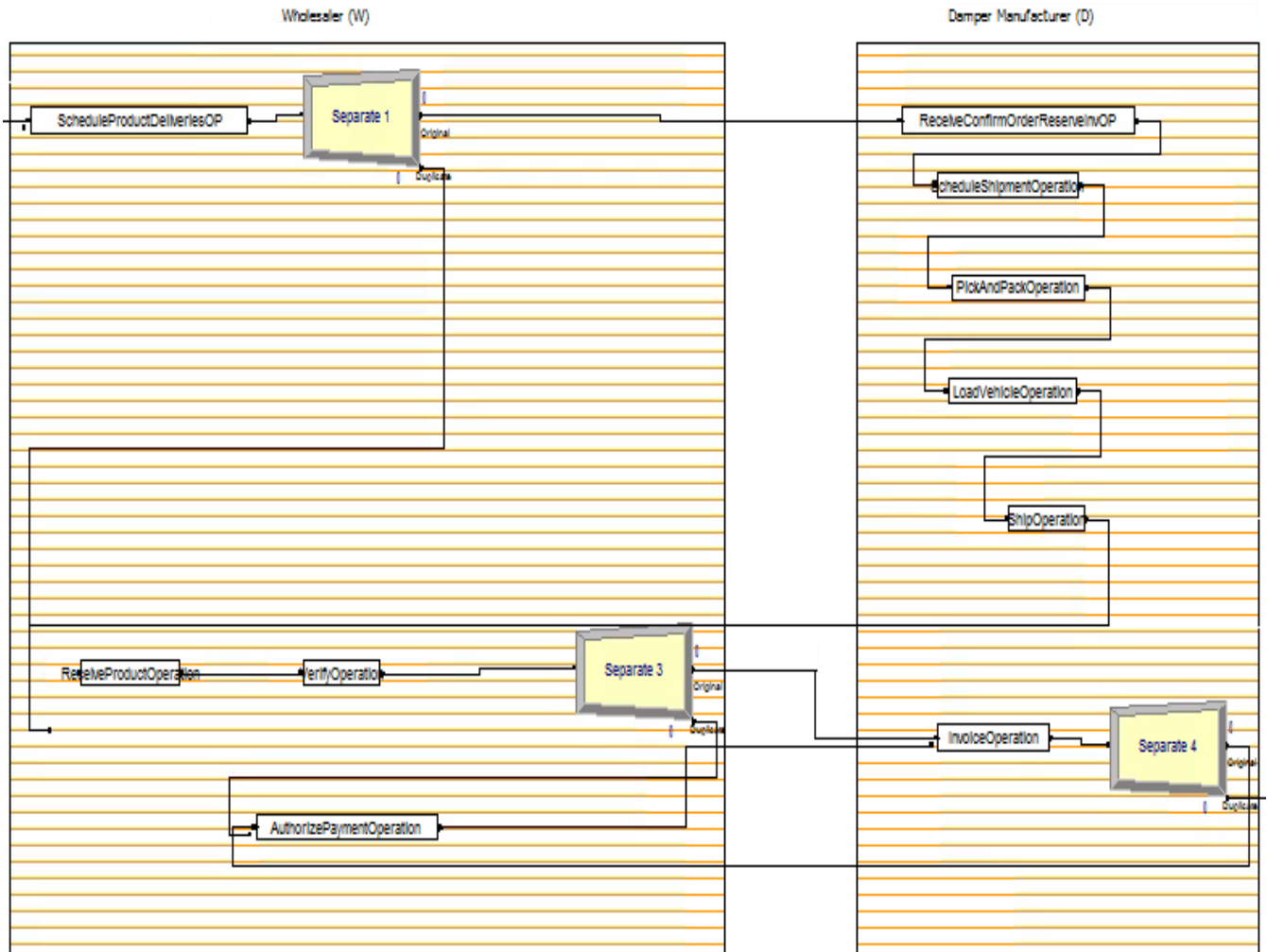


FIGURE 4.9: THE ARENA MODEL OF THE TRADING GOODS PROCESS

#### 4.1.3 TRANSLATION OF THE SC RISKS

We have proposed three modeling constructs in chapter 3 to be used by the SC practitioner to create a set of modeling constructs for the risks threatening the SC. In this section, we explain the translation of each modeling construct into a simulation pattern. We developed a library of SC risk modules within ARENA that are grouped into a Template. An overview of the template is shown in figure 4.10.

In next, we explain the translation of each Risk construct into a simulation module.

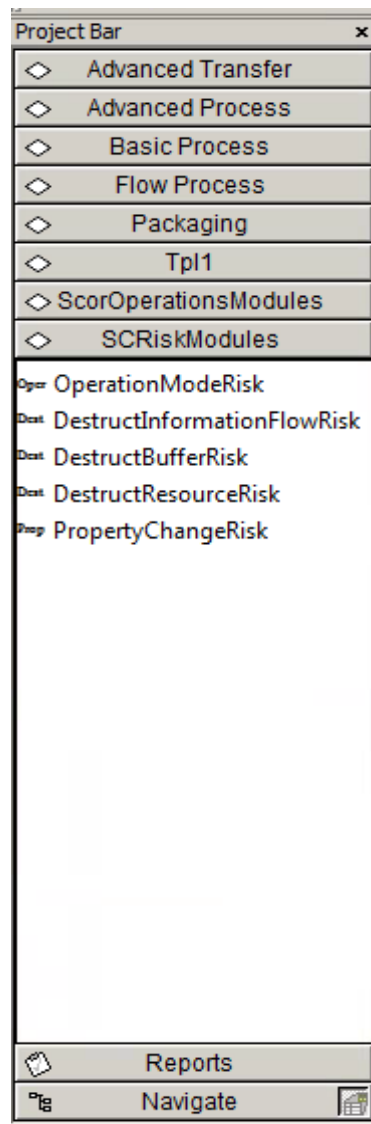


FIGURE 4.10: THE ARENA TEMPLATE FOR THE RISK MODULES

#### 4.1.3.1 TRANSLATION OF THE PROPERTYCHANGERISK

PropertyChangeRisk construct is used when a risk modifies one or several properties of a structural or behavioral element of the system (flow, resource, Actor...). The translation of this construct into an ARENA flowchart is shown in figure 4.11. The main role of this flowchart is to modify the attribute or variable values representing the property that is affected by a PropertyChangeRisk.

The flowchart of the simulation pattern is formed of two branches. The first branch modifies the values of the ARENA attributes or variables at the time of generation of an entity corresponding to the start time (or when the start time is reached for changing a flow entity attribute value). While the second branch reinitializes the value of the variable or the attribute in question (through subtracting the value of the previous change) at the time of generation of an entity corresponding to the end time (or when the end time is reached for the case of a flow entity attribute).

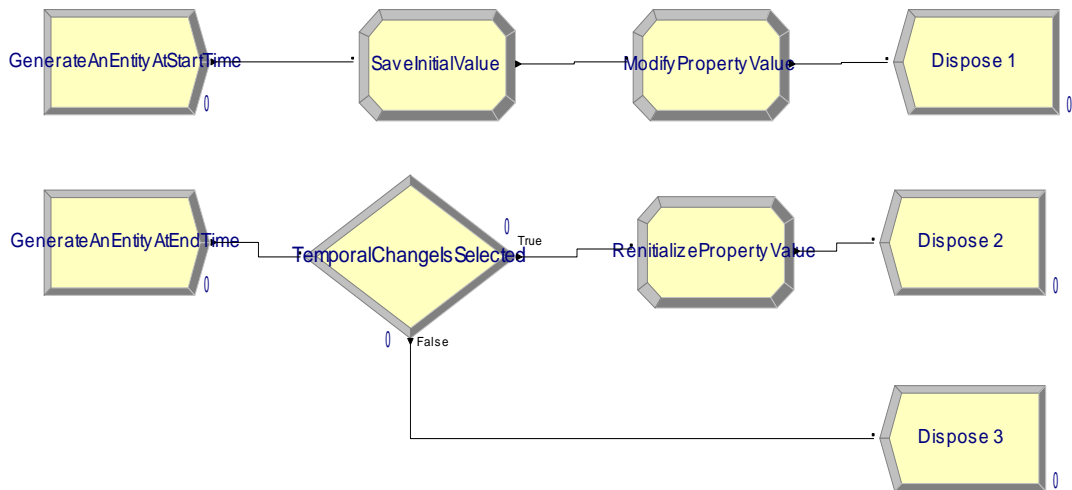


FIGURE 4.11: FLOWCHART OF THE PROPERTYCHANGERISK MODULE IN ARENA (FOR VARIABLES)

#### 4.1.3.2 TRANSLATION OF THE OPERATIONMODERISK

OperationModeRisk construct is used when a risk degrades a function of the system so that one or several operations are executed in a degraded mode for a period of time. We provide a translation of the OperationModeRisk construct in figure 4.12. The module has two parts:

The first part is responsible for setting the condition of activation to true or false and the second part is responsible of executing the degraded Mode. When activated, the degraded Mode of the related Operation will be executed until it is switched off (i.e. the systems resumes its normal operating conditions). We refer the reader to section 4.1.2.1 that explains the translation of the operation Modes. A degraded Mode of an Operation is translated in the same manner.

The flowchart module is shown in figure 4.12 describe the simulation module responsible for activating and deactivating the Operation mode Risk. The first branch activates the Operation mode when generating an entity at the *risk start time*. While the second branch deactivates the Operation mode when generating an entity at the risk end time.

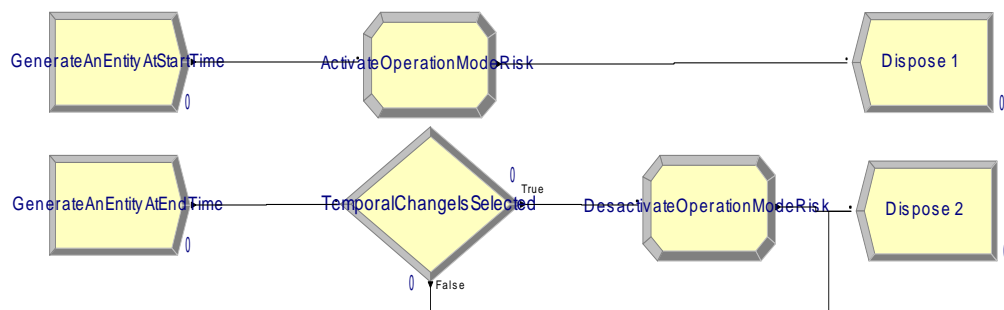


FIGURE 4.12: FLOWCHART OF THE OPERATIONMODERISK ARENA MODULE

In order to simulate this type of risks, the SC practitioner needs first to define the flow chart of the Risk operation mode sub-model, then he needs to instantiate a module that is responsible for activating and deactivating the risks and finally he needs to parameterize it.



#### 4.1.3.3 TRANSLATION OF THE OBJECTDESTRUCTIONRISK

ObjectDistructionRisk construct is used when a risk destroys an element of the system. In this case the impacted object has to be removed from the simulation model. In this section, we provide a translation of the ObjectDestructionRisk modeling construct. The translation depends on the kind of object impacted by this type of risk:

- Translation of the ObjectDestructionRisk modeling construct in the case of Resource objects

The proposed risk simulation pattern in the case of Resource objects is an ARENA flowchart module. The logic of the ARENA simulation module is shown in figure 4.13. The module is formed of a branch of ARENA elementary modules. In fact, the branch reserves infinitely the resource through the “Seize” module at the time of generation of an entity corresponding to the start time.

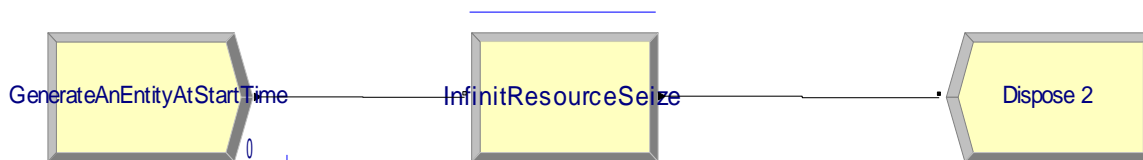


FIGURE 4.13: FLOWCHART OF THE ARENA OBJECTDESTRUCTIONRISK MODULE RELATIVE TO RESOURCE OBJECTS

Hence, to simulate this kind of risks, the SC practitioner needs to put an instance of this module in the model without connecting it to any other modules. It is parameterized through specifying the risk execution time.

- Translation of the ObjectDestructionRisk modeling construct for the case of Buffers or MoneyAccount objects

The proposed SC risk simulation pattern for the case of Buffers or MoneyAccount objects is an ARENA flowchart module. The logic of the ARENA simulation module is shown in figure 4.14. The flowchart deactivates the future use of the Buffer object or the MoneyAccount object by setting the value of the variable *Active(BufferId)/ Active(MoneyAccountId)* to “false” at the time of generation of an entity corresponding to the risk start time and through putting the values of the levels of the contained ( such as the *inventoryLevel* and the *moneyLevel...* ) to zero.

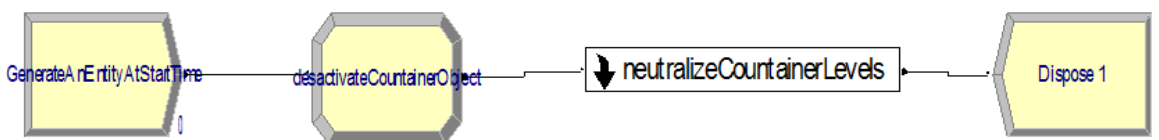


FIGURE 4.14: FLOWCHART OF THE ARENA OBJECTDESTRUCTIONRISK MODULE RELATIVE TO RESOURCE OBJECTS

So to simulate this kind of risks, the SC practitioner needs to put an instance of this module in the model without connecting it to any other models and parameterize it by defining the risk execution time.

- Translation of the ObjectDestructionRisk modeling construct for the case of Information Flow objects

The proposed ARENA simulation flowchart is shown in figure 4.15. The flowchart holds the information flow object infinitely through a “Hold” module when the simulation time equals the time of the risk execution.



FIGURE 4.15: FLOWCHART OF THE ARENA OBJECTDESTRUCTIONRISK MODULE RELATIVE TO INFORMATION FLOW OBJECTS

So to simulate this kind of risks, the SC practitioner needs to put an instance of this module at the arrow that transfers the Flow object in question.

- Translation of the ObjectDestructionRisk modeling construct for the case of Actor objects

The proposed Risk simulation module for the case of Actor objects is an ARENA flowchart module that is similar to the simulation module relative to Information flows objects. Hence, to simulate this kind of risks, the SC practitioner needs to put instances of the DestructInformationFlowObjectRisk module at the bow that connects the Actor with the other Actors to stop the exchange of flows.

#### 4.2 EXPERIMENTATION OF THE SIMULATION MODEL

The objective of the provided translation is to enable a quick and easy adaptation of the simulation model for experimentation. Hence, we assist the SC practitioner, by describing the adaptations to implement for conducting experiments. We define an experiment as a timed execution of a given configuration of the simulation model for a given set of parameters to evaluate the evolution of a set of metrics. We suggest the following adaptations for conducting experiments.

The SC practitioner needs to specify the duration of his/her experiment and the profile of the generated entities. Usually, the first generated entities are relative to the customers’ demands. Hence, he might adjust the following parameters (Demand arrival (product, quantity, inter-arrivals time)) to describe the demand profiles of the customers of his/her SC. The information about the demands might be generated internally or may be received from an external module.

Also, the SC practitioner may change the settings of Resources used by Operations by adjusting the following parameters (Resource (Number, capacity (per product), Time (by Product or by path or by route)) or he may change the setting of its stock by adjusting the following parameters (Stock (Initial inventory Level, replenishment level (if any), and target level (if any))).

Moreover, to evaluate the impacts relative to a given risk, the SC practitioner may implement a risk module and may set the following parameters (Risk (SC Risk Names, SC Risk types, Start times, End times, Magnitudes and the impacted elements (Facilities, Properties, Operations and Flows))).

---

Furthermore, the SC practitioner needs to collect data enabling the evaluation of the experimented phenomena. For example, he might implement the monitoring metrics (% of delivery on time for Actor, inventory level, resource utilization...).

Finally, after adapting the configuration of its SC model, the SC practitioner needs to specify the number of experiment's replications to be sure that the collected results are significant. To define the number of replications, we suggest the reader to use the existing statistical methods developed for this purpose (see for example (M. Law & Kelton 2000)).

When experimenting a new scenario, the SC practitioner needs to modify either the simulation and/or the conceptual model. It will be very helpful to automate a part of this modification process. In the next section, we provide guidelines for three possible modification scenarios: management policies, SC network or structure and the risk scenarios.

#### 4.2.1 DEFINE A SCENARIO CHARACTERIZED BY NEW POLICIES

For the experimentation of a scenario characterized by new policies, we adapt the model as follow:

If the newly adopted policy defines a different way by which an operation behave, the SC practitioner only needs to define a new Operation Mode by modifying the sub-model relative to the Operation in question. Hence, he needs to define a flowchart of the new Mode and the associated simulation variables. He needs to define the conditions of activation and deactivation. We refer the reader to section 4.1.2.1 for further information. For instance, the SC practitioner may integrate a sourcing policy by defining a new Operation Mode for the Operation SCHEDULEPRODUCTDELIVERIES.

If the newly adopted policy defines a different process network, the SC practitioner needs to modify his/her current process through changing the flowchart, instantiating new Operation modules, creating new Modes and defining the condition of activation and deactivation. For instance, the SC practitioner may integrate a mitigation policy through defining a new Process that permits storing the products in a secondary Buffer if the main Buffer is destroyed.

If the newly adopted policy defines new management parameters, the SC practitioner needs to modify the properties of the concerned objects. This is through modifying the values of the variables that translate these properties. For instance, the SC practitioner may increase the responsiveness of its SC through modifying the values of the properties that are responsible for managing the inventory.

#### 4.2.2 DEFINE A SCENARIO CHARACTERIZED BY A DIFFERENT SC STRUCTURE OR NETWORK

If the SC practitioner wants to simulate scenarios where the SC has a different structure or network, he needs to conduct specific modifications on his current model.

If the change concerns the SC infrastructure or the internal transportation network of the current SC, the SC practitioner needs to modify the related objects (e.g, Resource, Buffer, Path...) through modifying the values of the variables that represent their properties. He/she may also define new structure objects, for instance, a new Resource object that has better cycle time to increase the responsiveness of its SC.

---

If the change concerns the SC network (the Actor's relationships), the SC practitioner needs first to redefine the external transportation network through the creation or the modification of the Route objects by setting their variables. Then he needs to redefine the terms of the exchange relationships between partners by setting the variables that translate the properties of the Contract object. The Contract object specifies the terms of the trading and transportation relationships with partners. For instance, the SC practitioner may define a new Contract object to create a new relation with a redundant supplier in order to mitigate the disruption of supply.

Furthermore, the SC practitioner needs to modify its Process model through redefining new Operation instances, through adding new Process portions and their conditions of activation and through modifying their connections. For instance, the SC practitioner may define a new connection between the focal company and redundant supplier for mitigating a supply disruption.

#### 4.2.3 DEFINE A SCENARIO CHARACTERIZED BY RISKS

If the SC practitioner wants to simulate a risk scenario, he/she needs to adapt the simulation model as follows:

If the risks concern a sudden change in one of the SC parameters (e.g., a drop in the resource production capacity), the SC practitioner needs to instantiate the PropertyChangeRisk module. He has to parameterize it through specifying the property to modify, through setting the values of the start and the end times and through defining the amount of the change to simulate.

If the risk concerns a sudden change in the behavior of an Operation (e.g., a degraded functioning of the production operation), the SC practitioner needs to instantiate an OperationModeRisk, to parameterize it through setting the start and the end times and through specifying the mode to activate. Furthermore, if not programmed, the SC practitioner needs to create a mode sub-model within the Operation of interest.

If the risk concerns a sudden destruction of an object, the SC practitioner needs to instantiate the module relative to the object in question. Hence, if the object is a Resource or if it is a Buffer, the SC practitioner only needs to instantiate the Risk module without connecting it to the other elements of the simulation flowchart. If the object to be destroyed is of type Actor or of type Information flow, the SC practitioner needs to instantiate the Risk module and needs to connect it with each connector transferring the flows belonging to the concerned objects.

To give an example, we show how to adapt the simulation model for the simulation of a risk that destroys shipping orders between two SC Actors. The first step is to instantiate the Risk module and to parameterize it. The parameter setting is done through the ARENA dialog window shown in figure 4.16. Hence, we set values for the parameters: The *identifier* of the modeled instance and the *start time* that specifies when the risk occurs.

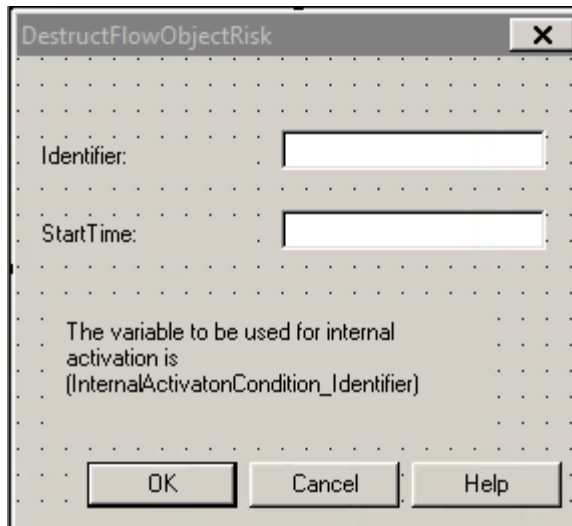


FIGURE 4.16: THE ARENA DIALOG WINDOW OF THE DESTROYOBJECTRISK MODULE RELATIVE TO FLOW OBJECTS

The second step is to connect the Risk module instance with the rest of the SC simulation model. Hence, the module is put between the instance of the SHIP Operation module and the instance of the RECEIVEPRODUCT Operation module for intercepting the shipment and to destroy the “shipping order”. The resulting process simulation model is shown in figure 4.17.

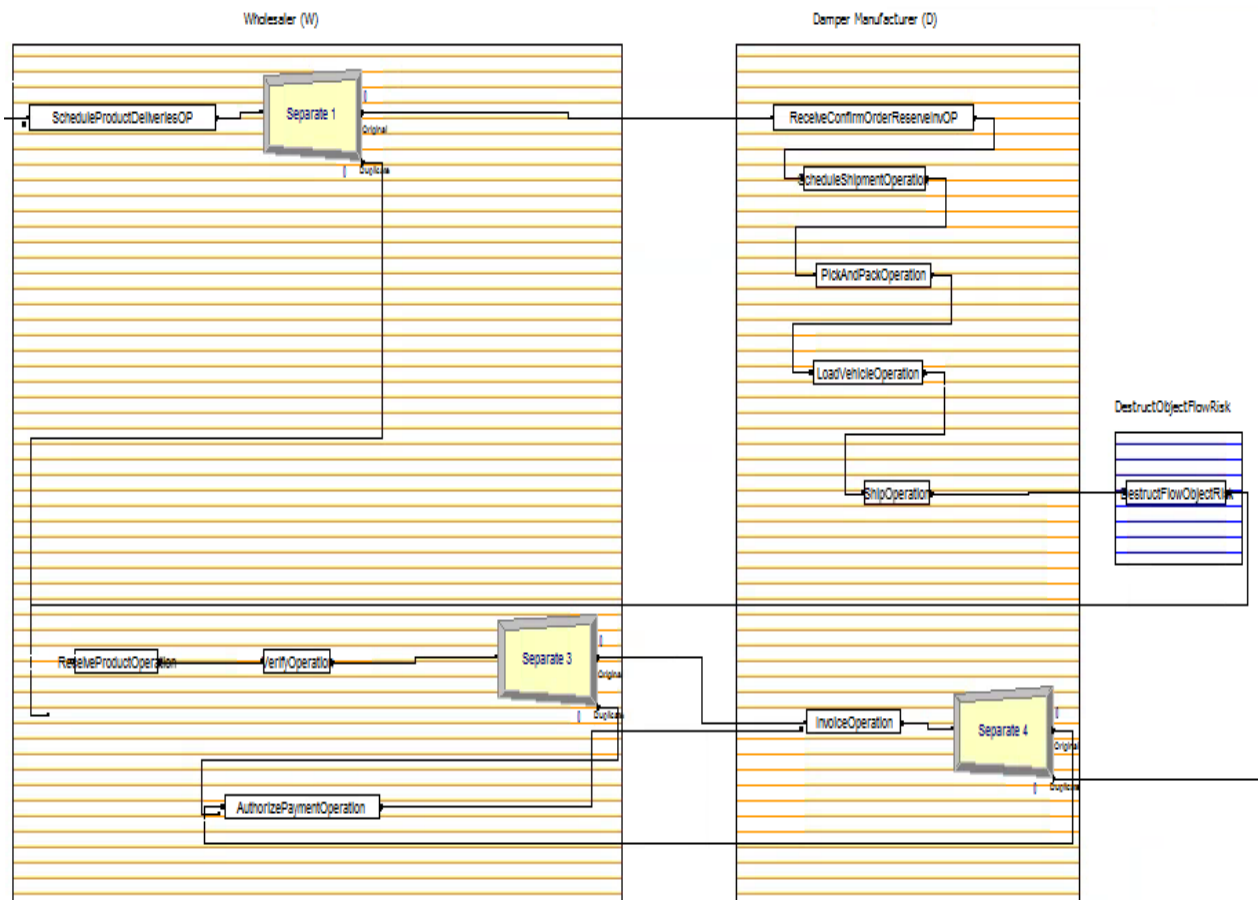


FIGURE 4.17: THE ARENA MODEL OF THE TRADING GOODS PROCESS INCLUDING AN INSTANTIATION OF THE DESTROYOBJECTRISK MODULE RELATIVE TO FLOW OBJECTS

---

## CONCLUSION

In this chapter, we provide a translation guideline for creating simulation models starting from the SC conceptual models.

The translation guideline is generic, different DES simulation languages can be used to develop the translation. The translation is illustrated using ARENA modules and sub-models.

The guideline enables an easy and more rapid construction of simulation models. Namely, it permits taking advantage from the syntax of the proposed meta-model and libraries (That defines a set of modeling constructs easily understandable and familiar to SC practitioners since they are based on the SCOR reference model) through providing how they can be directly translated into simulation modules and variables. Hence, we describe the translation for the structure element, the behavior elements (Operations and Flows) and for Risks. The developed modules are grouped into templates enabling a modular construction of simulation models.

Creating the simulation model is not sufficient for experimenting the scenarios of interest; know-how is required for modifying the simulation model to sketch those scenarios. Hence, based on the scenario to be experimented we specify the changes to implement. This enables an easier and more rapid experimentation of simulation models.

---

**CHAPTER** 5

**CASE STUDY: TRUCK-MUCH  
SUPPLY CHAIN**



---

## CHAPTER 5: CASE STUDY: TRUCK-MUCH SUPPLY CHAIN

### SUMMARY

This chapter illustrates through a case study the application of the methodology described in the previous chapters. To this end, we apply the following steps to a fictitious automotive SC operating in the manufacturing and the trading of trucks.

First we show the model of the structure of the SC, namely its static parts (facilities, resources, Buffers, routes...) and their relations. The usage of the structure meta-model is shown to procure a good description and to enable an easier creation of the conceptual model. Then we show the model of the behavior of the SC, namely the operations (PRODUCE, TRANSFER, SHIP...) and the processes using the behavior metamodel and library. The usage of the library of operations is shown to be capable of capturing the processes performed within the SC and to enable an easier construction of the process model. We then translate the conceptual model into a simulation model. The usage of the translation guideline and the library of simulation modules provide a reduction of the programming efforts and a quicker and easier creation of the simulation model. Finally we demonstrate how adapting the simulation model for integrating risk scenarios. The usage of the risk modules enable modeling and simulating risks with a simple drop and drag mechanism and to enable an advanced analysis of the risk effects.

### INTRODUCTION

In this chapter, we illustrate through a case study how the application of the framework supports risks analysis. We put into practice the tools proposed by the framework to support the proposed steps for analyzing the SC risk. Namely, we use the structure metamodel, the behavior metamodel, and the library of operations, the risk metamodel, the translation guideline, the library of simulation modules and the experimentation guideline.

The studied case is an academic case consisting of a fictive automotive SC operating in the field of trucks manufacturing. We start by creating the conceptual model for both the SC structure and behavior. Then, we translate the model into a simulation model following the translation guideline. Then, we verify the obtained simulation model and we conduct a set of experiments to analyze the risks of interest.

This case study is used to perform a first verification of the translation technique we propose from meta-model to simulation model. The verification phase of the produced simulation model enables testing the predefined SC operations we grouped in the ARENA library and the linkage made between them. The case study is thus provided to exemplify the proposed process and to have a first technical verification of the proposed modules and of the simulation model generation method.

## 5.1 DESCRIPTION OF THE CASE

In this section, we provide general information about the SC structure and its functioning.

The case of interest is the SC of a global automotive company "Truck-Much". Its plant (GTM) in Grenoble assembles the trucks. In this study we are interested in SCs of one

specific truck model called CGMV. This truck model is an assembly of two main sub-assemblies which are the body (Bd) and the lever (Lv). The sub-assemblies are supplied by two companies. The first one (Supplier1) supplies GTM with the Bd sub-assembly from its factory located in Tunisia. While the second one (Supplier2) provides GTM with the Lever sub-assembly from its factory located in Morocco.

The truck CGMV is sold to two vehicles distributors. They are respectively: The distribution center (DistC1) located in Paris and the distribution center (DistC2) located in Spain.

The studied SC is mapped in figure 5.1. Each of the SC members has its proper functioning. In next section, we detail the functioning of each one of them.

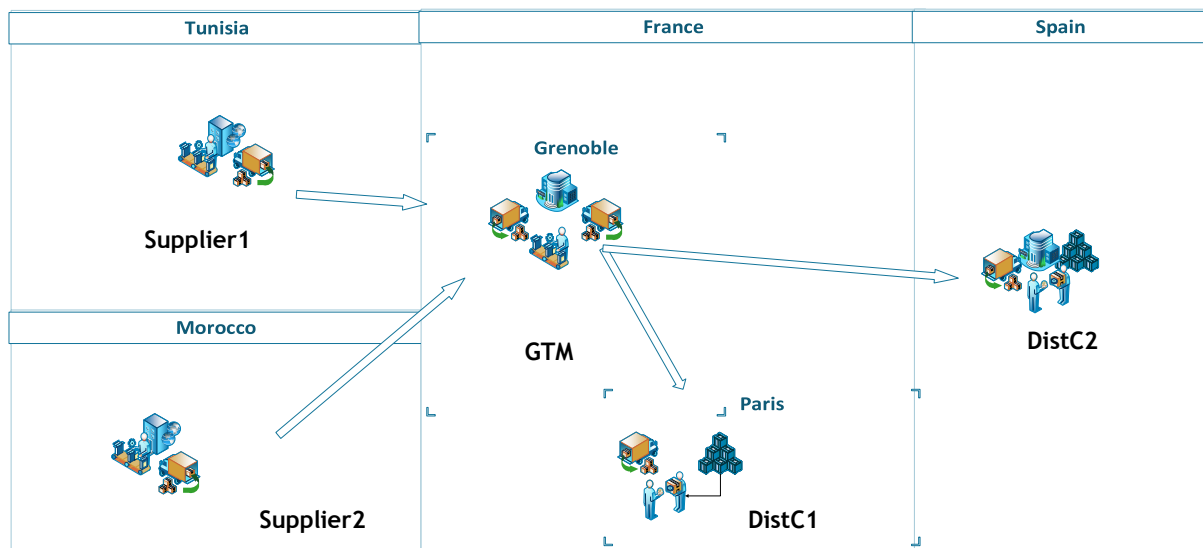


FIGURE 5.1: ILLUSTRATION OF THE TRUCK-MUCH SUPPLY CHAIN

#### ▪ Description of GTM

The truck model CGMV is manufactured in Grenoble factory. The plant follows a make to order policy. When a purchase order is received from one of the two distribution centers, a production order is launched. Then, the sub-assembly units required for production are transferred from the reception warehouse to the manufacturing station.

The inventory is managed by following the policy  $(s, S)$ . No order is edited until the inventory falls below a re-ordering point  $s$ , and the ordered quantity is defined in a way to restore the current level to the target level  $S$ . The purchase orders are sent either to supplier 1 or supplier 2 depending on the stock levels.

All the received sub-assemblies from suppliers are verified. The manufacturing process is an assembly line. The sourced body Bd is assembled with Lv to form the truck. The manufactured trucks are then loaded on transportation resources and shipped.

#### ▪ Description of DistC1 and DistC2

DistC1 and DistC2 serve their customers from their stock. They follow the source to stock policy. The inventory is managed following the same policy  $(s, S)$  as for the focal company GTM. Namely, after each trucks delivery the inventory trucks CGMV is revised and if the current level meets the ordering conditions, a purchase order is generated and sent to GTM. The ordered quantity will restore the on hand inventory to a target level  $S$ .

---

## ▪ Description of Supplier1 and Supplier2

After the reception of purchase orders from GTM, the two suppliers load their vehicles with the requested sub-assemblies and ship them to their customers. Both suppliers follow the make to order policy. The manufacturing process of subassemblies is not the focus of this study; we only retain an aggregated view of this process for modeling purposes.

## 5.2 MODELING THE STRUCTURE OF THE SC

In this section, we explain how we applied the framework for modeling the structure of the SC. Each part of the SC is modeled by conforming to the meta-model provided in figure 3.2 of chapter 3. We create for each part of the SC an instance of the corresponding construct. We set its properties using the collected information about the SC (Such as the identifier, the designation, the geographical localization ...).

Following the framework's steps, we do as follow:

- We start by modeling the Product view: i.e. the products traded within the SC and their components.
- Second, we model the Actor's network view: i.e. the agreements between Actors through Contracts.
- Third, we model the infrastructure view: i.e. the facilities of each SC Actor and their parts (the Resources, the Buffers ...)
- And finally, we model the transportation network view: i.e. the routes that link the facilities, the paths that link the buffers and the transportation resources and transfer resources used for moving goods.

To explain the creation of this model, we explain the creation of each portion of the model capturing a view of the SC structure.

### 5.2.1 PRODUCT VIEW

Thanks to the Product view, we define the bill of materials of the SC products. We create instances of the Product block for the truck CGMV and its sub-assemblies, Lv and Bd, respectively. Namely, we specify the *billOfMaterials* for the Truck 'CGMV' by setting the values 1,1 relative to the coefficient of its sub-assemblies Lv and Bd that are identified in the *listOfComponents* as 3 and 2, respectively and we specify the Truck dimensions. The resulting object diagram is shown in Figure 5.2.

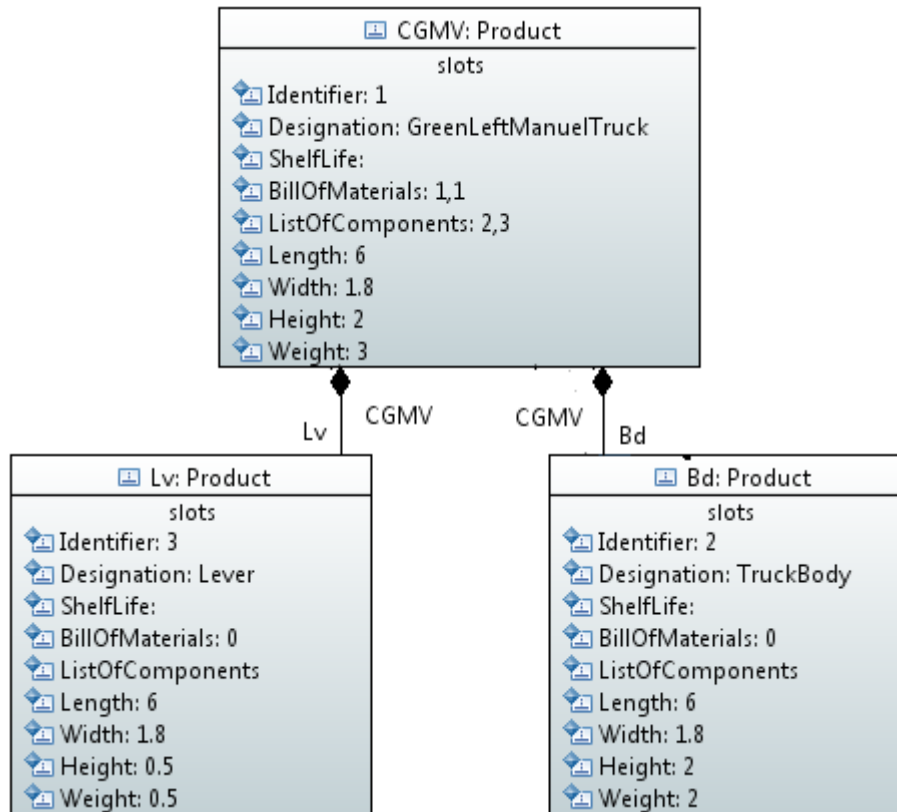


FIGURE 5.2: THE OBJECT DIAGRAM OF THE PRODUCTS CGMV, Bd AND Lv

### 5.2.2 ACTOR'S NETWORK VIEW

To model this view, we create instances of the Actor modeling construct for each SC member and instances of the Contract modeling construct for each of their relationships. The Contract construct enables the specification of the relationship between SC members. The global Actor's view is shown in the Annex A4.1.2 of this dissertation. In figure 5.3, we present only the part relative to TruckMuch and his relation with supplier1.

We specify for each property of the Contract block instance, the values relative to the clauses of the relationship. For instance, we specify the amount to be paid per day of delay, which is 0.02 % of the order payment and we specify the boundaries of the acceptable lead time (2 to 7 days).

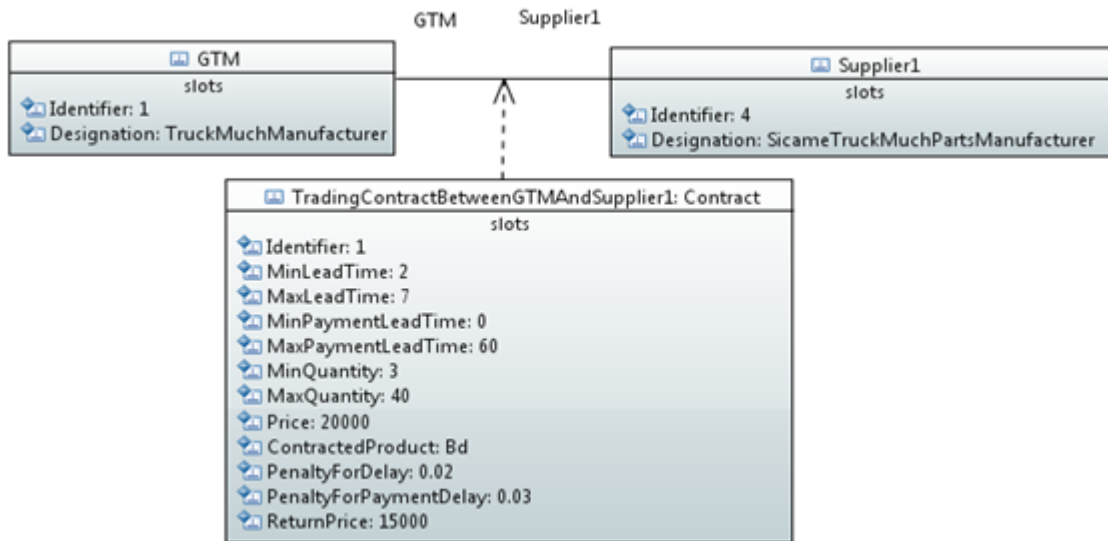


FIGURE 5.3: A PORTION OF THE OBJECT DIAGRAM OF THE ACTOR'S NETWORK VIEW OF THE STRUCTURE MODEL

### 5.2.3 INFRASTRUCTURE VIEW

To model this view, we specify for each SC member the owned infrastructure elements. Hence, for each Actor's physical grouping of warehouses or resources, we create an instance of the Facility construct and we create instances of the physical blocks which define it. The global model of this view is shown in the annex A4.1.3 of this dissertation.

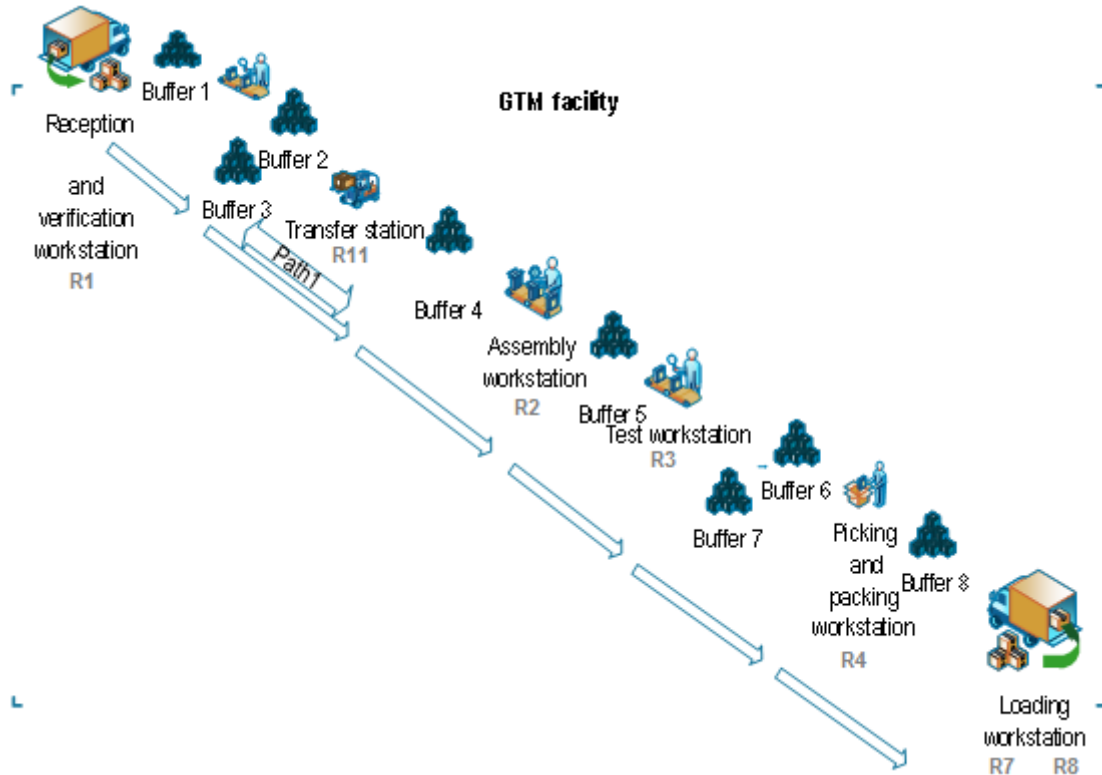


FIGURE 5.4: THE LAYOUT OF THE GTM FACILITY

In figure 5.4, we present an example of each infrastructure element (i.e. Facility, Resource, and Buffer) of TruckMuch. The GTM facility has a set of workstations used for performing operations and a set of temporary Buffers used for storing the products in process inventories.

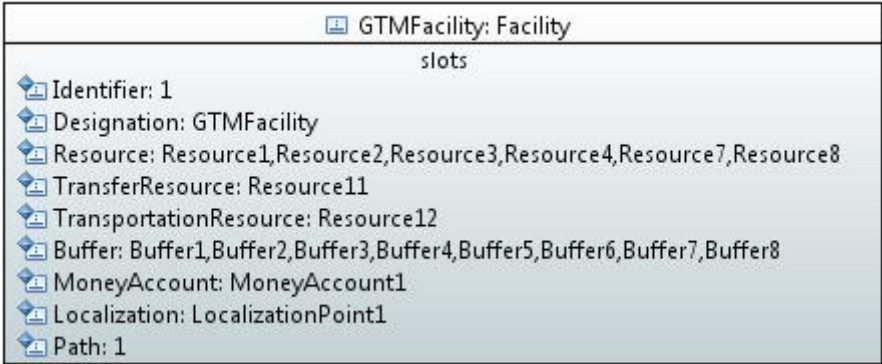


FIGURE 5.5: THE FACILITY INSTANCE OF THE GTM FACTORY

The Block Definition diagram for the GTM facility is presented in figure 5.5. We declare 5 internal resources to model the workstations (or equipment) and human resources that are present in the facility. Resources 1 (R1) to 4 (R4) represent the reception-verification workstation, the assembly workstation, the testing workstation and the picking and packing workstation, respectively. Resource 7 (R7) and Resource (R8) models the resources used for loading the vehicles which trucks and the resource 11 (R11) models the transfer station. To detail an example, the reception - verification workstation (Resource1) is shown in figure 5.6. As shown in figure 5.6, the Resource instance designates the products that are handled by the modeled resource, specifies the number of resources of the same type, the reception and the verification capacity and the cycle time for each handled Product and defines the values for a set of properties related to the current performances of the resources.

We also declare 8 Buffers, for modeling the GTM warehouses and temporary storages where the work-in-progress are held: the received subassemblies storage (Buffer 1), a Buffer for the verified subassemblies qualified as good (Buffer no.2), a Buffer for the verified subassemblies qualified as defective (Buffer 3), a Buffer for the issued subassemblies (Buffer no.4 ), a Buffer for the assembled trucks (Buffer no.5), a Buffer for the verified trucks qualified as good (Buffer no.6), a Buffer of the verified trucks qualified as defective (Buffer no.7), and a Buffer for the picked and packed trucks (Buffer no.8).

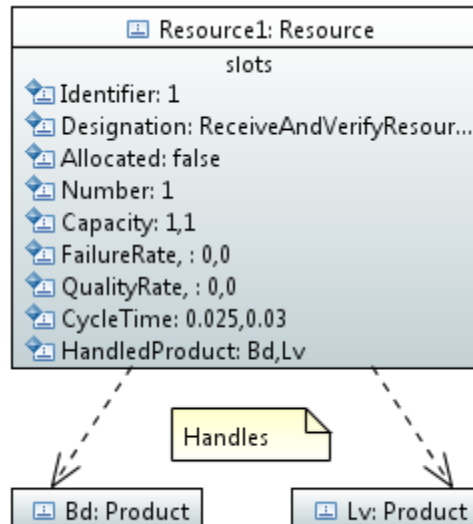


FIGURE 5.6: EXAMPLE OF A RESOURCE INSTANCE BELONGING TO THE GTM FACILITY

For each Buffer declared in the facility, we create an instance of the Buffer construct. The received sub-assemblies' Buffer is shown in figure 5.7. As shown in the figure, the Buffer instance names the stored sub-assemblies to which it is associated, and specifies the storage capacity and the current inventory levels for each sub-assembly.

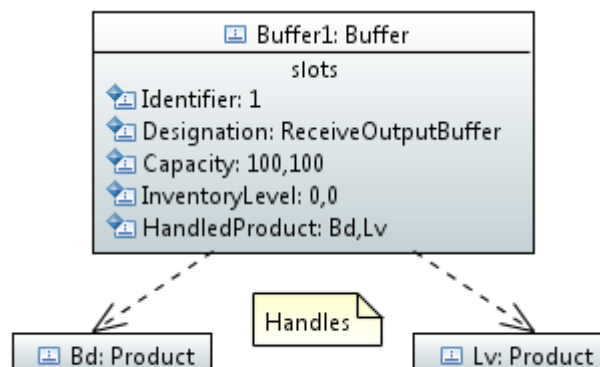


FIGURE 5.7: EXAMPLE OF A BUFFER INSTANCE BELONGING TO THE GTM FACILITY

#### 5.2.4 TRANSPORTATION NETWORK VIEW

To model this view, we create instances of the modeling constructs used for representing the internal and external transportation for each of the SC member. The global model of this view is shown in the Annex A4.1.4 of this dissertation.

Below, we present an example of each element of the transportation network view (i.e. route, transportationResource, transferResource, and path) relative to GTM. GTM uses a set of vehicles for transporting the manufactured trucks; the shipment is done through a set of routes linking between GTM and the other facilities. Hence, we declare 4 routes instances for modeling the routes linking GTM and DistC1, GTM and DistC2, Supplier1 and GTM and finally, Supplier2 and GTM, respectively. The instance for the route linking GTM and DistC1 is shown in figure 5.8.



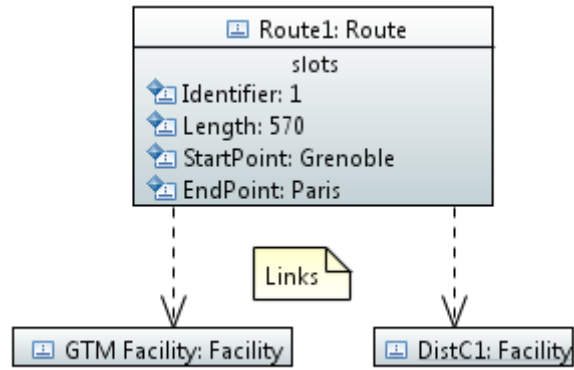


FIGURE 5.8: EXAMPLE OF A ROUTE INSTANCE LINKING GTM AND DISTC1

We declare one TransportationResource instance for modeling the vehicle used for transporting the manufactured trucks for DistC1 and DistC2. The instance for the shipping vehicle is shown in figure 5.9.

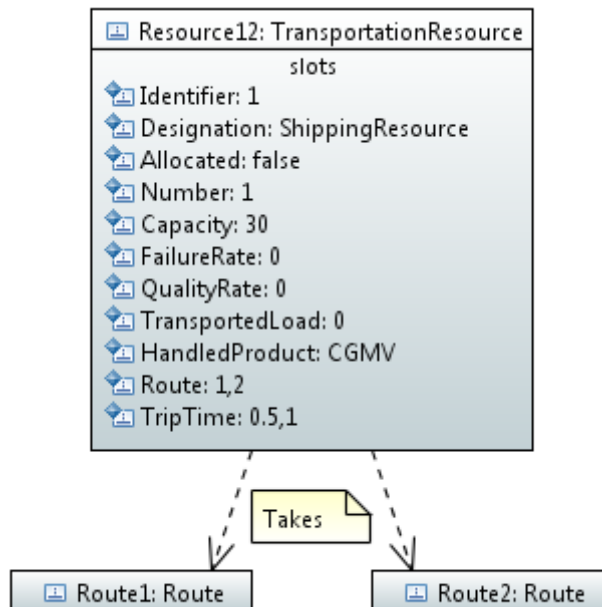


FIGURE 5.9: EXAMPLE OF AN INSTANCE OF THE TRANSPORTATIONRESOURCE USED FOR SHIPPING TO DISTC1 AND DISTC2 FACILITIES

Finally, we create an instance of the TransferResource construct and an instance of the Path construct for modeling respectively the resource used to transfer products between the Buffer 2 and the Buffer 4 and the line that links them. The TransferResource instance and the Path instance are shown in figure 5.10.

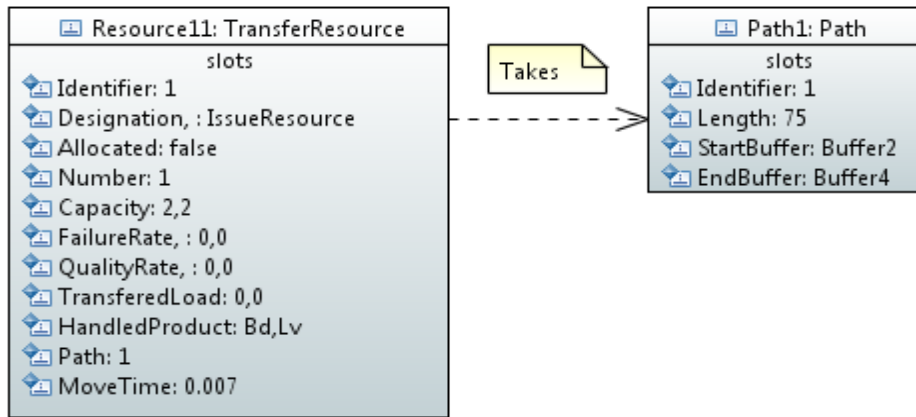


FIGURE 5.10: EXAMPLE OF A TRANSFERRESOURCE'S INSTANCE LINKED WITH THE USED PATH

We note that GTM also uses a transfer station for issuing sub-assemblies by taking paths linking the facility Buffers as shown in figure 5.4. The way we declare these paths between Buffers inside a facility is very similar to Route declaration between facilities.

## 5.3 MODELING THE BEHAVIOR OF THE SC

To model the behavior of the SC, we start by modeling the functions of each SC member and then the SC processes. We use the meta-model described in Chapter 3 (figure 3.8).

### 5.3.1 MODELING THE SC ACTIVITIES

To create the model of the SC activities, we generate the instances of the Operation modeling constructs. To model an activity of the SC, we have the choice between completely defining an Operation construct instance or to use the predefined operations library.

To quickly instantiate the required Operation constructs, we pre-filter them through the Role constructs corresponding to the capabilities of each SC member. For example, for modeling the capability of trucks assembling we declare an instance of the Role Maker construct. For each instantiated Role, we specify the name of the instances of the Operation constructs corresponding to the related SC functions. The overall model grouping the SC Role instances and the related Operation instances are shown in the Annex A4.2.1. Here, we only describe the functions related to the Actor "Truck-Much".

We declare four Role instances for Truck Much: The Maker Role, the Vendor Role, the Buyer Role and finally the Deliverer Role. Within each Role instance (e.g. as the Role Make), we name the declared Operation instances (e.g. we specify the name AssembleTrucks for the Produce Operation instance...). All Role instances for TruckMuch are shown in figure 5.11.

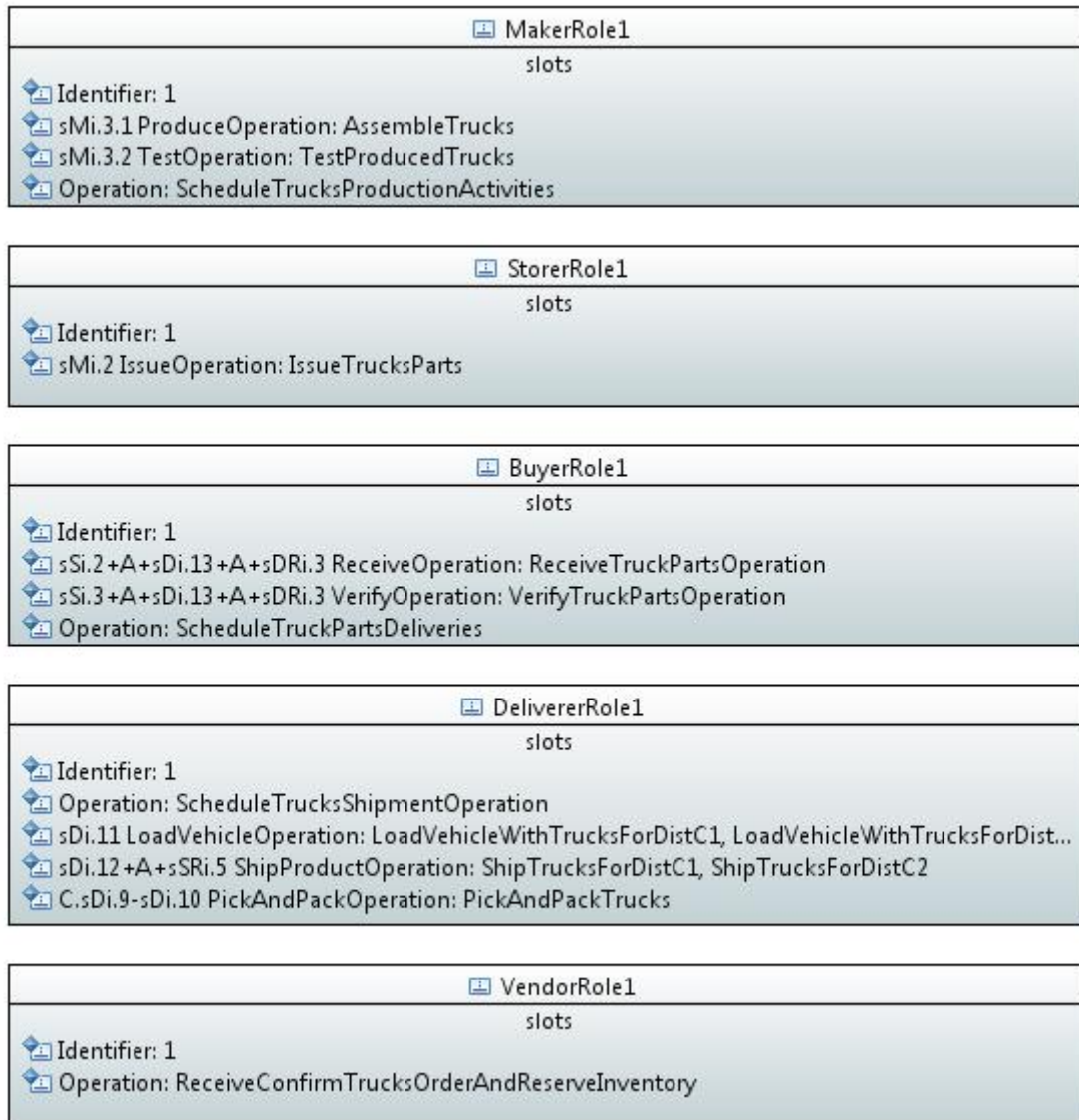


FIGURE 5.11: THE INSTANCES OF THE ROLE CONSTRUCTS RELATIVE TO TRUCKMUCH

The rest of the instances of the Role constructs for the remaining SC members are shown in the Annex A4 of this dissertation.

After naming the declared Operation's instances, we customize by defining their properties' values. Some values are references to the SC structure objects. As an example, the instance of the PRODUCE Operation construct named "AssembleTrucks" is shown in figure 5.12.

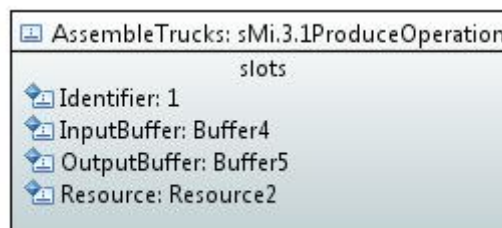


FIGURE 5.12: INSTANCE OF THE SMI.3 PRODUCE OPERATION RELATIVE TO TRUCKMUCH

### 5.3.2 MODELING THE SC PROCESSES

After modeling the activities performed within the studied SC, the next step is to create a model for the processes that organize their execution. To create the model of a given process, we first collect the information about how the SC activities interact and what they exchange as data flows.

To create the model for the processes, we create for each activity's connection an instance of the Operation Interaction modeling construct. Indeed, the Operation Interaction construct enables the specification of the connected operation instances and the exchanged data flows. Then the collected information are mapped within a SysML activity diagram, which enables the specification of series of connected Operation instances as a set of activity nodes connected with connectors. Each connector holds a representation of each type of the exchanged data flows between Operation instances. The activity node can also represent a sub-process.

Hence, we create activity diagrams for the studied SC process. We start by creating an aggregated view of the process where some activity nodes model the sub-process performed within an Actor. Then by following a top-down approach, we create an activity diagram for each sub process. The aggregated process is shown in figure 5.13. It describes data flow exchanged between the sub-processes of each facility. Namely, the GTM sub-process receives PurchaseOrder object flows from both the sub-process of DistC1 and the sub-process of DistC2 and returns PurchaseOrder object flows relative to shipping back to them.

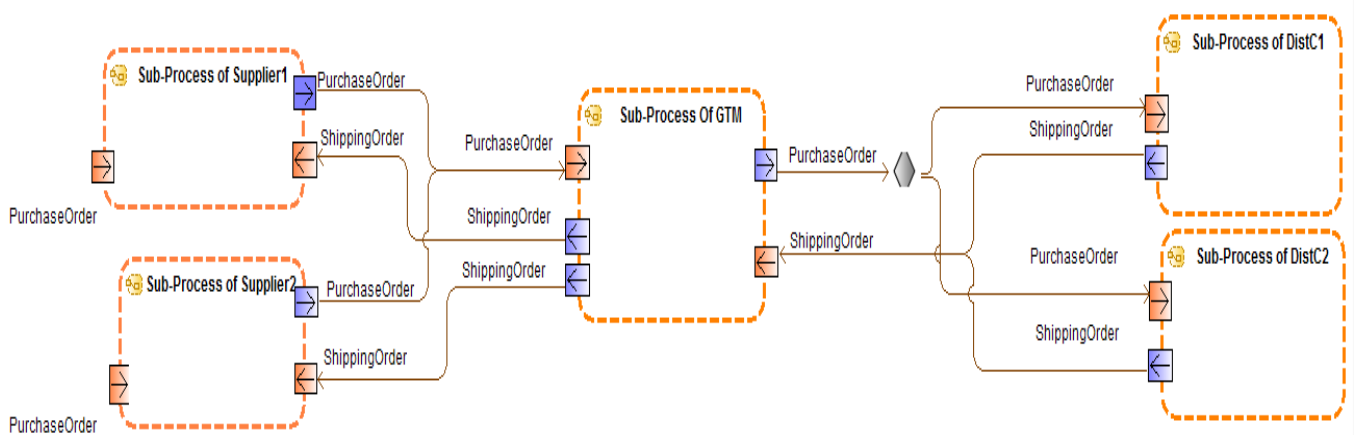


FIGURE 5.13: THE ACTIVITY DIAGRAM MODELING THE AGGREGATED SC PROCESS

The sub-process of each facility is presented as an activity diagram that details the organization of the execution of the Operation instances. The model of the facility sub-processes is shown in the Annex A4.2.1 of this dissertation.

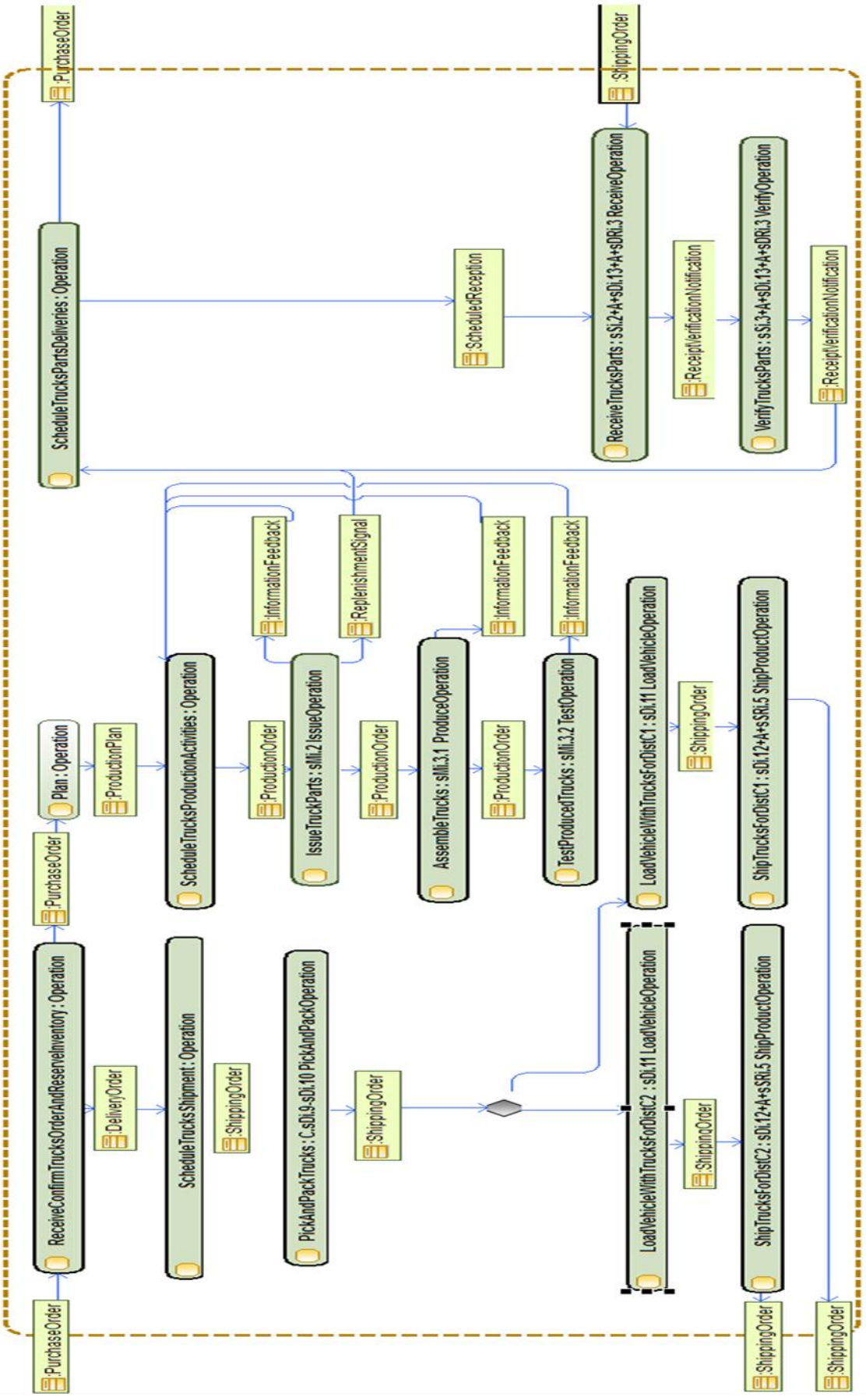


FIGURE 5.14: THE ACTIVITY DIAGRAM MODELING THE GTM SUB-PROCESS

---

We present as an example the model of the GTM sub-process, where each Operation instance is modeled as an activity node. It is shown in figure 5.14. The sub-process organization is as follows:

A PurchaseOrder is received by the Operation instance named receiveConfirmTrucksOrderAndReserveInventoryDate (Equivalent to the SCOR process elements ( "sD1.1/sD1.2 Process Inquiry and Quote", "sD1.2/sD2.2 Receive, Enter, and Validate Order" and "sD1.3/sD2.3 Reserve Inventory and Determine Delivery Date"). This one validates the PurchaseOrder and sends it to the Operation instance named Plan (equivalent to the "sP Plan" sub-processes of SCOR) and also edits a DeliveryOrder that specifies the delivery date. The DeliveryOrder is then received by the Operation instance named ScheduleShipment (Equivalent to the SCOR sub-processes "sD1.4/ sD2.4 Consolidate Orders", "sD1.5/ sD2.5 Build Loads", "sD1.6/ sD2.6 Route Shipments", "sD1.7/ sD2.7 Select Carriers and Rate Shipments"). This one edits a ShippingOrder that specifies the shipping details such as the vehicles to use and the routes to take. The ShippingOrder is received by the instance of the PICKANDPACK Operation named PickAndPackTrucks. This one waits until the manufactured Trucks become available in its input Buffer.

In the meanwhile, the Operation instance named Plan edits a ProductionPlan that is used by the Operation instance named ScheduleTrucksProductionActivities (Equivalent to the SCOR sub-process "sM1.1/sM2.1 ScheduleProductionActivities") for generating ProductionOrders. The ProductionOrders are sent to the instance IssueTruckPartsOperation of the ISSUEMATERIAL Operation. This one transfers the required components to the input Buffer of the instance AssembleTrucksOperation of the PRODUCE Operation.

After producing the required trucks, a ProductionOrder is sent to the instance TestProducedTrucks of the TEST Operation which verifies the quality of the manufactured trucks mentioned in the ProductionOrder and separates the defective ones from the correct ones. At this stage, the inventory becomes available in the input Buffer of the Operation PickAndPackTrucks. Hence, this one picks and packs the trucks and sends a ShippingOrder for the LOADVEHICLE Operation's instances (LoadVehicleWithTrucksForDistC1 and the LoadVehicleWithTrucksForDistC2). After loading vehicles with trucks, a ShippingOrder is sent to the SHIPPRODUCT Operation instances (ShipTrucksForDistC1 and ShipTrucksForDistC2) responsible for delivering Trucks to GTM customers.

## 5.4 CREATING AND VERIFYING THE SIMULATION MODEL

In this section, we present how the conceptual model is translated into a simulation model by following the framework translation guideline and how the created simulation model is verified and used to experiment a set of SC risk scenarios.

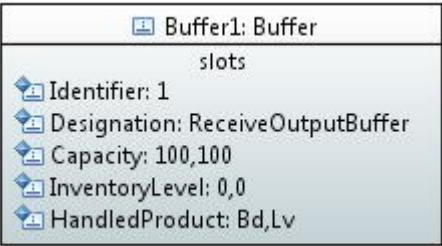
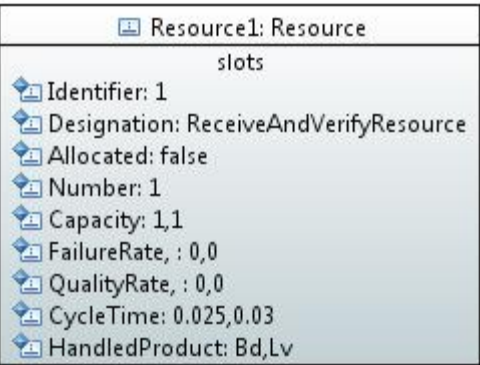
The simulation model is created following the framework's translation guideline described in chapter 4. First, we translate the SC structure objects into a set of simulation variables that are global and that can be used by the instantiated patterns. We illustrate the translation of the structural objects by giving examples on Buffer1 and Resource1.

To generate the simulation flowchart for Buffer1 (used for storing the received sub-assemblies) we start by creating an array variable for each property of the Buffer except for



the *Identifier* and the *HandledProduct* properties. The first column of the array is used to identify the Buffer instance and the second column is used to identify the product. Hence, for translating Buffer1 we only need to specify the values of a line of the Buffer array variable. The specified values are shown in Table 5.1. Since Buffer1 is not concerned with an inventory management policy, we do not assign any value for the properties *ReplenishmentLevel*, *SecurityLevel*, *TargetLevel*, *FixedOrderingQuantity*, and *CheckingPeriod*.

TABLE 5.1: EXAMPLE OF THE TRANSLATION OF SOME OF THE INSTANCES OF TRUCK MUCH STRUCTURE

Model objects	ARENA model variables
	<p>The capacity variable which is an integer array of 2 dimensions is initialized with the following values:</p> <ul style="list-style-type: none"> <li>Capacity: <ul style="list-style-type: none"> <li>(1,2)= 100</li> <li>(1,3)=100</li> </ul> </li> </ul> <p>The Inventory level variable which is an integer array of 2 dimensions is initialized with the following values:</p> <ul style="list-style-type: none"> <li>InventoryLevel: <ul style="list-style-type: none"> <li>(1,2)= 100</li> <li>(1,3)=100</li> </ul> </li> </ul> <p>Where 1 refers to the <i>identifier</i> of Buffer1, 2 refers to the <i>identifier</i> of the Product Lv and 3 refers to the <i>identifier</i> of the Product Bd.</p>
	<p>The Capacity variable which is an integer array of 2 dimensions is initialized with the following values:</p> <ul style="list-style-type: none"> <li>Capacity: <ul style="list-style-type: none"> <li>(1,2)= 1,</li> <li>(1,3)=1,</li> </ul> </li> </ul> <p>The CycleTime variable which is a float array of 2 dimensions is initialized with the following values:</p> <ul style="list-style-type: none"> <li>CycleTime: <ul style="list-style-type: none"> <li>(1,2)= 0.025,</li> <li>(1,3)=0.03</li> </ul> </li> </ul> <p>Where 1 refers to the <i>identifier</i> of Buffer1, 2 refers to the <i>identifier</i> of the Product Lv and 3 refer to the <i>identifier</i> of the Product Bd.</p> <p>The <i>number</i> property of the instance of the declared Resource element is initialized with the value 1.</p>

To translate Resource1, used for the reception and the verification of sub-assemblies, we start by creating an array variable for each property of the Resource constructs except for the identifier and the *HandledProduct* properties. The first column of this array is used to identify the resource instance in question and the second column is used to identify the product. Furthermore, as suggested by the guideline, we create an instance of an ARENA Resource element for each declared resource of the conceptual model. Hence, for translating Resource1, we only need to create an instance of the ARENA Resource element and we specify the



values for a line of the Resource array variable. The specified values are shown in Table 5.1. Since the failure and the quality issues are not modeled for this resource and since the cost issue is not considered here, we do not assign any value for *FailureRate*, *QualityRate*, and *CostRate*.

After translating the structure object, the next step is to translate the Operation instances and the modeled process.

We use the simulation patterns developed in ARENA for translating the Operation instances and we create sub-models when the pattern is not provided. The simulation modules are parameterized by setting their values and by referring the used resource elements. For example, we translate the instance of the PRODUCE Operation through using the developed ARENA Produce module. The parameterized module is shown in figure 5.15.

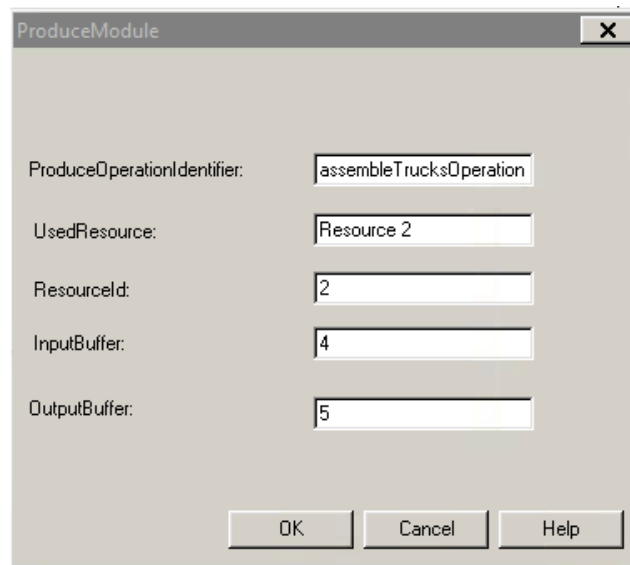


FIGURE 5.15: EXAMPLE OF TRANSLATION OF THE SMI.3 PRODUCE OPERATION INSTANCE INTO AN ARENA MODULE

Furthermore, we use a set of ARENA modules for creating the model of the operations that do not belong to the library and for collecting the information about the evolution of the SC variables. For example, a <Create> module is used to generate the entities that represent the final client purchase orders in Paris and in Spain and a <Write> module is used for saving results in an output file. We note that this is only for experimentation purposes. In case of an industrial application, the PurchaseOrders shall be read from a data base collecting this information.

The modeled process is translated through connecting the instantiated Operation modules and through using a set of ARNEA modules to complete it. For example, figure 5.16 shows how the instance AssembleTrucksOperation of the PRODUCE Operation module and the instance TestProducedTrucksOperation of the TEST Operation module are connected together for translating a portion of the modeled process.

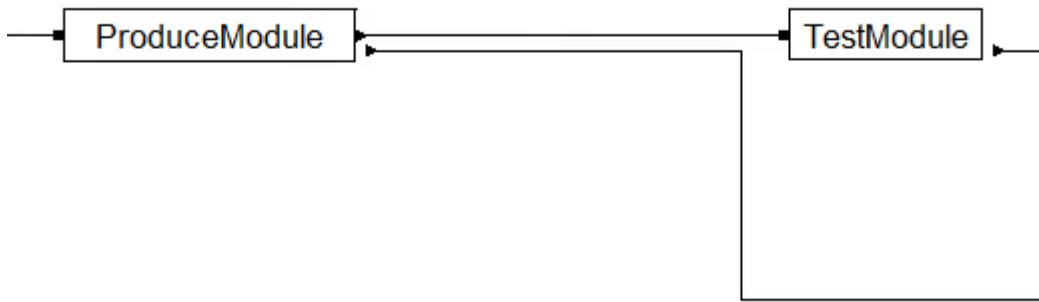


FIGURE 5.16: AN EXAMPLE OF ARENA MODEL WHERE OPERATION PATTERNS ARE CONNECTED

The resulting ARENA model for the whole modeled process is shown in figure 5.17. In this figure, we use a sub-model for each sub-process performed within each SC Actor.

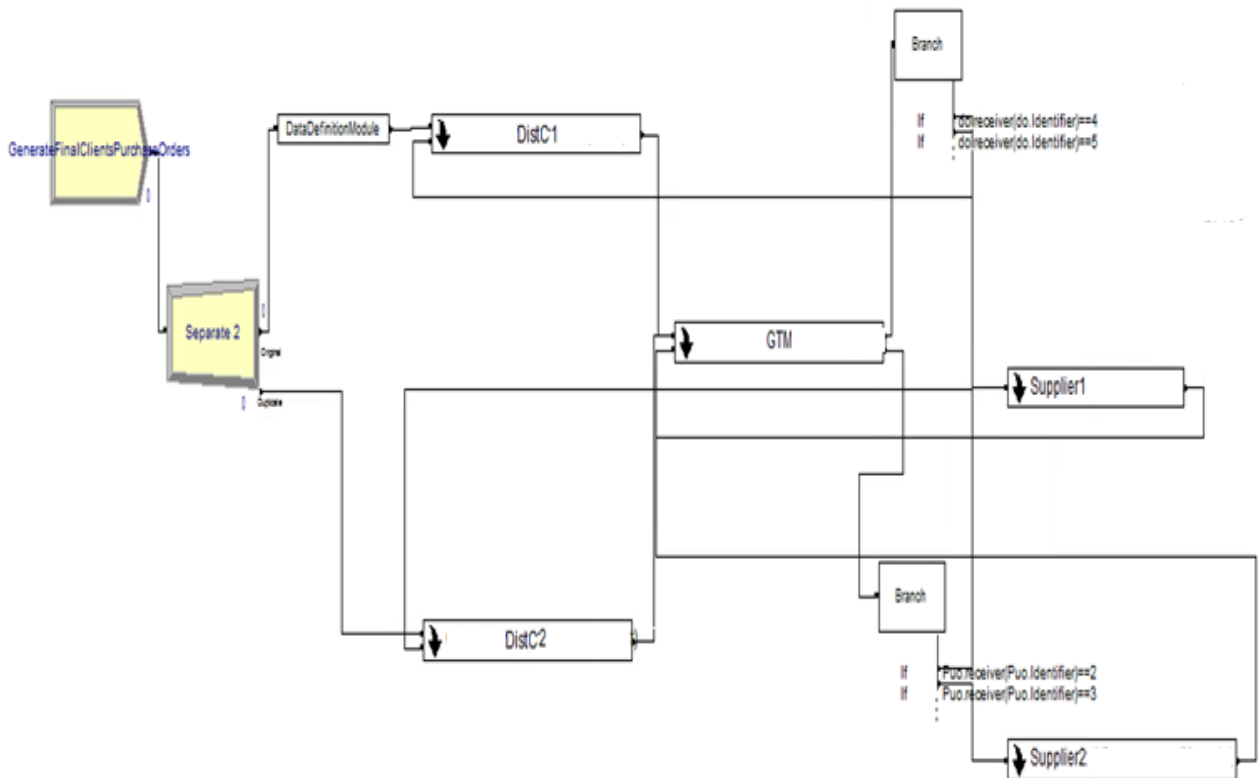


FIGURE 5.17: THE ARENA MODEL OF THE STUDIED SC PROCESS

Once created the simulation model needs to be verified for assuring that the simulation model is well constructed. The verification is conducted on a simple ‘test case’ that covers the interactions presented in the model. The Test case is built through feeding the simulation model with deterministic data. In this case, the verification consists in checking that the experimental outputs of each implemented module meet the analytical results.

To calculate the output of a given module, we use its mathematical formulas and the initial data. Hence, three steps are repeated in an iterative way to calculate outputs:

1. Gather received information flow (Such as PurchaseOrder, ProductionOrder ...) and save reception date.

2. Gather the data about current inventory levels and resource attributes required for the formulas.
  3. Injects required data into the aggregated mathematical formulas and save the results.
- In next, we detail the verification starting by presenting the Test case experiment and then by presenting the verification data set. The verification experiment

First, we adapt the simulation with the verification data. Hence, we set the values of the model variables with the ones shown in tables 5.2 to 5.5.

TABLE 5.2: RESOURCES SETTINGS

Names	Resource 1	Resource 2	Resource 3	Resource 4	Resource 5	Resource 6	Resource 7	Resource 8	Resource 9	Resource 10
<b>Identifiers</b>	1	2	3	4	5	6	7	8	9	10
<b>Designations</b>	GTM Receive And verify Resource	GTM Production Resource	GTM Test resource	GTM Pick And Pack resource	DistC1 Receive And Verify Resource	DistC2 Receive And Verify Resource	GTM primary Load Vehicle	GTM secondary Load Vehicle	Supplier1 Load Vehicle	Supplier2 Load Vehicle
<b>Numbers</b>	1	1	1	1	1	1	1	1	1	1
<b>Per products</b>	Cycle Times (Days) per product									
<b>CGMV</b>	0	0.3	0.08	0.03	0.07	0.07	0.002	0.002	0	0
<b>Bd</b>	0.05	0	0	0	0	0	0	0	0.003	0
<b>Lv</b>	0.05	0	0	0	0	0	0	0	0	0.003
<b>Per products</b>	Capacities per product									
<b>CGMV</b>	0	1	1	1	1	1	1	1	0	0
<b>Bd</b>	1	0	0	0	0	0	0	0	1	0
<b>Lv</b>	1	0	0	0	0	0	0	0	0	1

TABLE 5.3: TRANSPORTATION RESOURCES SETTINGS

Names	Resource12	Resource13	Resource14	Resource15
<b>Identifiers</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>Designations</b>	GTM vehicle for shipping to DistC1	GTM vehicle for shipping to DistC2	Supplier1 vehicle for shipping to GTM	Supplier2 vehicle for shipping to GTM
<b>Numbers</b>	1	1	1	1
<b>Per products</b>	Capacities per product			
<b>CGMV</b>	30	30	0	0
<b>Bd</b>	0	0	30	0
<b>Lv</b>	0	0	0	30
<b>Per routes</b>	Trip Times (Days) per route			
<b>route 1</b>	0.5	0	0	0
<b>route 2</b>	0	1	0	0
<b>route 3</b>	0	0	1	0
<b>route 4</b>	0	0	0	1

TABLE 5.4: TRANSFER RESOURCES SETTINGS

<b>Names</b>	Resource11
<b>Identifiers</b>	1
<b>Designations</b>	GTM transfer machine
<b>Number</b>	1
<b>Capacities per Product</b>	
CGMV	2
Bd	2
Lv	0
<b>Move Times (Days) per path</b>	
Path 1	0.007

TABLE 5.5: BUFFERS SETTINGS

<b>Identifiers</b>	12	6	2	9	10	The rest
<b>Designations</b>	Buffer of purchased trucks of DistC1	Buffer of manufactured trucks of GTM	Buffer of verified subassemblies of GTM	Buffer of produced Bd subassembly of Supplier1	Buffer of produced Bd subassembly of Supplier2	...
<b>Inventory Level IL(BufferID, ProductID)</b>	IL(12,1)=12 for CGMV	IL(6,1)=0 for CGMV	IL(2,2)=20 for Bd IL(2,3)=20 for Lv	IL(9,2)=900 for Bd	IL(10,3)=900 for Lv	0 for all

Second, we execute the simulation and we save the results after each execution of an instantiated module. Namely, the experiment is triggered with the first event that consists of the reception of a final client order for the Product CGMV by DistC1 at time zero. This event triggers a series of transition that modifies the SC states. To give an example of the collected results; we illustrate in figure 5.18 the evolution of the most important inventory levels of the GTM factory: for the sub-assemblies Bd, and Lv and the manufactured Product CGMV, respectively. The steps when inventory levels are saved are indicated by the execution end time of the GTM simulation modules instances.

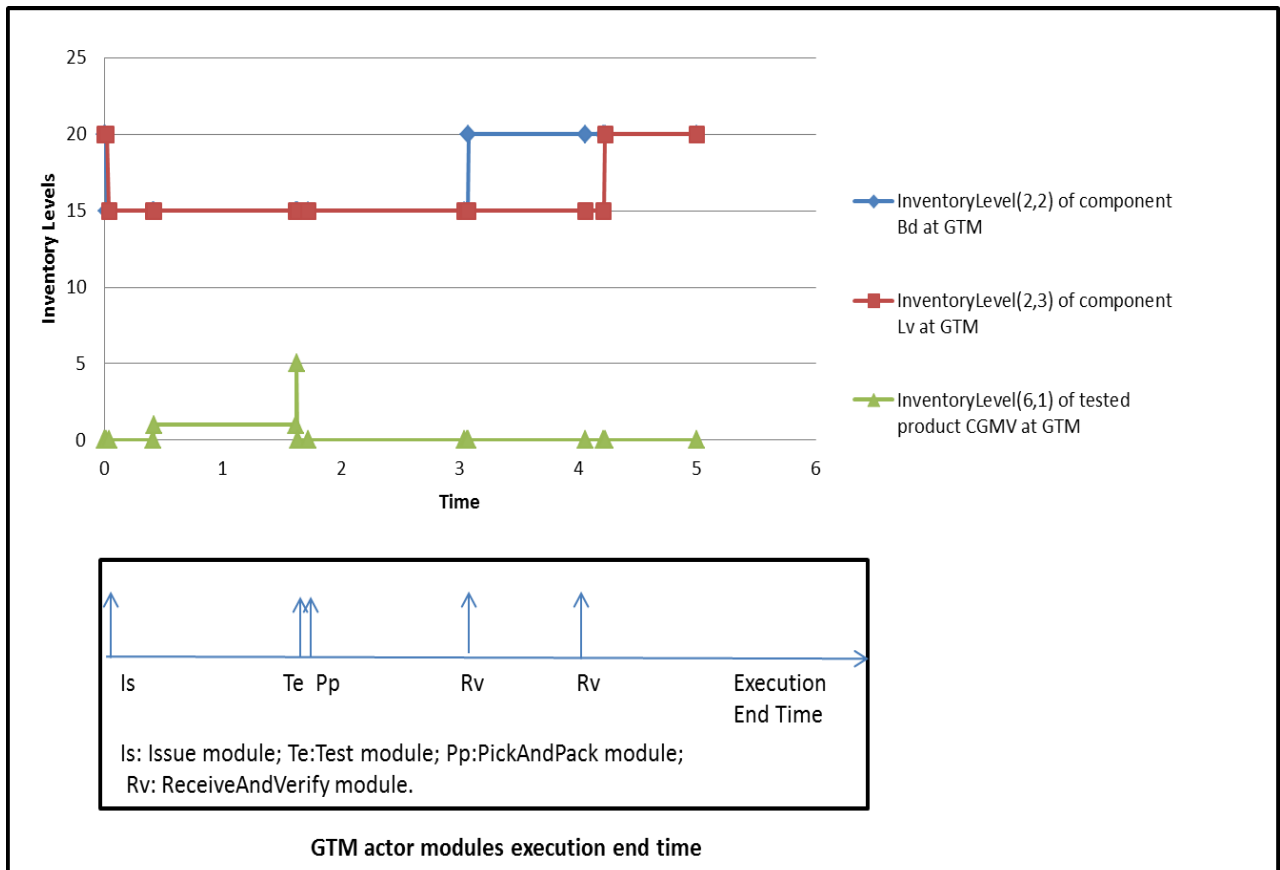


FIGURE 5.18: TEMPORAL VARIATION OF GTM INVENTORY LEVELS AFTER EXECUTING MODULES

- The verification results

The analysis of the test case results consists in verifying that the outputs after each Operation module execution are equal to the values calculated analytically. To give an example, we describe the analysis of the outputs of the ISSUEMATERIAL Operation module's instance named IssueTrucksParts belonging to GTM.

After receiving a ProductionOrder at the time ( $T=0$  days), the implemented algorithm of the IssueTrucksParts verifies the availability of sub-assemblies. Since the inventory level of the sub-assemblies Lv and Bd (20 for both) is greater than the quantity requested for issuing, the issuing is executed. Hence, the current time changes based on the number of used resources, the capacity and the cycle time. We deduce the module execution end time ( $T= 0.042$  days) and the new inventory levels ( $IL(2,2) = 15$ ) and ( $IL(2,3)=15$ ). The theoretical calculation of the time and quantity for the ISSUEMATERIAL Operation module are shown in Table 5.6.

TABLE 5.6: THEORETICAL CALCULATION OF THE OUTPUTS OF THE ISSUE MODULE

<p>New inventory levels calculation :</p> $IL(2,2) = IL(2,2) - Po.requestedQuantity \times BL(1,2) = 20 - 5 = 15$ $IL(2,3) = IL(2,3) - Po.requestedQuantity \times BL(1,3) = 20 - 5 = 15$ $IL(4,2) = IL(4,2) + Po.requestedQuantity \times BL(1,2) = 0 + 5 = 5$ $IL(4,3) = IL(4,3) + Po.requestedQuantity \times BL(1,3) = 0 + 5 = 5$ <p>New times calculation :</p> $Tf1 = Ti1 + transferResource.MoveTime(1,1) \times Arrondu.Sup ( Po.requestedQuantity \times BL(1,2) \div (TransferResource.Capacity(1,2) \times TransferResource.Number(1)))$ $Tf1 = 0 + 0.007 \times 3 = 0.021 \text{ days}$ $Tf2 = Tf1 + transferResource.MoveTime(1,1) \times Arrondu.Sup ( Po.requestedQuantity \times BL(1,3) \div (TransferResource.Capacity(1,3) \times TransferResource.Number(1)))$ $Tf2 = 0.021 + 0.007 * 3 = 0.042 \text{ days}$
---

To summarize, the experimental results provide the same

results as the theoretical calculation. This implies that the developed model fits with what is expected and that the verification is conclusive.

## 5.5 SIMULATING THE MODEL

In this section, we conduct a set of scenarios for evaluating the impacts of risks realization on SC performance. Hence, we start by explaining the evaluation of the simulation results, then we introduce the experimented scenarios and we finish by analyzing the found results.

- Defining the evaluation metrics

The evaluation of the experiments' results is based on the metrics provided by SCOR. This is to show how the usage of simulation permits putting in practices the SCOR proposal for evaluating SC processes.

TABLE 5.7: ADOPTED PERFORMANCES METRICS

Metrics	Formulas	Attributes
RL.2.1 % of Orders Delivered In Full	$[Total \text{ number of orders delivered in full}] / [Total \text{ number of orders delivered}] \times 100$	SC reliability
RL.2.2 Delivery Performance to Customer Commit Date	$[Total \text{ number of orders delivered on the original commitment date}] / [Total \text{ number of orders delivered}] \times 100$	
RS.1.1 Order Fulfillment Cycle Time	Average [order fulfillment process time per unit ]	SC responsiveness
Inventory Waiting Time	Average [time required for a given quantity to be consumed / consumed quantity ]	SC costs
Average Inventory per day	Sum over all days [inventory level per day]/ [Duration Of The Monitored Period In Days]	
Resource Utilization	[Time when resource is busy/ total time ]	SC asset management

---

The SCOR reference model defines five attributes for the SC performances. The first three attributes are considered customer focused (SC reliability, SC responsiveness, and SC agility), the latter are internally focused (SC costs and SC asset management). In order to get a complete picture of what is impacted in the SC, we propose a set metrics to cover all the above-mentioned SCOR performance attributes. Therefore, when it is possible we pick at least one metric from the metrics relative to the SCOR performance attributes (except the agility attribute) otherwise we define our proper metrics. The properly defined metrics do not have SCOR identification. The proposed performance metrics are shown in Table 5.7. For each performance metric, we provide the calculation formulas and the attribute to which the performance metric belongs. Defining a scenario

We conduct a set of experiments for evaluating the impacts of SC risk realization on performances. The first scenario is the base scenario that aims to evaluate the performances of the SC in normal conditions. To assess the impacts, the performances registered for the base scenario are compared with the performances of the risk scenarios. Namely, we conduct four risk experiments, the first three are relative to the individual risk events Supply Delay, Supply Cease and Errors In Edited Purchase Orders while the fourth one is relative to a combination of the risk events Supply Cease and Errors In Edited Purchase Orders. We made this selection to cover the 3 risks classes defined in section 3.2 of chapter 3 and to demonstrate the easiness to create more complex scenarios.

To define a scenario, we need first to set the configuration of the simulation model and then we need to set the parameters of the SC risk scenarios. The simulation model is configured by setting the following parameters:

- Resource (Number, capacity (per product)),
- Time (per Product or per path or per route),
- Stock (Initial inventory Level, replenishment level (if any), and target level (if any)),
- Demand arrival (product, quantity, inter-arrivals time).

The parameters of the current SC simulation model are specified in tables 5.8, 5.9 and 5.10 for resources, table 5.11 for stocks and table 5.12 for demand arrival.

Then the risk parameters are set as follows: Risk (SC Risk Names, SC Risk types, Start times, End times, Magnitudes and the impacted elements (Impacted Facilities, Impacted Properties, Impacted Operations and the Impacted Flows)).

Five risk configurations are tested. We test three single risks belonging to three different groups for evaluating the effects of each group apart. Furthermore, we test a combination of risks for evaluating their joint effects. The parameters of the SC risk scenarios are explained in table 5.13.

In the experiments, we propose to use deterministic data for SC risks so that at each replication we have the same appearance conditions. For example, for the Supply Delay Risk, we set the *Magnitude* property to 20 which means that the delay will last 20 days when it occurs. The user may of course set stochastic values for these parameters. For each experiment, 12 replications are conducted to achieve a 99 % confidence interval. The number of replications (12) is determined through the statistical analysis of the RL.2.2 The Delivery



Performance To Customer Commit Date using a significance level ( $\alpha = 1\%$ ) for the confidence interval calculation. Each replication has a duration of 60 days.

TABLE 5.8: RESOURCES SETTINGS

Names	Resource 1	Resource 2	Resource 3	Resource 4	Resource 5	Resource 6	Resource 7	Resource 9	Resource 10
<b>Identifiers</b>	1	2	3	4	5	6	7	9	10
<b>Designations</b>	Receive and Verify resource of GTM	Produce station of GTM	Test station of GTM	Pick And Pack resource of GTM	Receive and Verify resource of DistC1	Receive and Verify resource of DistC2	Loading resource of GTM	Loading resource of Supplier1	Loading resource of Supplier2
<b>Numbers</b>	1	1	1	1	1	1	1	1	1
<b>Per Products</b>	<b>Cycle Time (Days) per product</b>								
CGMV	0	NORM (0.265,0.03)	NORM (0.09,0.02)	0.05	0.04	0.04	0.004	0	0
Bd	0.025	0	0	0	0	0	0	0.004	0
Lv	0.03	0	0	0	0	0	0	0	0.004
<b>Per Products</b>	<b>Capacity per product</b>								
CGMV	0	1	1	1	1	1	1	0	0
Bd	1	0	0	0	0	0	0	1	0
Lv	1	0	0	0	0	0	0	0	1

TABLE 5.9: TRANSPORTATION RESOURCES SETTINGS

Names	Resource 12	Resource 14	Resource 15
<b>Identifiers</b>	1	3	4
<b>Numbers</b>	1	2	2
<b>Designations</b>	Shipping vehicle of GTM	Shipping vehicles of DistC1	shipping vehicles of DistC2
<b>Per Products</b>	<b>Capacities per product</b>		
CGMV	30	0	0
Bd	0	30	0
Lv	0	0	30
<b>Per Routes</b>	<b>Trip Times (Days) per route per product</b>		
1	0.5	0	0
2	1	0	0
3	0	1	0
4	0	0	1

TABLE 5.10: TRANSFER RESOURCES SETTINGS

<b>Names</b>	Resource 11
<b>Identifiers</b>	1
<b>Numbers</b>	1
<b>Designations</b>	Issuing resource of GTM
<b>Per Products</b>	<b>Capacities per product</b>
CGMV	0
Bd	2
Lv	2
<b>Per Paths</b>	<b>Move Times (Days) per path</b>
1	0.007

TABLE 5.11: BUFFERS SETTINGS

<b>Identifiers</b>	<b>Designations</b>	<b>Product Identifiers</b>	<b>Product Names</b>	<b>Inventory levels</b>	<b>Replenishment Inventory levels</b>	<b>Target levels</b>
<b>12</b>	<b>DistC1 Buffer of received trucks from GTM</b>	1	CGMV	12	6	12
<b>15</b>	<b>DistC2 Buffer of received trucks from GTM</b>	1	CGMV	12	6	12
<b>2</b>	GTM Buffer of non-defective received and verified trucks from suppliers	2	Lv	20	25	30
		3	Bd	20	25	30

TABLE 5.12: DEMAND ARRIVALS (FINAL CLIENT'S DEMANDS)

<b>Received By</b>	<b>Products</b>	<b>Quantities</b>	<b>inter-arrivals times</b>
DistC1	CGMV	Norm(5,1)	3 days
DistC2	CGMV	Norm(4,1)	3 days

TABLE 5.13: RISK EXPERIMENTS

	SC Risk Names	SC Risk types	Start times	End times	Magnitudes	Impacted elements			
						Impacted Facilities	Impacted Properties	Impacted Operations	Impacted Flows
R 1	Rk1: Supply delay	Operation mode change risk	10 (days)	30	20 (days) of duration	NA	NA	ShipTrucksL evers (of Supplier2)	NA
R 2	Rk2: Supply cease	Object Destruction Risk	30 (days)	NA	NA	NA	NA	NA	ShippingOrder (of Supplier1)
R 3	Rk3: Error in purchase order quantity	Property Change Risk	30 (days)	41 (days)	+ 30 units	GTM	required Quantity (of GTM Purchase orders)	NA	NA
R 4	RK2 + RK3		30 (days)	0	NA	NA	NA	NA	NA
			30 (days)	3 consecutive orders	+ 30 units	GTM	required Quantity (of GTM Purchase orders)	NA	NA
R5	Rk1 + Rk2		10 (days)	30	20 (days) of duration	NA	NA	ShipTrucksL evers (of Supplier2)	NA
			30 (days)	NA	NA	NA	NA	NA	ShippingOrder (of Supplier1)

- Experimental results

Each simulation experiment's results are put in a separate table. Hence, Table 5.14 shows the results of the base scenario experiment. Tables 5.15 to 5.20 show the results for the risk scenarios R1 to R5 (see table 5.13), respectively.

For each measured performance metric, we provide the minimum, the average and the maximum values for the 12 conducted simulation replications. Results are grouped by Actors. Some of the metrics are concerned with evaluating a feature of the Actor's relations (e.g. RL2.2 % Delivery Performance To Customer Commit Date). Some of them are relative to a particular Product (e.g. Inventory Waiting Time) and some of them are relative to a particular resource. We specify the relation, the Product name, and resource name in separated cells

joining the Actor and concerned metrics. The suppliers 1 and 2 are noted respectively S1 and S2. In some cases, the calculation of a metric is not applicable, so we mention (NA) in the relative cell. For instance, the calculation of the metric RS1.1 Order Fulfillment Cycle Time Per Unit is not applied to the Actors DistC1 and DistC2 since the delivery of trucks CGMV to the final clients is assumed to be done immediately in this example. Moreover, the calculation of the metric RS1.1 Order Fulfillment Cycle Time is not applied for the Actor and for the Product facing a complete disruption, since it is not possible to define the Product delivery time. When the calculation of some metrics is not interesting for the analysis we indicate not calculated (Nc). For instance, we restrict the calculation of the resource utilization metric to the most important GTM resource which is the production resource.

Furthermore, each performance metric is mapped within a graph. The registered values for RL2.2 % Delivery Performance To Customer Commit Date are shown in figure 5.19, the values for the Average Inventory Per Day Level are shown in figure 5.20, the values for the Orders Delivered In Full are shown in figure 5.21, the values for the Resource Utilization are shown in figure 5.22.

TABLE 5.14: RESULTS FOR THE BASE SCENARIO

Metrics	RL2.1 % of Orders Delivered In Full [Min,Avg,Max]	RL2.2 % Delivery Performance To Customer Commit Date [Min,Avg,Max]	RS1.1 Order Fulfillment Cycle Time [Min,Avg,Max]	Inventory Waiting Time [Min,Avg,Max]	Average Inventory Per Day [Min,Avg,Max]	Resource Utilization [Min,Avg,Max]
GTM	For relation: GTM- DistC1			For GTM manufactured product: CGMV		For GTM resource: Resource 2 [0.66,0.71,0.81]
	[100,100,100]	[70,93.83,100]	[0.50,0.56,0.77]	Nc	[1.06,1.56,2.73]	
	For relation: GTM- DistC2			For GTM sourced sub-assembly: Bd		
	[100,100,100]	[63.6, 82.4,100]	[0.76,0.90,1.03]	[0.80,1.09,1.28]	[15.5,16.58,17.7]	
DistC1	For relation: DistC1-FClient			For DistC1 sourced product: CGMV		Nc
	[85.7,95.8,100]	[100,100,100]	NA	Nc	[7.1,8.62,10.8]	
DistC2	For relation: DistC2-FClient			For DistC2 sourced product: CGMV		Nc
	[86.6,97.77,100]	[100,100,100]	NA	Nc	[9.31,10.28,11.16]	
S2	For relation: Supplier2-GTM			For Supplier2 sold product: Lv		Nc
	[100,100,100]	[100,100,100]	[0.40,0.45,0.5]	Nc	Nc	
S1	For relation: Supplier1-GTM			For Supplier1 sold product: Bd		Nc
	[100,100,100]	[100,100,100]	[0.49,0.55,0.6]	Nc	Nc	

TABLE 5.15: RESULTS FOR THE SUPPLY DELAY (R1)

Metrics	RL2.1 % of Orders Delivered In Full [Min,Avg,Max]	RL2.2 % Delivery Performance To Customer Commit Date [Min,Avg,Max]	RS1.1 Order Fulfillment Cycle Time [Min,Avg,Max]	Inventory Waiting Time [Min,Avg,Max]	Average Inventory Per Day [Min,Avg,Max]	Resource Utilization [Min,Avg,Max]
GTM	For relation: GTM- DistC1			For GTM manufactured product: CGMV		For GTM resource: Resource 2 [0.54, 0.62,0.67]
	[100,100,100]	[50,62.4,71.4]	[2.1,3.40,4.2]	Nc	[2.51,2.78,2.98]	
	For relation: GTM- DistC2			For GTM sourced sub-assembly: Bd		
	[100,100,100]	[30.7,50.42,75.0]	[3.0,4.06,5.3]	Nc	[13.2,14.4,16.7]	
DistC1	For relation: DistC1-FClient			For DistC1 sourced product: CGMV		Nc
	[71.4, 76.7, 85.7]	[100,100,100]	NA	Nc	[ 6.68,7.18,8.02]	
DistC2	For relation: DistC2-FClient			For DistC2 sourced product: CGMV		Nc
	[60, 79.4, 100]	[100,100,100]	NA	Nc	[ 6.48,8.58,10.80]	
S2	For relation: Supplier2-GTM			For Supplier2 sold product: Lv		Nc
	[100,100,100]	[79.1,87.9,92.0]	[0.55,0.74,0.86]	Nc	Nc	
S1	For relation: Supplier1-GTM			For Supplier1 sold product: Bd		Nc
	[100,100,100]	[91.6,99.3,100]	[0.42,0.44,0.48]	Nc	Nc	

TABLE 5.16: RESULTS FOR THE SUPPLY CEASE (R2)

Metrics	RL2.1 % of Orders Delivered In Full [Min,Avg,Max]	RL2.2 % Delivery performance to customer commit date [Min,Avg,Max]	RS1.1 Order fulfillment cycle time [Min,Avg,Max]	Inventory waiting time [Min,Avg,Max]	Average Inventory Per Day [Min,Avg,Max]	Resource Utilization [Min,Avg,Max]
GTM	For relation: GTM- DistC1			For GTM manufactured product: CGMV		For GTM resource: Resource 2 [0.45,0.51,0.62]
	[100,100,100]	[52.8,61.1,70.4]	NA	Nc	[0.88,1.62,2.67]	
	For relation: GTM- DistC2			For GTM sourced sub-assembly: Bd		
	[100,100,100]	[32.3,48.26,60.5]	NA	NA	[9.6,11.18,12.5]	
DistC1	For relation: DistC1-FClient			For DistC1 sourced product: CGMV		Nc
	[42.8,48.2,57.1]	[100,100,100]	NA	Nc	[4.01,4.51,5.35]	
DistC2	For relation: DistC2-FClient			For DistC2 sourced product: CGMV		Nc
	[46.6,56.6,66.6]	[100,100,100]	NA	Nc	[5.03,6.12,7.20]	
S2	For relation: Supplier2-GTM			For Supplier2 sold product: Lv		Nc
	[100,100,100]	[100,100,100]	[0.40 ,0.48, 0.53]	Nc	Nc	
S1	For relation: Supplier1-GTM			For Supplier1 sold product: Bd		Nc
	[100,100,100]	[63.6,71.7,81.8]	NA	Nc	Nc	

TABLE 5.17: RESULTS FOR AN ERROR IN THE PURCHASE ORDER QUANTITY (R3)

Metrics	RL2.1 % of Orders Delivered In Full [Min,Avg,Max]	RL2.2 % Delivery Performance To Customer Commit Date [Min,Avg,Max]	RS1.1 Order Fulfillment Cycle Time [Min,Avg,Max]	Inventory Waiting Time [Min,Avg,Max]	Average Inventory Per Day [Min,Avg,Max]	Resource Utilization [Min,Avg,Max]
GTM	For relation: GTM- DistC1			For GTM manufactured product: CGMV		For GTM resource:Resource 2 [0.65,0.71,0.81]
	[100,100,100]	[70.0,93.14,100]	[0.50,0.56,0.77]	Nc	[0.98,1.60,2.73]	
	For relation: GTM- DistC2			For GTM sourced sub-assembly: Bd		
	[100,100,100]	[63.6,82.6,100]	[0.78,0.90,1.03]	[0.78,1.06,1.15]	[14.8,16.1,18.7]	
DistC1	For relation: DistC1-FClient			For DistC1 sourced product: CGMV		Nc
	[85.5,95.8,100]	[100,100,100]	NA	Nc	[7.1,8.56,10.8]	
DistC2	For relation: DistC2-FClient			For DistC2 sourced product: CGMV		Nc
	[86.6,97.7,100]	[100,100,100]	NA	Nc	[9.3,10.23,11.1]	
S2	For relation: Supplier2-GTM			For Supplier2 sold product: Lv		Nc
	[100,100,100]	[26.6,83.3,94.7]	[0.37,0.41,0.51]	Nc	Nc	
S1	For relation: Supplier1-GTM			For Supplier1 sold product: Bd		Nc
	[100,100,100]	[94.1,99.0,100]	[0.49,0.57,0.65]	Nc	Nc	

TABLE 5.18: RESULTS FOR COMBINATION OF AN ERROR FOR PURCHASE ORDER QUANTITY AND A SUPPLY CEASE (R4)

Metrics	RL2.1 % of Orders Delivered In Full [Min,Avg,Max]	RL2.2 % Delivery Performance To Customer Commit Date [Min,Avg,Max]	RS1.1 Order Fulfillment Cycle Time [Min,Avg,Max]	Inventory Waiting Time [Min,Avg,Max]	Average Inventory Per Day [Min,Avg,Max]	Resource Utilization [Min,Avg,Max]
GTM	For relation: GTM- DistC1			For GTM manufactured product:		For GTM resource: Resource 2 [0.45,0.51,0.62]
	[100,100,100]	[52.8,61.1,70.4]	NA	CGMV		
	For relation: GTM- DistC2			For GTM sourced sub-assembly: Bd		
	[100,100,100]	[32.3,48.18,60.5]	NA	NA	[9.1,10.55,12.5]	
DistC1	For relation: DistC1-FClient			For DistC2 sourced product: CGMV		Nc
	[42.8,48.2,57.1]	[100,100,100]	NA	Nc	[3.55,4.00,4.74]	
DistC2	For relation: DistC2-FClient			For DistC1 sourced product: CGMV		Nc
	[46.6,56.6,66.6]	[100,100,100]	NA	Nc	[5.00,6.08,7.15]	
S2	For relation: Supplier2-GTM			For Supplier2 sold product: Lv		Nc
	[100,100,100]	[33.3,78.5,90.9]	[34.8,39.3,51.05]	Nc	Nc	
S1	For relation: Supplier1-GTM			For Supplier1 sold product: Bd		Nc
	[100,100,100]	[60.0,69.7,75.0]	NA	Nc	Nc	

TABLE 5.19: RESULTS FOR COMBINATION OF A SUPPLY DELAY AND A SUPPLY CEASE (R5)

Metrics	RL2.1 % of Orders Delivered In Full [Min,Avg,Max]	RL2.2 % Delivery Performance To Customer Commit Date [Min,Avg,Max]	RS1.1 Order Fulfillment Cycle Time [Min,Avg,Max]	Inventory Waiting Time [Min,Avg,Max]	Average Inventory Per Day [Min,Avg,Max]	Resource Utilization [Min,Avg,Max]
GTM	For relation: GTM- DistC1			For GTM manufactured product: CGMV		For GTM resource: Resource 2 [0.30,0.33,0.37]
	[100,100,100]	[21.4,22.0,28.5]	NA	Nc	[0.57,0.76,0.98]	
	For relation: GTM- DistC2			For GTM sourced sub-assembly: Bd		
	[100,100,100]	[13.3,22.2,33.3]	NA	NA	[16.4,17.4,18.6]	
DistC1	For relation: DistC1-FClient			For DistC1 sourced product: CGMV		Nc
	[35.7,45.8,57.1]	[100,100,100]	NA	Nc	[2.36,3.62,5.79]	
DistC2	For relation: DistC2-FClient			For DistC2 sourced product: CGMV		Nc
	[40,54.4,66.6]	[100,100,100]	NA	Nc	[3.18,4.21,6.21]	
S 2	For relation: Supplier2-GTM			For Supplier2 sold product: Lv		Nc
	[100,100,100]	[57.1,63.3,71.4]	[1.35,1.82,2.22]	Nc	Nc	
S1	For relation: Supplier1-GTM			For Supplier1 sold product: Bd		Nc
	[100,100,100]	[50,56.5,62.5]	NA	Nc	Nc	

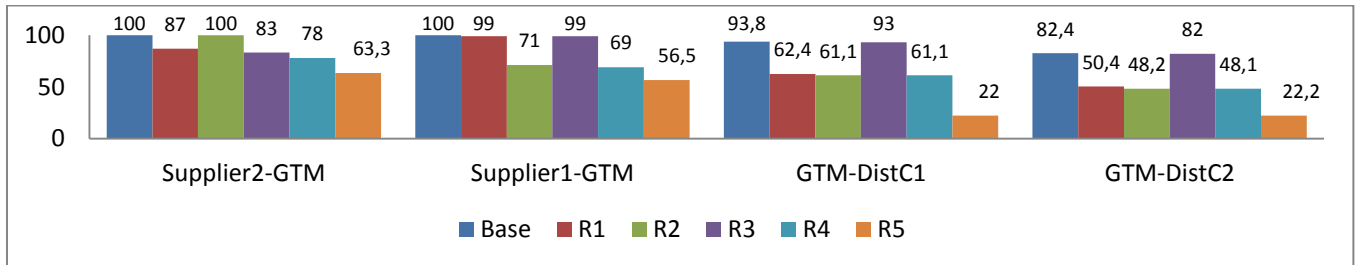


FIGURE 5.19: IMPACTS ON THE RL2.2 % DELIVERY PERFORMANCE TO CUSTOMER COMMIT DATE

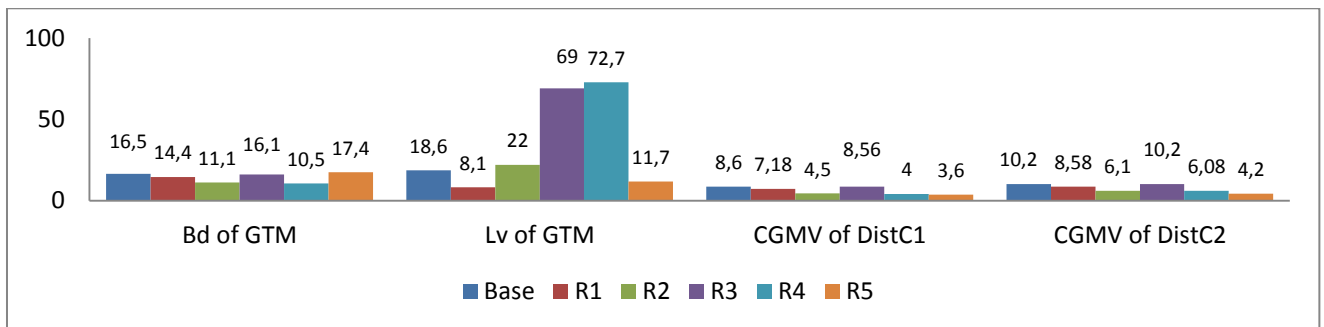


FIGURE 5.20: IMPACTS ON THE AVERAGE INVENTORY PER DAY



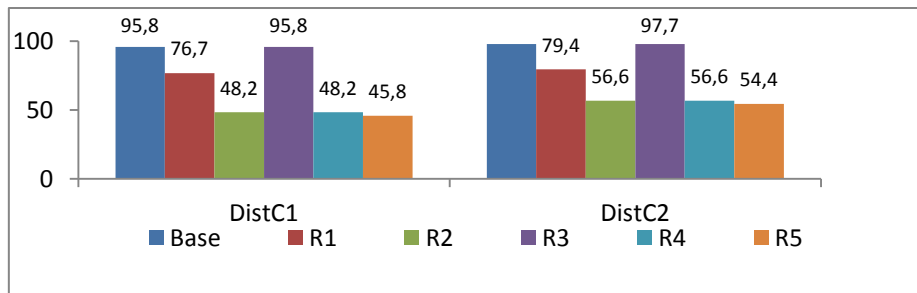


FIGURE 5.21: IMPACTS ON THE RL1.2 % OF ORDERS DELIVERED IN FULL

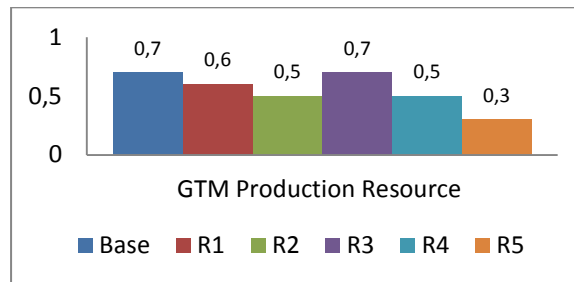


FIGURE 5.22: IMPACTS ON THE PRODUCTION RESOURCE UTILIZATION

- **Analysis of experiments on R1: Supply Delay Risk**

The Supply delay risk tested here is an **Operation Mode Risk** that acts by modifying the functioning mode of the SHIP Operation belonging to Supplier 2. The dysfunctional mode operates by adding a delay to the shipping time.

In comparison to the base scenario results, shown in table 5.14, the noticed effects of the risk from the results registered in table 5.15 are as follows:

First, the risk caused the failure of Supplier2 in meeting a part of its customers' requests (Drop of the metric %Performance To Commit Date from 100% to 89.7% as shown in figure 5.19). Furthermore, it caused the failure of GTM in meeting a part of its customers' requests (Drop in the %Performance To Commit Date from 93.83 to 62.4 for DistC1 and from 82.4 to 50.42 for DistC2 as shown in figure 5.19). This is due to the sold out of its sub-assembly Lv (Drop of the metric Average Inventory Per Day from 18.56 to 8.18 as shown in figure 5.20) that impacted its production capability (Drop of the metric Resource Utilization from 0.71 to 0.6 as shown in figure 5.22) and caused late deliveries.

Moreover, the risk caused the failure of DistC1 and DistC2 in delivering their final clients demands (Drop in the % of Orders Delivered In Full from 100 to 76.7 for DistC1 and from 100 to 79.4 for DistC2 as shown in figure 5.21). This is due to the sold out of the CGMV product.

- **Analysis of results on R2: Supply Cease Risk**

The supply cease risk is an **Object Destruction Risk** that acts by destroying an object in the model (here Supplier 1). The destruction of Supplier1 is done through cutting the shipping flow exchanged between Supplier 1 and GTM.

---

In comparison to the base scenario results, shown in table 5.14, the noticed effects of the risk “Supply cease” from the results registered in table 5.16 are as follows:

First, the capability of Supplier1 in meeting its customers’ requests was greatly impacted ( Drop in the %Performance To Commit Date from 93.83 to 61.1 for DistC1 and from 82.4 to 48.26 for DistC2 as shown in figure 5.19). As a consequence, GTM encountered sold outs of its sub-assembly Bd (Drop of the metric Average Inventory Per Day from 15.58 to 11.18 as shown in figure 5.20) which impacted its production capability (Drop of the metric Resource Utilization from 0.71 to 0.51 as shown in figure 5.22) and consequently reduced the produced quantities (Drop in CGMV Average Inventory Per Day from 1.56 to 1.06 as shown in figure 5.20). Therefore, GTM encountered difficulties in meeting its customers’ requests (Drop in the %Performance To Commit Date from 93.83 to 48.2 for DistC1 and from 82.04 to 56.6 for DistC2 as shown in figure 5.19). Consequently, DistC1 and DistC2 encountered a sold out of the CGMV product. This impacted their capability in delivering their final clients demands (Drop in the % of Orders Delivered In Full from 100 to 48.2 for DistC1 and from 100 to 56.6 for DistC2 as shown in figure 5.21).

- **Analysis of results on R3: Errors In Edited Purchase Order Quantity Risk**

Here, we test a **Property Change Risk** that acts on the property of an object. The affected property is the *requestedQuantity* of the edited “PurchaseOrder” object exchanged between the Supplier 2 and GTM. This is to analyze how errors in editing purchase order quantity may affect the system performance.

In comparison to the base scenario results shown in table 5.14, the noticed effects of this type of risk are shown in table 5.17.

First, the risk impacted the capability of the Supplier 2 of delivering GTM requests in time (Drop of the %Performance To Commit Date metric from 100 % to 83.3 % as shown in figure 5.19). This is due to unprepared supplier (Supplier2) that mismanaged the excessive ordered quantity of the sub-assembly Lv.

Furthermore, the risk slightly impacted the capability of GTM in the delivery of its customers’ orders (Drop of the metric %Performance To Commit Date from 94.10% to 93.4 % for the relation GTM-DistC1 and from 80.29% to 78.7% for the relation GTM-DistC2 as shown in figure 5.19). Also, the risks increased the inventory holding costs of GTM (Increase of Average Inventory Per Day of “Lv” from 18.5 to 45.1 as shown in figure 5.20).

- **Analysis of results on R4: Combined risks on Errors In Edited Purchase Order Quantity R3 and the Supply Cease R2**

We now analyze joint realization of R2 and R3. In comparison to the Supply Cease risk scenario (R2) shown in table 5.16 and the Errors In Edited Purchase Order Quantity risk scenario (R3) shown in table 5.17 the noticed effects for the combination of these two risks from the results shown in table 5.18 are as follows:

The joint realization of the two risks the Supply Cease risk applied on Supplier 1 and the Errors In Edited Purchase Order risk applied on the *requestedQuantity* property of the PurchaseOrder object exchanged between Supplier2 and GTM leads to negative effects on SC performances. Neither a compensation nor an amplification of effects emerged from the

---

combination but rather the effects of individual risks appear. For instance, we find that the metric Average Inventory Per Day increased with the same value as for the individual execution of the risk Errors In Edited Purchase Order for Product Lv of as shown in figure 5.20.

- **Analysis of results on R5: Combined risks on Supply Delay R1 and Supply Cease R2**

In comparison to the **Supply Delay Risk R1** scenario shown in table 5.15 and to the **Supply Cease Risk R2** scenario shown in table 5.16 the noticed effects for the combination of these two risks from the results shown in table 5.19 are as follows:

The joint realization of the two risks Supply Delay and Supply Cease leads to an amplification of the negative effects on SC members. For instance, the Performance To Commit Date % for supplier 2 has dropped down to 63,3% for this combined risk, while it was 87% for R1 as shown in figure 5.19. This is caused by the modification in GTM demand profile that has adapted to the variation of the consumption profile of sub-assemblies. Moreover, we witness an amplification of negative impacts on the delivery performances of both DistC1 and DistC2 which decreased from 100 % to 45.5 % and from 100 % to 54.4 % respectively as shown in figure 5.21.

## CONCLUSION

In this chapter, we present a case study of the SC of a truck manufacturer (TruckMuch) located in Grenoble. First, we describe the model that captures the SC structure and the SC behavior. Second, we explain its translation into a simulation model. The created simulation model is then verified and a set of risk experiments are conducted. The analysis shows that the tool is effective in assessing the impacts of SC risks on SC performances. The impacts are seen in different SC levels. The impacts propagation through the SC levels is easily analyzed. The risk analysis enables the SC practitioner to sort the risks by their importance in order to prioritize them in terms of countermeasures implementation and in terms of investments.

---

**CHAPTER**

**6**

**CONCLUSION**

---

## CHAPTER 6: CONCLUSION

In this chapter we provide an overview of the major results and the findings of this dissertation. Then, we expose the major constraints and the limits of this work, as well as the perspectives.

### 6.1 RESOLVED RESEARCH QUESTIONS AND MAIN FINDINGS

The focus in this dissertation is to assist the SC practitioner in analyzing the risks threatening his/her SC by using simulation. To this end, we searched the weaknesses of the existing modeling frameworks for simulation proposed in the literature, in order to propose an easier to use framework and useful modeling guidelines for the SC practitioners. We show that difficulties may arise from the complexity of using directly the simulation languages/models that may be hard to master for the SC professional. To drive the SC analysis the practitioner needs to conduct several steps from burdening its system of interest, eliciting its structure and functioning, modeling it in a simulation language and defining & testing relevant scenarios. We assume that directly performing these tasks with a simulation model is not easy. The nature of simulation language does not help in structuring the model since it mixes system description and simulation execution mechanisms. We thus proposed to use an intermediary model to structure the modeling approach and support the simulation model creation. We hence dealt with supporting the creation of SC conceptual models, modeling risks and supporting the creation of simulation model.

The first contribution is assisting the SC practitioners building a conceptual model for SCs and risks translatable into a simulation model.

More precisely, the reviewed literature works highlight a problem of adoption of simulation despite its interest for SC risk analysis. Most of the researchers (Wu et al. (2006), Cigolini et al. (2010),...) explain it by the lack of user friendly simulation tools and by the difficulty of constructing simulation models from scratch using the current simulation formalisms (DES, ABS, SDS). This makes modeling for simulation a hard and time consuming task that requires learning efforts and advanced skills.

By looking deep in the problem, a cause is related to the fact that the simulation softwares' building blocks are far from the SC domain and have a low level of aggregation regarding the elements to be modeled.

Hence, some researchers tried to propose modeling frameworks for simulation that define meta-models for SCs. Nevertheless, the proposed frameworks do not satisfy all the expectations. For instance, some of the works failed in covering the SC domain knowledge, others failed in communicating a well-structured SC modeling building blocks. Furthermore, most of the frameworks do not include modeling constructs for risks: this is due to a lack of a consensus about the definition and the categorization of SC risks.

To overcome this lack, three research directions are followed:

- Building a modeling framework for simulation that is easily adopted by SC practitioners.
- Proposing generic modeling constructs capable of capturing the SC domain knowledge.
- Integrating the risks modeling.

---

The provided solution consists of a framework permitting an easy construction of SC models including risks. The framework defines a modeling language formed of the meta-model of the SC structure, the meta-model for the SC behavior, the meta-model for risks and a library of SC specific constructs. This choice was suggested by (Beamon 1998) and (Min & Zhou 2002) who reveal the need for a modeling language for describing and/or for dynamic analysis of SC scenarios.

Each part of the meta-model defines a set of interconnected building blocks presented as a SysML profile that can be instantiated to model a given part of the SC. The usage of SysML as a meta-modeling language increases the expressiveness of the meta-model and the perceived ease of use since it is dedicated to non-software systems description.

The meta-model of the SC structure is defined based on the analysis of the static elements used by the SC processes described by SCOR. The meta-model of the SC behavior and the library of SC domain specific Operation and Role are also defined based on SCOR. This increases the fidelity of the modeling constructs to the modeled reality and makes them easy to understand and more familiar to SC practitioners. We recall that SCOR reference model is one of the most commonly used references in industry. It provides a textual description of the SC processes associated with a set of performance metrics used to benchmark their operations.

Unlike the existing literature works, the constructs that we propose capture more advanced SC features than SCOR. In fact, we propose to define modeling constructs for the flows transferred between the SC functions described in SCOR as processes' inputs and processes' outputs. Also, we propose to extend the SCOR formalism to capture the relations and the interactions that exist between the SC partners through specific constructs. Furthermore, we propose to define detailed algorithms for the operations captured from the SCOR designation of processes. Finally, we propose a set of risk constructs mapped with the SC building blocks extracted from SCOR. The inclusion of risks enhances the analysis capability of the framework outputs and enlarges the scope of its use.

The second contribution is in the provided assistance for creating and experimenting SCs and risks simulation models.

Through the analyses of the literature, we found that few of the previous works provide a description of the translation from a conceptual model into a simulation model (such as the work of (Long 2014)). There are a few works that provide modeling building blocks specific to the SC domain, but without explaining how to translate them into simulation modules (such as the work of (Persson 2011)). Also, due to the lack of a consensus about the definition of risks and their grouping into categories, only few works provide generic conceptual or simulation models for SC risks that can be used to create simulation risk modules. Therefore, the treated risks are case specific. Only the work of (Saleh Ebrahimi et al. 2012) provides generic conceptual risk modules but without explaining their translation into simulation.

To overcome this lack, two research directions are followed:

- Defining an easy and quick translation of the conceptual model into a simulation model.
- Supporting SC practitioners in experimenting risk scenarios using the SC simulation model.

---

We propose a simulation framework that assists the SC practitioner making the move from this conceptual model to an executable model (e.g. a simulation model) for performance and risk analysis. The framework describes the adaptations to implement for representing a given scenario (e.g. a risk scenario) and for monitoring performances.

The provided solution consists of a translation guideline for creating the simulation variables for translating the SC structure elements, for creating Operation and Risk simulation modules and for creating simulation flow entities. The translation is illustrated through creating a library of simulation modules in ARENA.

The solution enables the SC practitioner to build rapidly and easily their own simulation models, by setting the values of the simulation variables, by instantiating the simulation modules, by connecting them and by parameterizing them. Furthermore, the description of the adaptations to implement on simulation models enables the SC practitioners to conduct various experiments.

The application of the developed solutions on a case study is used to demonstrate how they support a better and easier generation of a simulation model for risk analysis. The case study provides a first technical verification of the meta-model, the translation and the library of simulation modules. The simulation results analysis shows that the tool is effective in assessing the impacts of SC risks on SC performances.

## 6.2 LIMITS OF SCOR

The major constraints encountered in this dissertation are first the high level description provided in the SCOR reference model. Namely, it is difficult to give interpretation and to define Operation constructs or flow constructs when the SCOR description presents some lacks or when some of the inputs or outputs of the SCOR processes present some contradictions. An example of SCOR contradictions in inputs is that SCOR does not define inputs of type material (e.g, product) for some process elements responsible of moving materials. For instance, SCOR does not define an input of type material for the process element “sM1.2/sM2.2 issue material” responsible of moving components for production while it defines inputs of type materials (e.g, product, DefectiveProduct...) for the Source process element “sS1.2/sS2.2 receiveProduct”.

An example of a lack in SCOR description is the absence of explanation of the used inventory policies within the Process elements. For instance, the Source process element “sM1.2/sM2.2 issue material” does not provide details about how the inventory of issued components is managed and when to generate replenishment signals.

## 6.3 LIMITS ON THE COVERAGE OF THE PROPOSAL

The major limit of this work is that the provided library does not cover all the process elements described by SCOR. Furthermore, when defining the algorithms of the Operations constructs some assumptions are considered. Even if those assumptions are met in the real life, still the scope of application is reduced.



---

Another limit is that the provided modeling and simulation modules are defined in a dedicated DES formalism (e.g. ARENA). The SC practitioner needs to make some modifications to adapt them for other simulation formalisms.

## 6.4 PERSPECTIVES

One of the interesting directions that we can follow is to develop a library for SC risk management policies. As revealed in chapter 1 many authors highlight a need for defining reactive risk countermeasures to deal with the perturbations that face SCs (Ivanov et al. (2014)). Furthermore, only a few studies tackled the modeling issue of the countermeasures in general as stated by (Talluri et al. 2013). The integration of countermeasures may be done using the described adaptations on SC simulation models for experimenting scenarios.

Another direction is to tackle the integration of the modeling framework for simulation within the process of SC risk management of a focal company. Hence, a classical problem to tackle is modeling the data gathered from SC practitioners for feeding the simulation model.

A third direction is to tackle the issue of risk monitoring and collaboration. Indeed, as revealed by the study made by MIT and Pwc in 2013 monitoring is required for assuring the maturity of the SC risk management process. This is by providing architecture for the data to be monitored and shared between the SC members and the associated procedures.

A fourth research direction is to tackle the issue of cyber supply chains (such as AMAZON, ALIBABA...) and to study the usage of modeling and simulation for the analysis of their risks and their activities and for their optimization.

---

## BIBLIOGRAPHY

- Adhitya, A., Srinivasan, R. & Karimi, I.A., 2008. Supply Chain Risk Management through HAZOP and Dynamic Simulation. *18th European Symposium on Computer Aided Process Engineering (ESCAPE 18)*, pp.37–42.
- Alaca, H. & Ceylan, C., 2011. Value Chain Analysis using Value Stream Mapping : White Good Industry Application. *Proceedings - 2011 International Conference on Industrial Engineering and Operations Management*.
- Albores, P. et al., 2006. An evaluation of SCOR modelling techniques and tools. *Proceedings of the Second European Conference on the Management of Technology*, (November 2015).
- Aqlan, F. & Mustafa Ali, E., 2014. Integrating lean principles and fuzzy bow-tie analysis for risk assessment in chemical industry. *Journal of Loss Prevention in the Process Industries*, 29(1), pp.39–48.
- Arisha, A. & Mahfouz, A., 2010. The Analysis of Rush Orders Risk in Supply Chain: a Simulation Approach. *Modism World Conference*, pp.161–174.
- Asgari, N. et al., 2016. Supply chain management 1982-2015: A review. *IMA Journal of Management Mathematics*, 27(3), pp.353–379.
- Baines, T.S. & Harrison, D.K., 1999. An opportunity for system dynamics in manufacturing system modelling. *Production Planning & Control*, 10(6), pp.542–552.
- Beamon, B.M., 1998. Supply chain design and analysis:: Models and methods. *International Journal of Production Economics*, 55(3), pp.281–294.
- Berger, P.D., Gerstenfeld, A. & Zeng, A.Z., 2004. How many suppliers are best? A decision-analysis approach. *Omega*, 32(1), pp.9–15.
- Blackhurst et al., 2005. An empirically derived agenda of critical research issues for managing supply-chain disruptions. *International Journal of Production Research*, 43(19), pp.4067–4081.
- Blackhurst, J. V, Scheibe, K.P. & Johnson, D.J., 2008. Supplier risk assessment and monitoring for the automotive industry. *International Journal of Physical Distribution & Logistics Management*, 38(2), pp.143–165.
- Blos, M.F. & Miyagi, P.E., 2015. Modeling the supply chain disruptions: A study based on the supply chain interdependencies. *IFAC-PapersOnLine*, 28(3), pp.2053–2058.
- Bogataj, D. & Bogataj, M., 2007. Measuring the supply chain risk and vulnerability in frequency space. *International Journal of Production Economics*, 108(1–2), pp.291–301.
- Brandenburg, M. et al., 2014. Quantitative models for sustainable supply chain management: Developments and directions. *European Journal of Operational Research*, 233(2), pp.299–312.
- Brown, G.W., 2009. Value Chains, Value Streams, Value Nets, and Value Delivery Chains. *Business Process Trends*, (April), pp.1–12.
- Bullinger, H. & Kühner, M., 2010. Analysing supply chain performance using a balanced measurement method. *International Journal of Production Research (ABS2015:3)*, (March 2013), pp.37–41.
- Byrne, P.M., 2007. Impact and ubiquity: two reasons to proactively manage risk. *Logistics Management*, (Apri), pp.24–25.
- Cai, J. et al., 2009. Improving supply chain performance management: A systematic approach to analyzing iterative KPI accomplishment. *Decision Support Systems*, 46(2), pp.512–521.
- Campuzano, F., & Bru, J.M., 2011. *Supply chain simulation: A system dynamics approach for improving performance*, Springer Science & Business Media.
- Carvalho, H. et al., 2012. Supply chain redesign for resilience using simulation. *Computers and Industrial Engineering*, 62(1), pp.329–341.

- 
- Casella, F., Miragliotta, G. & Uglietti, L., 2005. Analysis of Supply Chain Dynamics through Object Oriented Simulation. *Research Methodologies in Supply Chain Management*, pp.461–476.
- Cavinato, J.L., 2004. Supply chain logistics risks: From the back room to the board room. *International Journal of Physical Distribution & Logistics Management*, 34(5), pp.383–387.
- Chan, F.T.S. et al., 2002. A simulation approach in supply chain management. *Integrated Manufacturing Systems*, 13(2), pp.117–122.
- Chatfield, D.C., Hayya, J.C. & Harrison, T.P., 2007. A multi-formalism architecture for agent-based, order-centric supply chain simulation. *Simulation Modelling Practice and Theory*, 15(2), pp.153–174.
- Chen, F. et al., 2000. Quantifying the Bullwhip Effect in a Simple Supply Chain: The Impact of Forecasting, Lead Times, and Information. *Management Science*, 46(3), pp.436–443.
- Chen, J., Sohal, A.S. & Prajogo, D.I., 2012. Supply chain operational risk mitigation: a collaborative approach. *International Journal of Production Research*, 7543(June), pp.1–14.
- Chopra, S. & ManMohan, S.S., 2014. Managing risk to avoid supply-chain breakdown. *MIT Sloan management review*, 34(5), pp.360–387.
- Chopra, S. & Meindl, P., 2007. Supply chain management. Strategy, planning & operation. In *Das summa summarum des management*. pp. 265–275.
- Christopher, M. & Lee, H., 2004. Mitigating Supply Chain Risk Through Improved Confidence. *International Journal of Physical Distribution & Logistics Management*, 34(5), pp.388–396.
- Christopher, M. & Peck, H., 2004. Building the resilient supply chain. *The International Journal of Logistics Management*, 15(2).
- Christopher, M. & Towill, D.R., 2000. Supply chain migration from lean and functional to agile and customised. *Supply Chain Management: An International Journal*, 5(4), pp.206–213.
- Cigolini, R., Pero, M. & Rossi, T., 2011. An object-oriented simulation meta-model to analyse supply chain performance. *International Journal of Production Research*, 49(19), pp.5917–5941.
- Cimino, A., Longo, F. & Mirabelli, G., 2010. A General Simulation Framework for Supply Chain Modeling: State of the Art and Case Study. *International Journal of Computer Science Issues*, 7(2), pp.1–9.
- Clark, T., Sammut, P. & Willans, J., 2015. *Applied metamodelling: a foundation for language driven development* arXiv:1505., arXiv preprint.
- Coad, P., Yourdon, E. & Coad, P., 1991. *Object-oriented design* NJ: Yourdon press., ed., Englewood Cliffs.
- Cope, D. et al., 2007. Supply chain simulation modeling made easy: An innovative approach. *2007 Winter Simulation Conference*, pp.1887–1896.
- Dai, H., Lin, J. & Long, Q., 2014. A fractal perspective-based methodological framework for supply chain modelling and distributed simulation with multi-agent system. *International Journal of Production Research*, 52(22), pp.6819–6840.
- Davis, F., 1989. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13(3), pp.319–340.
- Deleris, L.A., Elkins, D. & Pate-Cornell, E., 2004. Analyzing Losses From Hazard Exposure: A Conservative Probabilistic Estimate Using Supply Chain Risk Simulation. *Proceedings of the 2004 Winter Simulation Conference, 2004.*, 2, pp.323–330.
- Dong, J. et al., 2006. IBM SmartScor - A SCOR Based Supply Chain Transformation Platform Through Simulation And Optimization Techniques. In *Proceedings of the 2006 Winter Simulation Conference*.
- Ellis, S.C., Henry, R.M. & Shockley, J., 2010. Buyer perceptions of supply disruption risk: A behavioral view and empirical assessment. *Journal of Operations Management*, 28(1), pp.34–46.
- Ellram, L.M. & Cooper, M.C., 2014. Supply chain management: It's all about the journey, not the destination.

---

*Journal of Supply Chain Management*, 50(1), pp.8–20.

- Enyinda, C.I., Ogbuehi, A. & Briggs, C., 2008. Global Supply Chain Risks Management: a New Battleground for Gaining Competitive Advantage. *Proceedings of ASBBS*, 15(1), pp.278–292.
- Epstein, J.M. & Axtell, R., 1996. *Growing artificial societies: social science from the bottom up*, Brookings Institution Press.
- Evers, P.T. & Wan, X., 2012. Systems analysis using simulation. *Journal of Business Logistics*, 33(2), pp.80–89.
- Fahimnia, B. et al., 2015. Quantitative models for managing supply chain risks: A review. *European Journal of Operational Research*, 247(1), pp.1–15.
- Forrester, J.W., 1968. Industrial Dynamics - After the First Decade. *Management Science*, 14(7), pp.398–415.
- Gaudenzi, B. & Borghesi, A., 2006. Managing risks in the supply chain using the AHP method. *International Journal of Logistics Management*, 17(1), pp.114–136.
- Ge, Y. et al., 2004. System dynamics modelling for supply-chain management: A case study on a supermarket chain in the UK. *International Transactions in Operational Research*, 11(5), pp.495–509.
- Gensym, 2008. Gensym.
- George, K., Nagali, V. & Rassey, L., 2012. *Expect the unexpected : Reduce corporate exposure and create value through supply chain risk management*,
- Ghadge, A. et al., 2013. A systems approach for modelling supply chain risks. *Supply chain management: an international journal*, 18(5), pp.523–538.
- Giannakis, M. & Louis, M., 2011. A multi-agent based framework for supply chain risk management. *Journal of Purchasing and Supply Management*, 17(1), pp.23–31.
- Guertler, B. & Spinler, S., 2015. When does operational risk cause supply chain enterprises to tip? A simulation of intra-organizational dynamics. *Omega (United Kingdom)*, 57, pp.54–69.
- Hall, D.C. & Saygin, C., 2012. Impact of information sharing on supply chain performance. *International Journal of Advanced Manufacturing Technology*, 58(1–4), pp.397–409.
- Hallikas, J. et al., 2004. Risk management processes in supplier networks. *International Journal of Production Economics*, 90(1), pp.47–58.
- Hallikas, J., Virolainen, V.M. & Tuominen, M., 2002. Risk analysis and assessment in network environments: A dyadic case study. *International Journal of Production Economics*, 78(1, SI), pp.45–55.
- Harland, C., Brenchley, R. & Walker, H., 2003. Risk in supply networks. *Journal of Purchasing and Supply Management*, 9(2), pp.51–62.
- Hasani, A. & Khosrojerdi, A., 2016. Robust global supply chain network design under disruption and uncertainty considering resilience strategies: A parallel memetic algorithm for a real-life case study. *Transportation Research Part E: Logistics and Transportation Review*, 87, pp.20–52.
- Heath, S.K. et al., 2011. Cross-paradigm simulation modeling: challenges and successes. *Proceedings of the Winter Simulation Conference*, pp.2783–2797.
- Heckmann, I., Comes, T. & Nickel, S., 2015. A critical review on supply chain risk - Definition, measure and modeling. *Omega (United Kingdom)*, 52, pp.119–132.
- Hendricks, K.B. & Singhal, V.R., 2005. An Empirical Analysis of the Effect of Supply Chain Disruptions on Long-Run Stock Price Performance and Equity Risk of the Firm. *Production and Operations Management*, 14(1), pp.35–52.
- Hines, P. & Rich, N.L., 1997. The Seven Value Stream Mapping Tools. *International Journal of Operations & Production Management*, 17(1), pp.46–64.
- Hishamuddin, H., Sarker, R.A. & Essam, D., 2012. A disruption recovery model for a single stage production-

- 
- inventory system. *European Journal of Operational Research*, 222(3), pp.464–473.
- Ho, W. et al., 2015. Supply chain risk management: a literature review. *International Journal of Production Research*, 53(16), pp.5031–5069.
- Hult, G.T.M. & Craighead, C.W., 2010. Risk Uncertainty and Supply Chain Decisions : A Real Options Perspective. *Decision Sciences Journal*, 41(3), pp.435–459.
- Ivanov, D., Sokolov, B., & Dolgui, A., 2014. The Ripple effect in supply chains: Trade-off “efficiency-flexibility- resilience” in disruption management. *International Journal of Production Research*, 52(7), pp.2154–2172.
- Jahangirian, M. et al., 2010. Simulation in manufacturing and business: A review. *European Journal of Operational Research*, 203(1), pp.1–13.
- Jain, S. et al., 2001. Development of a high-level supply chain simulation model. In *The 2001 Winter Simulation Conference B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, eds.* pp. 1129–1137.
- Janamanchi, B., & Burns, J.R., 2007. Reducing bullwhip oscillation in a supply chain: A system dynamics model-based study. *International Journal of Information Systems and Change Management*, 2(4), pp.350–371.
- Jüttner, U., 2005. Supply chain risk management Understanding the business requirements from a practitioner perspective. *The International Journal of Logistics Management*, 16(1), pp.120–141.
- Jüttner, U., Peck, H. & Christopher, M., 2003. An agenda for future research Supply chain risk management : Outlining an agenda for future research. *International Journal of Logistics Research and Applications: A Leading Journal of Supply Chain Management*, 6(4), pp.37–41.
- Kasi, V., 2005. Systemic Assessment of SCOR for Modeling Supply Chains. *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 0(C), pp.1–10.
- Kayis, B. & Dana Karningsih, P., 2012. Scris. *Journal of Manufacturing Technology Management*, 23(7), pp.834–852.
- Keeney, R.L. & McDaniels, T.L., 1992. Value-Focused Thinking about Strategic Decisions at BC Hydro. *Interfaces*, 22(6), pp.94–109.
- Kersten, W., Hohrath, P. & Böger, M., 2007. An empirical approach to supply chain risk management: development of a strategic framework. In *Proceedings POMS 2007 Conference*.
- Kim, J. & Rogers, K.J., 2005. An object-oriented approach for building a flexible supply chain model. *International Journal of Physical Distribution and Logistics Management*, 35(7), pp.481–502.
- Kitagawa, T. et al., 2000. A Description Language based on Multi Functional Modeling and a Supply Chain Simulation Toolg. In *2000 IEEE*.
- Kleijnen, J.J.P.C., 2005. Supply chain simulation tools and techniques: a survey. *International Journal of Simulation and Process ...*, (2004), pp.82–89.
- Kleijnen, J.P.C. & Smits, M.T., 2003. Performance metrics in supply chain management. *Journal of the Operational Research Society*, 54(5), pp.507–514.
- Kleindorfer, P.R. & Germaine, H.S., 2013. Managing Disruption Risks in Global Supply Chains. *Production and Operations Management*, 14(1), pp.53–68.
- Klimov, R. & Merkurjev, Y., 2008. Simulation model for supply chain reliability evaluation. *Technological and Economic Development of Economy*, 14(3), pp.300–311.
- Labarthe, O. et al., 2007. Toward a methodological framework for agent-based modelling and simulation of supply chains in a mass customization context. *Simulation Modelling Practice and Theory*, 15(2), pp.113–136.
- Langroodi, R.R.P. & Amiri, M., 2016. A system dynamics modeling approach for a multi-level, multi-product, multi-region supply chain under demand uncertainty. *Expert Systems with Applications*, 51(2016), pp.231–

- Lavastre, O., Gunasekaran, A. & Spalanzani, A., 2012. Supply chain risk management in French companies. *Decision Support Systems*, 52(4), pp.828–838.
- Law, A.M. & Law, A.M., 2008. Proceedings of the 2008 Winter Simulation Conference S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds. , pp.39–47.
- Leitch, M., 2010. ISO 31000:2009 - The new international standard on risk management: Perspective. *Risk Analysis*, 30(6), pp.887–892.
- Li, C., 2013. Controlling the bullwhip effect in a supply chain system with constrained information flows. *Applied Mathematical Modelling*, 37(4), pp.1897–1909.
- Lockamy III, A., 2014. Assessing disaster risks in supply chains. *Industrial Management & Data Systems*, 114(5), pp.755–777.
- Long, Q., 2014. Distributed supply chain network modelling and simulation: integration of agent-based distributed simulation and improved SCOR model. *International Journal of Production Research*, 52(23), pp.6899–6917.
- Long, Q. & Zhang, W., 2014. An integrated framework for agent based inventory-production-transportation modeling and distributed simulation of supply chains. *Information Sciences*, 277(March), pp.567–581.
- Longo, F. & Mirabelli, G., 2008. An advanced supply chain management tool based on modeling and simulation. *Computers and Industrial Engineering*, 54(3), pp.570–588.
- Lundin, J.F., 2012. Redesigning a closed-loop supply chain exposed to risks. *International Journal of Production Economics*, 140(2), pp.596–603.
- M. Law, A. & Kelton, W.D., 2000. *Simulation Modelling and Analysis* 3rd ed., McGraw-Hill Book Company.
- Manuj, I. et al., 2008. Global supply chain risk management strategies. *International Journal of Physical Distribution & Logistics Management*, 38(3), pp.192–223.
- Manuj, I., Esper, T.L. & Stank, T.P., 2014. Supply Chain Risk Management Approaches Under Different Conditions of Risk. *Journal of Business Logistics*, 35(3), pp.241–258.
- March, J.G. & Shapira, Z., 1987. Managerial Perspectives on Risk and Risk Taking. *Management Science*, 33(11), pp.1404–1418.
- Marchese, K. & Paramasivam, S., 2013. *The Ripple Effect How manufacturing and retail executives view the growing challenge of supply chain risk*, 2013 Deloitte Development LLC.
- Matinrad, N., Roghanian, E. & Rasi, Z., 2013. Supply chain network optimization: A review of classification, models, solution techniques and future research. *Uncertain Supply Chain Management*, 1, pp.1–24.
- Miller, H.E. c & Engemann, K.J.. d, 2008. A Monte Carlo simulation model of supply chain risk due to natural disasters. *International Journal of Technology, Policy and Management*, 8(4), pp.460–480.
- Min, H. & Zhou, G., 2002. Supply chain modeling: past, present and future. *Computers & Industrial Engineering*, 43(1–2), pp.231–249.
- Minar, N. et al., 1996. The Swarm Simulation System : A Toolkit for Building Multi-agent Simulations. *Simulation*, pp.1–11.
- Mohammadi, M., Mukhtar, M.B. & Peikari, H.R., 2011. A Grammar-Based Process Modeling and Simulation Methodology for Supply Chain Management. *Lecture Notes in Computer Science*, 7066, pp.77–86.
- Neiger, D., Rotaru, K. & Churilov, L., 2009. Supply chain risk identification with value-focused process engineering. *Journal of Operations Management*, 27(2), pp.154–168.
- Norrman, A. & Jansson, U., 2004. Ericsson's proactive supply chain risk management approach after a serious sub-supplier accident. *International Journal of Physical Distribution & Logistics Management*, 34, pp.434–456.



- 
- Oehmen, J. et al., 2009. System-oriented supply chain risk management. *Production Planning and Control*, 20(4), pp.343–361.
- Oyarbide, A. et al., 2003. Manufacturing systems modelling using system dynamics: forming a dedicated modelling tool. *Journal of Advanced Manufacturing Systems*, 21(1), pp.71–87.
- Peng, M., Peng, Y. & Chen, H., 2014. Post-seismic supply chain risk management: A system dynamics disruption analysis approach for inventory and logistics planning. *Computers and Operations Research*, 42, pp.14–24.
- Persson, F., 2011. SCOR template—A simulation based dynamic supply chain analysis tool. *International Journal of Production Economics*, 131(1), pp.288–294.
- Persson, F. et al., 2012. Supply chain dynamics in the SCOR model- A simulation modeling approach. In *Proceedings of the 2012 Winter Simulation Conference*. pp. 3821–3832.
- Petrovic, D., 2001. Simulation of supply chain behaviour and performance in an uncertain environment. *International Journal of Production Economics*, 71(1–3), pp.429–438.
- Pfohl, H.-C., Gallus, P. & Thomas, D., 2013. Interpretive structural modeling of supply chain risks. *International Journal of Physical Distribution & Logistics Management*, 41(9), pp.839–859.
- Pirard, F., Iassinovski, S. & Riane, F., 2008. A generic scalable simulation model for strategic supply chain management with emphasis on production activities. *International Journal of Computer Integrated Manufacturing*, 21(4), pp.455–467.
- Pirard, F., Iassinovski, S. & Riane, F., 2011. A simulation based approach for supply network control. *International Journal of Production Research*, 49(24), pp.7205–7226.
- Pundoor, G. & Herrmann, J., 2006. A hierarchical approach to supply chain simulation modelling using the Supply Chain Operations Reference model. *International Journal of Simulation and Process Modelling*, 2(3–4), pp.124–132.
- Robinson, S., 2004. *Simulation: The Practice of Model Development and Use*, Chichester, UK: Wiley.
- Saleh, M. et al., 2010. A comprehensive analytical approach for policy analysis of system dynamics models. *European Journal of Operational Research*, 203(3), pp.673–683.
- Saleh Ebrahimi, D., David, P. & Alpan, G., 2012. A Model Based Specification For A decision Support Tool For Supply Chain Risk Management. In *CIE42 Proceedings*. Cape Town, South Africa.
- Sawik, T., 2015. On the fair optimization of cost and customer service level in a supply chain under disruption risks. *Omega (United Kingdom)*, 53(November), pp.58–66.
- Scheer, A.W. & Nüttgens, M., 2000. ARIS architecture and reference models for business process management. In *Business Process Management*. pp. 301–304.
- Schmitt, A.J. & Singh, M., 2012. A quantitative analysis of disruption risk in a multi-echelon supply chain. *International Journal of Production Economics*, 139(1), pp.22–32.
- Schmitt, a. J. & Singh, M., 2009. Quantifying supply chain disruption risk using Monte Carlo and discrete-event simulation. *Winter Simulation Conference (WSC), Proceedings of the 2009*, pp.1237–1248.
- Seck, M., Rabadi, G. & Koestler, C., 2015. A Simulation-based Approach to Risk Assessment and Mitigation in Supply Chain Networks. *Procedia Computer Science*, 61, pp.98–104.
- Seila, A.F., 2004. 2004: Spreadsheet Simulation. In *2004 Winter Simulation Conference R. G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds*. pp. 41–48.
- Sellitto, M.A. et al., 2015. A SCOR-based model for supply chain performance measurement: application in the footwear industry. *International Journal of Production Research*, 53(16), pp.4917–4926.
- Seuring, S., 2013. A review of modeling approaches for sustainable supply chain management. *Decision Support Systems*, 54(4), pp.1513–1520.



- 
- Shafer, S.M. & Smunt, T.L., 2004. Empirical simulation studies in operations management: Context, trends, and research opportunities. *Journal of Operations Management*, 22(4 SPEC. ISS.), pp.345–354.
- Shao, X.F., 2012. Demand-side reactive strategies for supply disruptions in a multiple-product system. *International Journal of Production Economics*, 136(1), pp.241–252.
- Sheffi, Y. & Rice Jr., J.B., 2005. A Supply Chain View of the Resilient Enterprise. *MIT Sloan Management Review*, 47(1), pp.41–48.
- Simchi-levi, D. et al., 2015. Identifying Risks and Mitigating Disruptions in the Automotive Supply Chain. *Interfaces*, 45(No. 5, September–October 2015), pp.375–390.
- Simchi Levi, D., Vassiladis, C. & Kyrtzoglou, I., 2013. Making the right risk decisions to strengthen operations performance. In *PwC and the MIT Forum for Supply Chain Innovation*.
- Snyder, L. V et al., 2016. OR/MS models for supply chain disruptions: a review. *A Review, IIE Transactions*, 8830(December).
- Sprock, T. & McGinnis, L.F., 2014. Towards automated access to supply chain analyses. In *Proceedings of the 2014 POMS Annual Conference*.
- Steele, P. & Brian H. Court, 1996. *Profitable purchasing strategies: a manager's guide for improving organizational competitiveness through the skills of purchasing*, McGraw-Hill Book Company.
- Sterman, J.D., 2000. *Business Dynamics: Systems Thinking and Modeling for a Complex World* IL: Irwin, ed., Chicago: McGraw Hill.
- Sumari, S. & Ibrahim, R., 2013. Comparing Three Simulation Model Using Taxonomy: System Dynamic Simulation, Discrete Event Simulation and Agent Based Simulation. *International Journal of Management Excellence*, 1(3), pp.4–9.
- Tako, A.A. & Robinson, S., 2012. The application of discrete event simulation and system dynamics in the logistics and supply chain context. *Decision Support Systems*, 52(4), pp.802–815.
- Talluri, S. (Sri) et al., 2013. Assessing the Efficiency of Risk Mitigation Strategies in Supply Chains. *Journal of Business Logistics*, 34(4), pp.253–269.
- Tang, C.S., 2006. Perspectives in supply chain risk management. *International Journal of Production Economics*, 103(2), pp.451–488.
- Tang, O. & Nurmaya Musa, S., 2011. Identifying risk issues and research advancements in supply chain risk management. *International Journal of Production Economics*, 133(1), pp.25–34.
- Terzi, S. & Cavalieri, S., 2004. Simulation in the supply chain context: A survey. *Computers in Industry*, 53(1), pp.3–16.
- Thun, J.-H. & Hoenig, D., 2011. An empirical analysis of supply chain risk management in the German automotive industry. *International Journal of Production Economics*, 131(1), pp.242–249.
- Tomlin, B., 2006. On the Value of Mitigation and Contingency Strategies for Managing Supply Chain Disruption Risks. *Management Science*, 52(5), pp.639–657.
- Trkman, P. & McCormack, K., 2009. Supply chain risk in turbulent environments-A conceptual model for managing supply chain network risk. *International Journal of Production Economics*, 119(2), pp.247–258.
- Tuncel, G. & Alpan, G., 2010. Risk assessment and management for supply chain networks: A case study. *Computers in Industry*, 61(3), pp.250–259.
- U.S. Resilience Project, 2011. De-Risking the Supply Chain : Cisco 's Risk Intelligence and Analytic Tools. , (August 8), pp.1–6.
- Udenio, M., Fransoo, J.C. & Peels, R., 2015. Destocking, the bullwhip effect, and the credit crisis: Empirical modeling of supply chain dynamics. *International Journal of Production Economics*, 160, pp.34–46.
- Umeda, S. & Zhang, F., 2008. Hybrid Modeling Approach for Supply- Chain Simulation. In *Lean Business*

---

*Systems and Beyond*, Tomasz Koch. Boston: Springer, p. 8.

Vilko, J.P.P. & Hallikas, J.M., 2012. Risk assessment in multimodal supply chains. *International Journal of Production Economics*, 140(2), pp.586–595.

Wagner, S.M. & Bode, C., 2006. An empirical investigation into supply chain vulnerability. *Journal of Purchasing and Supply Management*, 12(6 SPEC. ISS.), pp.301–312.

Wan, X., Pekny, J.F. & Reklaitis, G. V., 2005. Simulation-based optimization with surrogate models - Application to supply chain management. *Computers and Chemical Engineering*, 29(6 SPEC. ISS.), pp.1317–1328.

Wu, D.D. & Olson, D., 2010. Enterprise risk management: a DEA VaR approach in vendor selection. *International Journal of Production Research*, 48(16), pp.4919–4932.

Wu, T. et al., 2013. Supply Chain Risk Management: An Agent-Based Simulation to Study the Impact of Retail Stockouts. *Ieee Transactions on Engineering Management*, 60(4), pp.676–686.

Wu, T., Blackhurst, J. & Chidambaram, V., 2006. A model for inbound supply risk analysis. *Computers in Industry*, 57(4), pp.350–365.

Yates, J., 1992. *Risk Taking Behaviour* Wiley, ed., Chichester, UK.

Van Der Zee, D. & Van Der Vorst, J.G.A., 2005. A Modeling Framework for Supply Chain Simulation : Opportunities for Improved Decision Making \*. *Decision Sciences*, 36(1), pp.65–95.

Zhang, K. et al., 2011. Pre-warning analysis and application in traceability systems for food production supply chains. *Expert Systems with Applications*, 38(3), pp.2500–2507.

Zsidisin, G.A., 2003. A grounded definition of supply risk. *Journal of Purchasing & Supply Management*, 9, pp.217–224.

Zsidisin, G. a. et al., 2008. Supply risk perceptions and practices: an exploratory comparison of German and US supply management professionals. *International Journal of Technology, Policy and Management*, 8(4), p.401.

---

# ANNEXES

# A1 THE LIBRARY OF DOMAIN SPECIFIC OPERATIONS

## A1.1 THE ISSUEMATERIAL OPERATION (sMi.2)

### Definition

The operation is responsible for transferring the products (components or raw material) from one location to the production location and requesting to replenish the components stock when it's required.

### Inputs and Outputs from the SCOR model

This ISSUEMATERIAL Operation (sMi.2) is defined based on the SCOR Process element “sM1.2/sM2.2 Issue Material”. The SCOR model specifies the inventory availability and the production schedule as inputs to the “sM1.2/sM2.2 Issue Material” Process element and specifies the information feedback, the inventory availability, the replenishment signal and the workflow as its outputs. We retain the ProductionOrder as an input for the ISSUEMATERIAL Operation (sMi.2). Furthermore, we consider that the production schedule contains the information about the TransferResource to be used, the input Buffer, the output Buffer and the Path. For the outputs, the modified ProductionOrder is used instead of the workflow to trigger the next operation; the ReplenishmentSignal is used to request for components’ replenishment when the *inventoryLevel* reaches a given level. The inventory availability is not retained as input or output for the ISSUEMATERIAL Operation since its not used to trigger current or next Operation. Furthermore, the availability of products is checked directly through accessing the concerned Buffer, so there is no need to receive or to send information about the available Products. Table A1.1 summarizes the inputs and the outputs we have retained for the ISSUEMATERIAL Operation from the SCOR reference model and the variable names that will be used to represent them in the model. Figure A1.1 describes the relation between blocks of the variables and the ISSUEMATERIAL Operation. Figure A1.2 provides a detailed description of them.

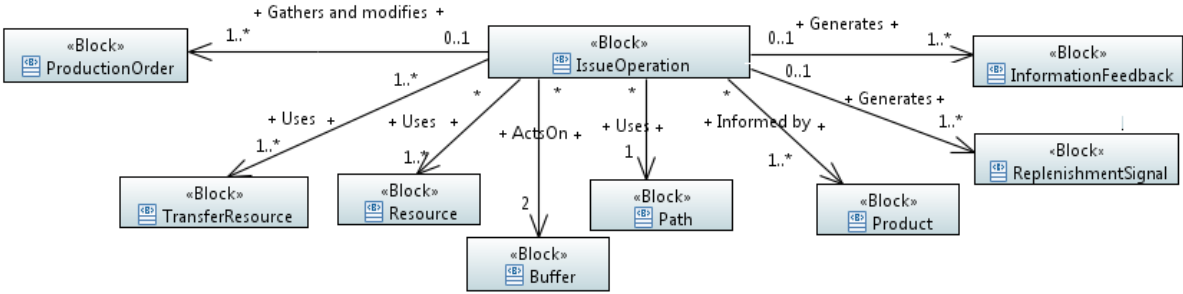


FIGURE A1.1: THE ISSUEMATERIAL OPERATION BLOCK DEFINITION DIAGRAM

TABLE A1.1: RETAINED INPUTS AND OUTPUTS FOR THE ISSUE MATERIAL OPERATION FROM THE SCOR MODEL

<b>SCOR inputs and outputs</b>	<b>Retained Inputs and outputs:</b>	<b>Designation</b>
<b>SCOR inputs:</b>	<b>Inputs</b>	
Inventory availability Production Schedule	pO : ProductionOrder [1..*] {Ordered},	The received production orders that define the quantities to be issued.
	iB: Buffer	The input Buffer from where components have to be issued.
	oB: Buffer	The input Buffer to which products need to be transferred.
	tR: TransferResource	The transfer resource used to issue products.
	pT: Path	The path followed to transfer products from a Buffer to another.
<b>SCOR outputs:</b>	<b>Outputs</b>	
Information Feedback, Replenishment Signal, Workflow, Inventory Availability.	iFd: InformationFeedback [1..*] {Ordered},	The information feedback that informs about the state of execution.
	rO: ReplenishmentOrder [1..*] {Ordered},	The replenishment signal requests for components when their inventory level reaches a critical level.
	pO: ProductionOrder [1..*] {Ordered},	The production order with a modified status after issuing components.

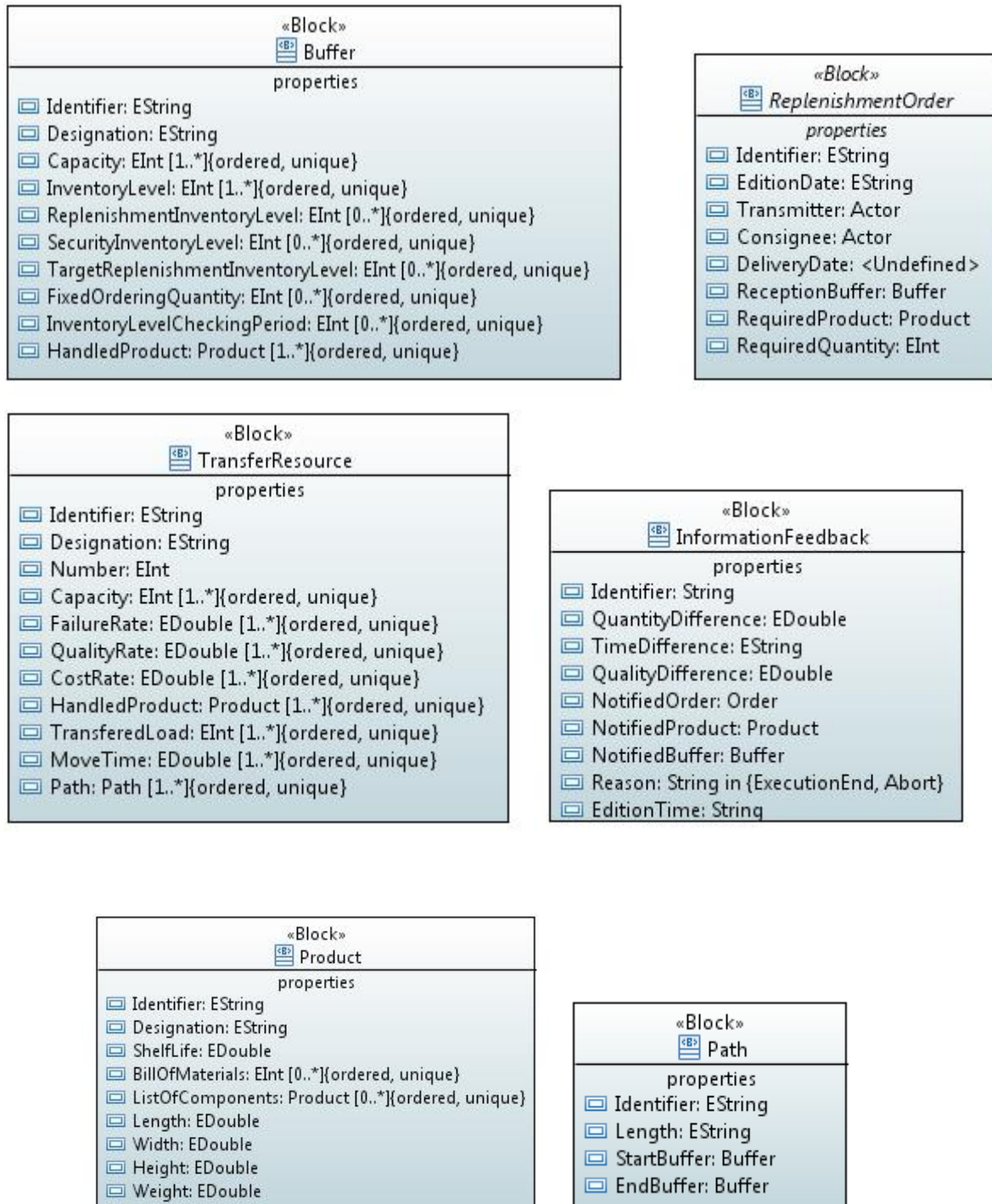


FIGURE A1.2: DETAILS OF THE USED INPUTS AND OUTPUTS FOR THE ISSUEMATERIAL OPERATION

### Assumptions

For the ISSUEMATERIAL Operation we assume the following:

- The Operation handles a set of ProductionOrders
- The ProductionOrders are treated by one, following first in first out rule.
- The issue of Products is executed only if the required inventory is available in the input Buffer.

### The operation algorithm

The most common standard mode of the ISSUEMATERIAL Operation is as follow. First, we start by storing the set of received the production orders. A ProductionOrder is selected by following first in first out rule. The selected order is released only if the required quantities are available in the input Buffer. Using the bill of material and the quantity of the final Product that has to be produced, the algorithm determines the quantities of the components to be issued.

The TransferResource is allocated and moved to the input Buffer; it picks the Product units with respect to the transfer *capacity*. The TransferResource moves the Product units to the output Buffer and returns to the input Buffer position for the remaining quantities. After finishing the transfer execution, a Notification is sent to inform about the availability of Products in the output Buffer. The Operation algorithm is illustrated in the state machine of figure A1.3.

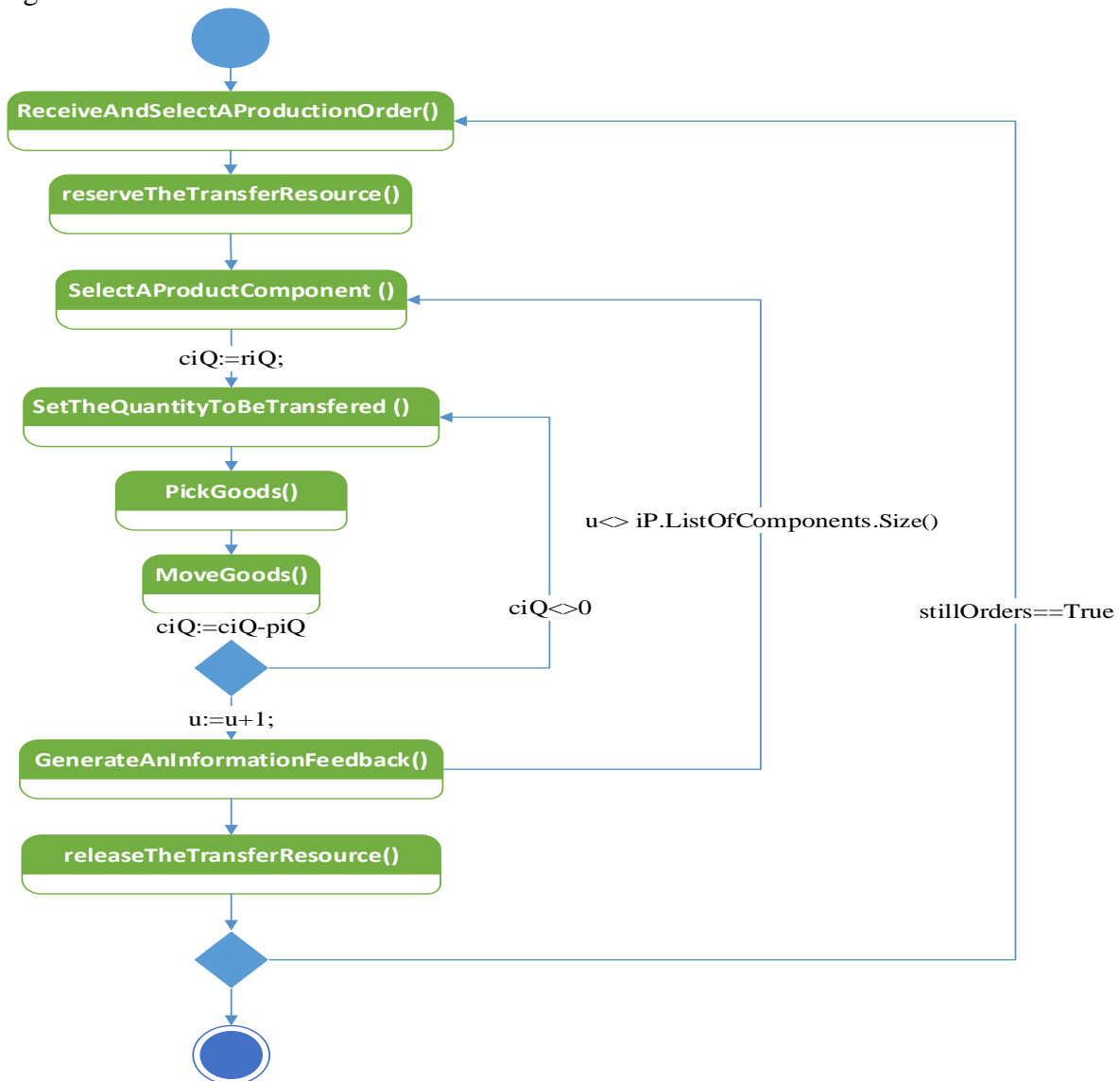


FIGURE A1.3: STATE MACHINE OF THE ISSUEMATERIAL OPERATION'S STANDARD MODE

### Algorithm internal variables



Aside from the variables already mentioned in the table, we need a set of internal variables used by the algorithm of the ISSUEMATERIAL Operation. In Table A1.2, we summarize those variables.

TABLE A1.2: THE INTERNAL VARIABLES OF THE ISSUEMATERIAL OPERATION ALGORITHM

Internal variables	Designations
ListOfNewOrders	The list of received orders.
ExecutionStatus	The execution status of the current treated order.
ReceptionOfANewOrder	A boolean variable which indicates if a new order is received.
ListOfInventoryAvailabilityNotifications	The list of inventory availability notifications.
ListOfReceiptVerificationNotifications	The list of receipt verification notifications.
currentOrderRank	the selected order rank in the list of received orders.
selectedOrderType	The type of the selected order.
pN	The number of Product types in the currently selected order.
tQ	the list of quantities to be transferred by Product type.
tP	The list of Product types to be transferred.
ctQ	remaining current quantity to be transferred;
ctP	current Product to be transferred;
rtQ	the remaining quantity to be transferred to the current Product type.
ptQ	the Quantity portion of the current Product to be transferred that respects the transfer resource capacity.

## Methods

In the following tables, we provide the pseudo codes of the procedures (methods) that are used in the algorithm of the ISSUEMATERIAL Operation. They are as follows:

<b>Method 1:</b> Public void ReceiveAndSelectAProductionOrder()
<b>Description</b> The method is responsible for receiving production orders saving them in a list and selecting the order to be treated when the conditions of its treatment become true.
<b>Algorithm</b> <b>Method 1:</b> Public void ReceiveAndSelectProductionOrder() { / receiveProductionOrdersAndSaveThemInAList/ If (ReceptionOfANewProductionOrder==true) { ListOfProductionOrders.Add(ProductionOrder); ExecutionStatus[ListOfProductionOrders.size()]:= “waiting”; }EndIF Do { num:=0; For k from 1 to ListOfProductionOrders.size() { If (ExecutionStatus[k]== “waiting”) { num:=num+1; if (num==1){

```

    currentOrderRank:=k;
    pO:= ListOfProductionOrders[k];
    iQ:= pO.requiredQuantity;
    iP:= pO.requiredProduct;
    ExecutionStatus[k]:= “Released” ;
    stillOrders:=true;
    }EndIf
    Else { stillOrders:=false;}
    }EndIf
    }EndFor
    wait ;
    } while (ExecutionStatus[k] != “Issued”)
    EndWhile}

```

<b>Method 2: Public void reserveTheTransferResource ()</b>
<b>Description</b> This method is responsible for reserving the transfer resource to be used to issue the Product components from the input Buffer to the output Buffer. If the resource is not available, the method waits until the transfer resource becomes available.
<b>Algorithm</b> <b>Method 2: Public void reserveTheTransferResource () {</b> While (tR.available== false) {wait;} tR.available:= false; }

<b>Method 3: Public void SelectAProductComponent ()</b>
<b>Description</b> This method is responsible for selecting a Product component to be treated.
<b>Algorithm</b> <b>Method 3: Public void SelectAProductComponent () {</b> ciC:= iP.ListOfComponents[u]; ciQ:= iQ × iP.BillOfMaterial[u]; }

<b>Method 4: Public void SetTheQuantityToBeTransferred ()</b>
<b>Description</b> This method is responsible for cutting the required quantity into smaller quantities that respect the available capacity of the transfer resource.
<b>Algorithm</b> <b>Method 4: Public void SetTheQuantityToBeTransferred (){</b> If (riQ = tR.capacity[ciP] × tR.Number[ciP] ) { piQ:= riQ;} Else {piQ:= tR.capacity[ciP] × tR.Number[ciP] ;} EndIf }

<b>Method 5: Public Void PickGoods ()</b>
<b>Description</b> This method picks the goods to be transferred from the input Buffer (iB).
<b>Algorithm</b> <b>Method 5: Public Void PickGoods() {</b>

```
iB.inventoryLevel(ciP):=iB.inventoryLevel(ciP)- piQ;  
tR.transferredLoad(ciP):= tR.transferredLoad(ciP)+ piQ; }
```

**Method 6: Public Void moveGoods ()**

**Description**

This method adjusts the inventory level of the output Buffer by adding the unloaded quantity to the current inventoryLevel and modifies the load size (tR.transferredLoad) of the transfer resource.

**Algorithm**

```
Method 6: Public Void moveGoods () {  
simulation.currentTime:= simulation.currentTime+ MoveTime(Pt) ;  
oB.inventoryLevel(ciP):=oB.inventoryLevel(ciP)+ tR.TransferredLoad(ciP);  
tR.transferredLoad(ciP):= tR.transferredLoad(ciP)- unloadedQuantity; }
```

**Method 7: public Void GenerateAnInformationFeedback ()**

**Description**

When all the required quantity for a given component is issued an information feedback is generated to inform about the new inventory state and the new state of the operation execution.

**Algorithm**

```
Method 7: public Void GenerateAnInformationFeedback () {  
/Notify about the inventory level change to concerned operation./  
Identifier:= IF.generatedId();  
NotifiedProduct:= ciC;  
NotifiedBuffer:= oB;  
NotifiedOrder:=pO;  
EditionTime:=CurrenTime;  
Reason:= "ExecutionEnd"; }
```

**Method 8: Public void releaseTheTransferResource ()**

**Description**

This method is responsible for releasing the transfer resource, after finishing treating the current order.

**Algorithm**

```
Method 8: Public void releaseTheTransferResource () {  
tR.available:= true; }
```

## A1.2 THE TEST OPERATION (sMi.3.2)

### Definition

The Operation is responsible for checking the respect of manufactured Products' quality requirements and separating the non-conforming Products from conforming ones.

### Inputs and outputs from the SCOR model

This Operation is defined based on the SCOR Process element "sM1.3/sM2.3 Produce and test". In our work, we divided it into two operations: The PRODUCE Operation (sMi.3.1) and the TEST Operation (sMi.3.2). The SCOR model specifies the workflow and the information feedback as inputs to this Process element, the waste produced and the workflow as outputs. We estimate that the information included in the workflow can be described by a

ProductionOrder. Also, we assume that the workflow includes the information about the input Buffer, the information about the output Buffer of good Products and the information about the output Buffer of defective Products. Furthermore, we estimate that the waste produced can be assimilated to a modification of the *inventoryLevel* of the output Buffer relative to defectives. Table A1.3 summarizes the inputs and the outputs we have retained for the TEST Operation from the SCOR model and the variable names that will be used to represent them in the model. Figure A1.4 describes the relation between the retained variables' blocks and the TEST Operation block. Figure A1.5 provides the elements of the Meta-model related to the inputs and outputs of the TEST Operation.

TABLE A1.3: RETAINED INPUTS AND OUTPUTS FROM THE SCOR MODEL FOR THE TEST OPERATION

SCOR inputs and outputs	Retained Inputs and outputs	Designations
<b>SCOR inputs</b>	<b>Inputs</b>	
Workflow.	pO : ProductionOrder [1..*] {Ordered},	It describes the details of what is requested to be manufactured.
	iB: Buffer	The input Buffer of the manufactured products to be tested
	dB: Buffer	The defective products Buffer
	gB:Buffer	The tested good products Buffer
<b>SCOR outputs</b>	<b>Outputs</b>	
Information feedback,  Workflow,  Waste produced,	iFd : Informationfeedback [1..*] {Ordered},	It notifies about the state of the tested products
	pO : ProductionOrder [1..*] {Ordered},	The production order with modified status to "Tested"

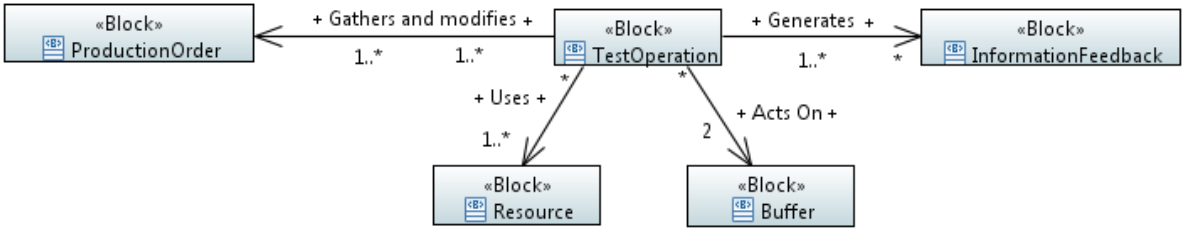


FIGURE A1.4: THE TEST OPERATION BLOCK DEFINITION DIAGRAM

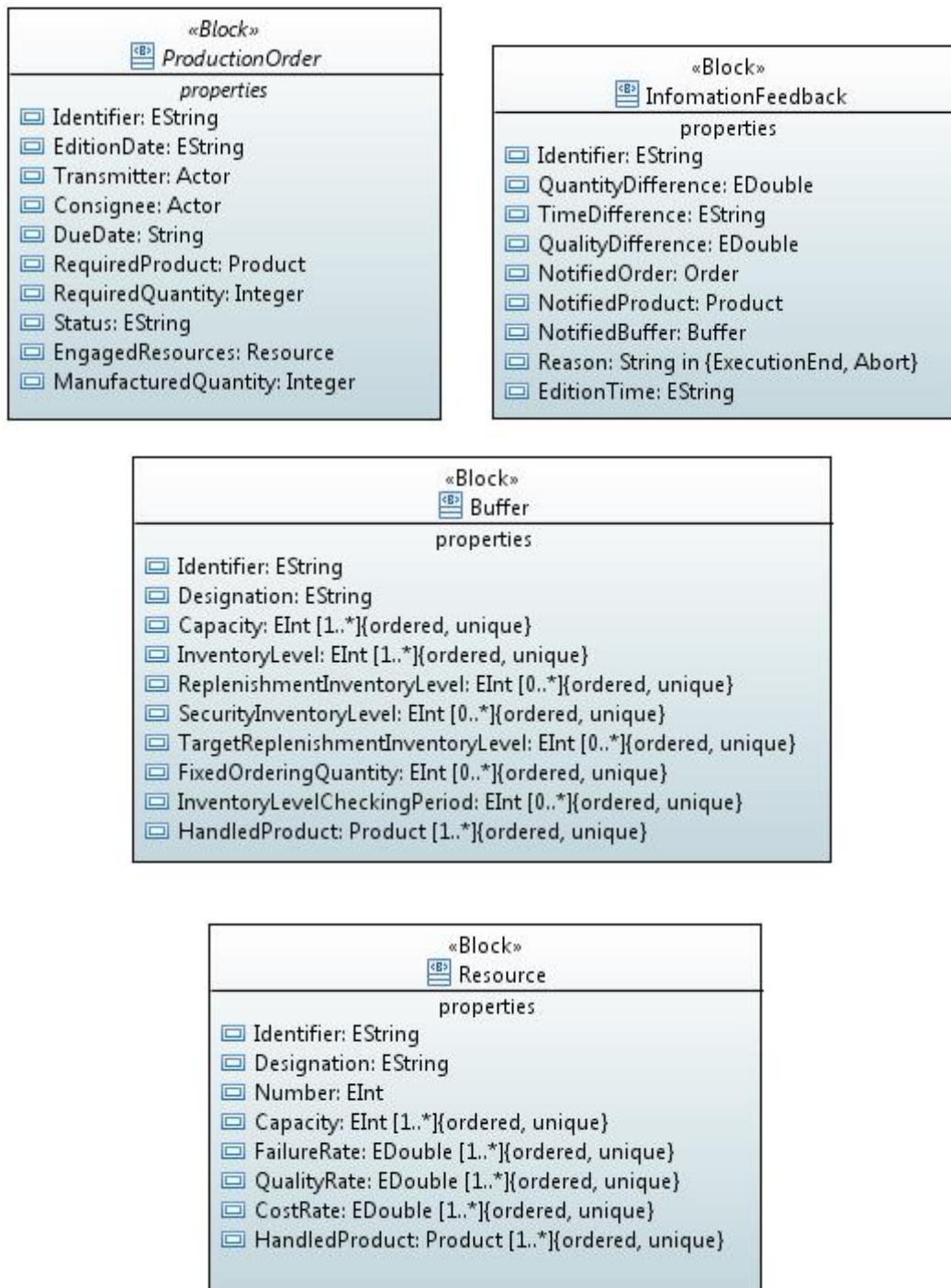


FIGURE A1.5: DETAILS OF THE USED INPUTS AND OUTPUTS FOR THE TEST OPERATION

## Assumptions

For the TEST Operation, we assume the following:

- The TEST Operation executes the quality check by ProductionOrder and by a lot. When the remaining manufactured quantity of a given ProductionOrder is less than the lot size (in the case of production abortion) only the available quantity is checked.
- The quality check is triggered when the quantity of Product in the input Buffer becomes equal to the test lot.

- Every Product has a test execution *cycle time* that depends on the used Resource.

### Operation algorithm

The algorithm of the main standard Operation Mode for the TEST Operation is as follows. The Operation checks the quality of the manufactured Products by a lot. When the quantity in the input Buffer becomes equals to the test lot size, the quality of the Products is checked. If the production is aborted and the manufactured quantity is less than the lot size, all the remaining manufactured Products are checked. The Product with a good quality and the defective Products are separated into two different Buffers. A time is spent to execute test Operation, this time equals to the test *cycle time*. An InformationFeedback is generated by the Operation to inform the scheduling Operation about the defective quantity. The TEST Operation algorithm is illustrated in the state machine of figure A1.6.

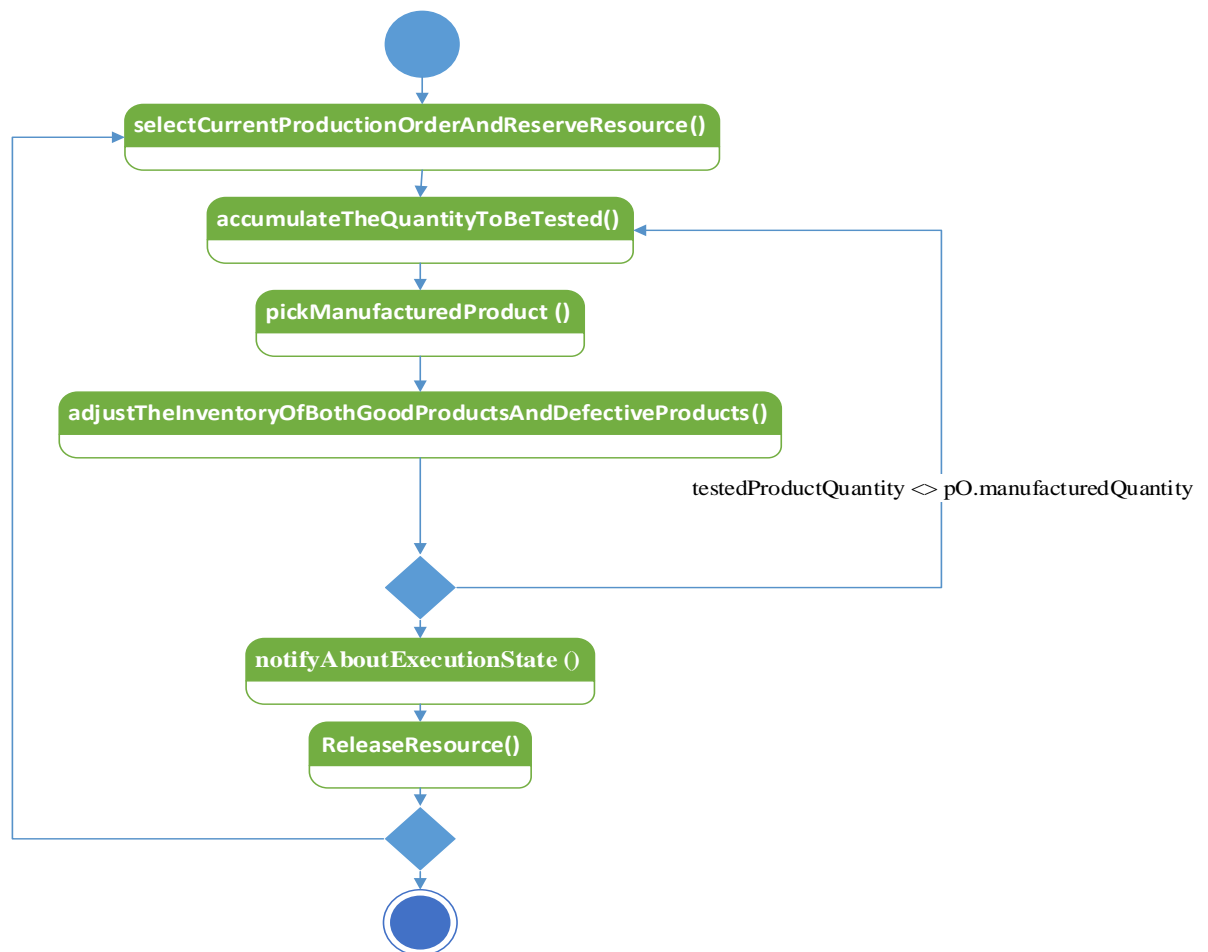


FIGURE A1.6: THE STATE MACHINE OF THE ALGORITHM OF THE STANDARD MODE OF THE TEST OPERATION

### Internal variables

Aside from the variable already mentioned in TableA1.3, we need some internal variables for the algorithm of the TEST Operation. In Table A1.4, we summarize those variables.

TABLE A1.4: INTERNAL VARIABLES USED IN THE ALGORITHM OF THE TEST OPERATION

Internal variables	Designations
Tl	The test lot size.
ResourcesAreAvailableAndAllocable	Boolean variable that takes “True” if the variable is available and allocable.
testedUnitsNumber	A counter of the received Product units for every production order.
ReceivedUnitsNumber	A counter of the received Product units for every lot.
teCT	The test resource cycle time.
pODefectiveQuantity	An integer variable that refers to the discovered quantity of the defective products for the current production order.

## Methods

In the following tables, we provide the pseudo codes of the procedures (methods) that are used in the algorithm of the TEST Operation (see figure A1.6).

<b>Method 1:</b> Public void selectCurrentProductionOrderAndReserveTestResource()
<p><b>Description</b></p> <p>This method determines if the received Product unit belongs to a new production order or not (currentPO pO.Identifier). If the production order is a new one, the monitored tested quantity, the monitored defective quantity is put to zero and the information about the current order to be tested is saved.</p>
<p><b>Algorithm</b></p> <pre> Public void selectCurrentProductionOrderAndReserveTestResource(){ Gather(); If (InventoryLevelChangeInTheInputBuffer==true) then {   If (currentPO pO.Identifier) then {     currentPO :=pO.Identifier;     testedUnitsNumber:=0;     pODefectiveQuantity:=0;     tP:=pO.requiredProduct;     rQ:=pO.requiredQuantity;     teCT:=resource.cycleTime(tP);     Do {       ResourcesAreAvailableAndAlocable := true;       If (tR.Available==false Or tR.Allocated==true ) then {         ResourcesAreAvailableAndAllocable := false; wait();} EndIF }       while (ResourcesAreAvailableAndAllocable == false )       If ( ResourcesAreAvailableAndAllocable == true ) then{ tR.Allocated:=true ;       } EndIF     }EndIF   }EndIf } </pre>



---

**Method 2:** Public void acumulateTheQuantityToBeTested()

**Description**

This method accumulates the Product units until getting the quantity to be tested. The quantity to be tested is usually equaled to the test lot size. There is a case where the quantity to be tested is different from the predefined test lot:  
The case where the remaining manufactured quantity to be tested is less than the test lot size: In this case, all the remaining manufactured quantity is tested.

**Algorithm**

```
public void acumulateTheQuantityToBeTested(){  
Count received products units.  
If (receivedUnitsNumber == tL or testedUnitsNumber == PO.manufacturedQuantity) then {  
quantityToBeTested= receivedUnitsNumber;  
receivedUnitsNumber=0; } }
```

**Method 3:** Public void pickManufacturedProduct ()

**Description**

This method picks the Product to be tested from the input Buffer when the received quantity becomes equals to the predetermined quantity to be tested. The method decreases the current inventory level with a quantity equals to the predetermined quantity to be tested (quantityToBeTested).

**Algorithm**

```
public void pickManufacturedProduct() {  
/Adjust the inventory of the input Buffer,  
tB.inventoryLevel(tP):= tB.inventoryLevel(tP)- quantityToBeTested; }
```

**Method 4:** Public void adjustTheInventoryOfBothGoodProductsAndDefectiveProducts()

**Description**

This method adjusts the inventory of both the Buffer defective products (dB) and the Buffer of good products (gB). The method advances first the simulation current time by the required time to test the batch size (teCT \* quantityToBeTested). The defective quantity of products is summed with the defective inventory level (dB.DefectiveInventoryLevel(tP)) of defective products. While the quantity of good products (quantityToBeTested –defectiveNumber) is summed with the inventory level (gB.inventoryLevel(tP)) of the good products.

**Algorithm**

```
Public void AdjustTheInventoryOfBothGoodProductsAndDefectiveProducts() {  
/Advance time by the testing duration. /  
currentTime:=currentTime+ teCT * quantityToBeTested;  
/ Determine the number of defective Product units of the current batch and the number of good  
products and adjust the inventory levels. /  
dB.InventoryLevel(tP):= dB.DefectiveInventoryLevel(tP)+DefectiveQuantity ;  
pODefectiveQuantity := pODefectiveQuantity + DefectiveQuantity;  
/Separate defective Product from the non-defective product. /  
gB.inventoryLevel(tP):= gB.inventoryLevel(tP)+ quantityToBeTested - DefectiveQuantity;  
testedUnitsNumber= testedUnitsNumber + quantityToBeTested; }
```

<b>Method 5 : Public void NotifyAboutExecutionState ()</b>
<p><b>Description</b> This method notifies modifies the status of the production order to “tested” and notifies about the state of the tested products.</p>
<p><b>Algorithm</b>  Public void NotifyAboutExecutionState () {  Generate();  iF.Identifier:= iF.GenerateId();  iF.NotifiedProduct=rP  iF.EditionTime:=currentTime;  iF.QualityDifference:= pODefectiveQuantity;  iF.NotifiedOrder=pO.identifier;  iF.NotifiedBuffer=DefectiveBuffer;  if (testedProductQuantity==pO.manufacturedQuantity) {  if (pO.Status=="Aborted") then  { pO.Status=="AbortedAndTested"; iF.Reason="ExecutionEnd"; }  }EndIF  if (pO.Status=="Executed") then  { pO.Status=="Tested"; iF.Reason="ExecutionEnd"; }  EndIF }</p>

<b>Method 6: Public void releaseResource ()</b>
<p><b>Description</b> This method is releasing the reserved resource for production.</p>
<p><b>Algorithm</b>  Public void releaseResource () {  / For every Units batch of the quantity to be manufactured do  tR.Available = true; }</p>

### A1.3 THE PICKANDPACK OPERATION (C.SDI.9-SDI.10)

#### Definition

The Operation is responsible of capturing the functionality of picking the Products from the stock of final Products and packing them. Picking is defined as the selection and the retrieval of Products while packing is defined as the grouping of Products into a pack to facilitate the transportation.

#### Inputs and outputs from the SCOR model

The operation is defined based on a combination of two Process elements. The first one is “sD1.9/sD2.9 pick product” and the second Process element is “sD1.10/ sD2.10 pack product”.

As shown in Table A1.5, SCOR specifies a set of inputs for the Process elements “sD1.9/sD2.9 pick product” and “sD1.10/sD2.10 pack product”. We think that the information communicated by the scheduled deliveries input and the workflow input can be captured using the proposed input ShippingOrder. The SCOR input inventory availability is

not required since the Operation is able to check the state of the Buffer via the *inventory level*. Furthermore, we assume that the workflow includes the information about the input Buffer, the information about the output Buffer and the information about the Resource to be used. SCOR defines two outputs: The information feedback and the workflow. We capture the information provided by the workflow through the ShippingOrder output. We keep the informationFeedback, the output of the SCOR Process elements as an output for the PICKANDPACK Operation.

Table A1.5 summarizes the inputs and the outputs we have retained for the PICKANDPACK Operation from the SCOR model and the variable names that will be used to represent them in the model. Figure A1.7 describes the relation between the retained variables' blocks and the PICKANDPACK Operation block. Figure A1.8 provides the elements of the Meta-model related to the inputs and outputs of the PICKANDPACK operation.

TABLE A1.5: RETAINED INPUTS AND OUTPUTS FROM THE SCOR MODEL FOR THE PICKANDPACK OPERATION

SCOR inputs and outputs	Retained inputs and outputs	Designations
<b>SCOR inputs:</b>	<b>Inputs:</b>	
<ul style="list-style-type: none"> <li>Scheduled deliveries</li> <li>Workflow</li> <li>Inventory availability</li> </ul>	sO:ShippingOrders [1..*] {Ordered},	It defines what is required to be shipped and what to be used for shipping.
	iB: Buffer	The input Buffer from products are picked
	oB: Buffer	The output Buffer where packed products are put
	ppR: Resource	The resource used for picking and packing products
<b>SCOR outputs:</b>	<b>Outputs :</b>	
<ul style="list-style-type: none"> <li>Information feedback</li> <li>Workflow</li> </ul>	iFd : Informationfeedback [1..*],	It provides a notification about the execution state of picking and packing.
	sO:ShippingOrders [1..*] {Ordered},	They are the shipping orders with a modified status.

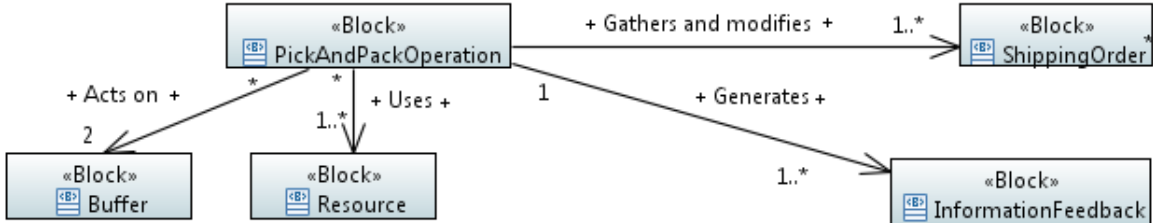


FIGURE A1.7: THE PICKANDPACK OPERATION BLOCK DEFINITION DIAGRAM

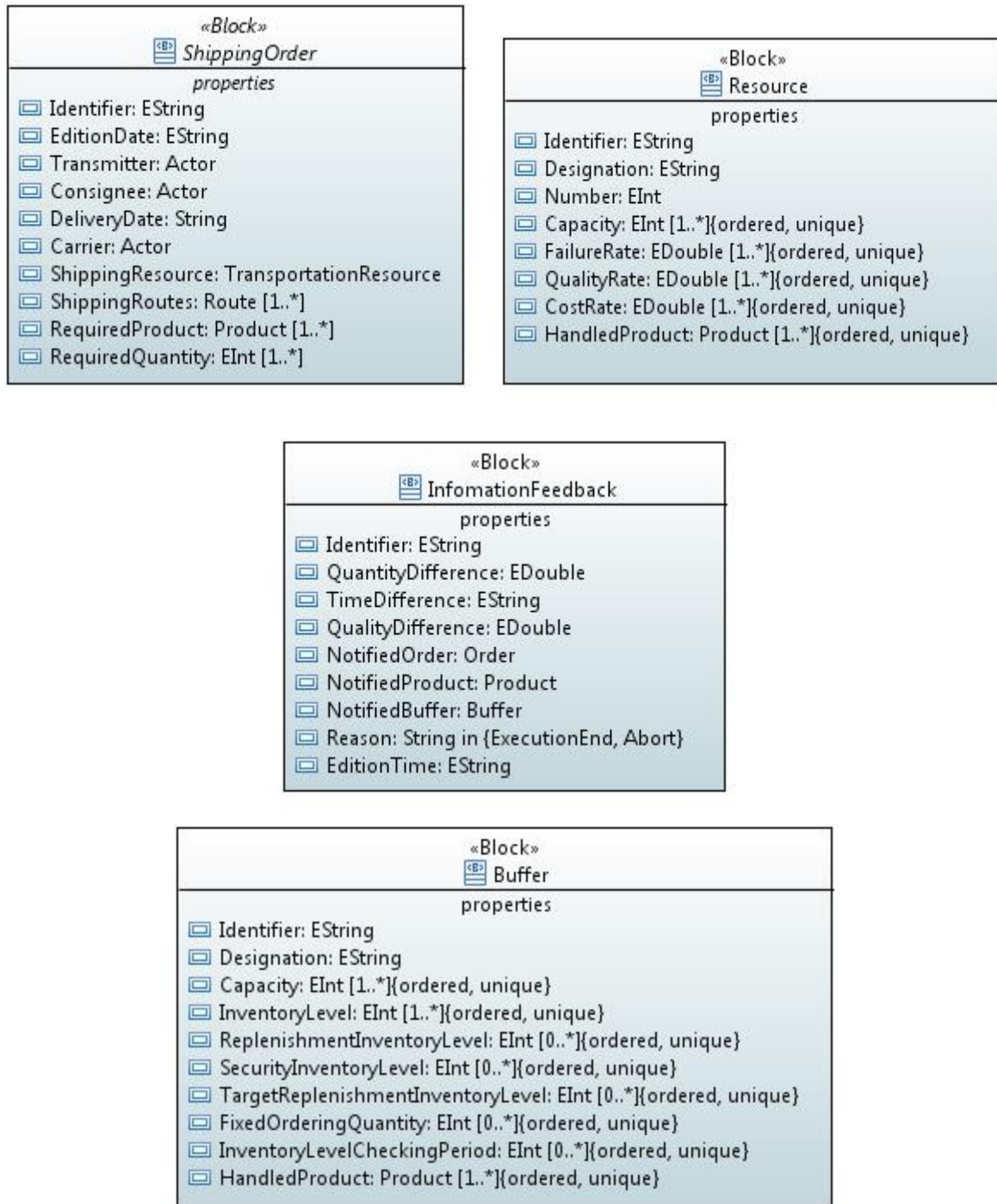


FIGURE A1.8: DETAILS OF THE USED INPUTS AND OUTPUTS FOR THE PICK AND PACK OPERATION

### Assumptions

The picking and packing of ShippingOrders are done by following the “first in first out rule”.

The products are shipped as they are sorted in the ShippingOrder.

The picking and packing are triggered only if all required quantity is available in the input Buffer.

## Operation algorithm

The algorithm of the most common standard Mode of the PICKANDPACK Operation is shown in the state chart of figure A1.9. For each product, the Operation acts on the input Buffer by reducing the *inventory level* and by increasing the *inventoryLevel* of the output Buffer. The movement of products respects the capacity of all the engaged Resources. The Operation algorithm is illustrated in the state machine of figure A1.9.

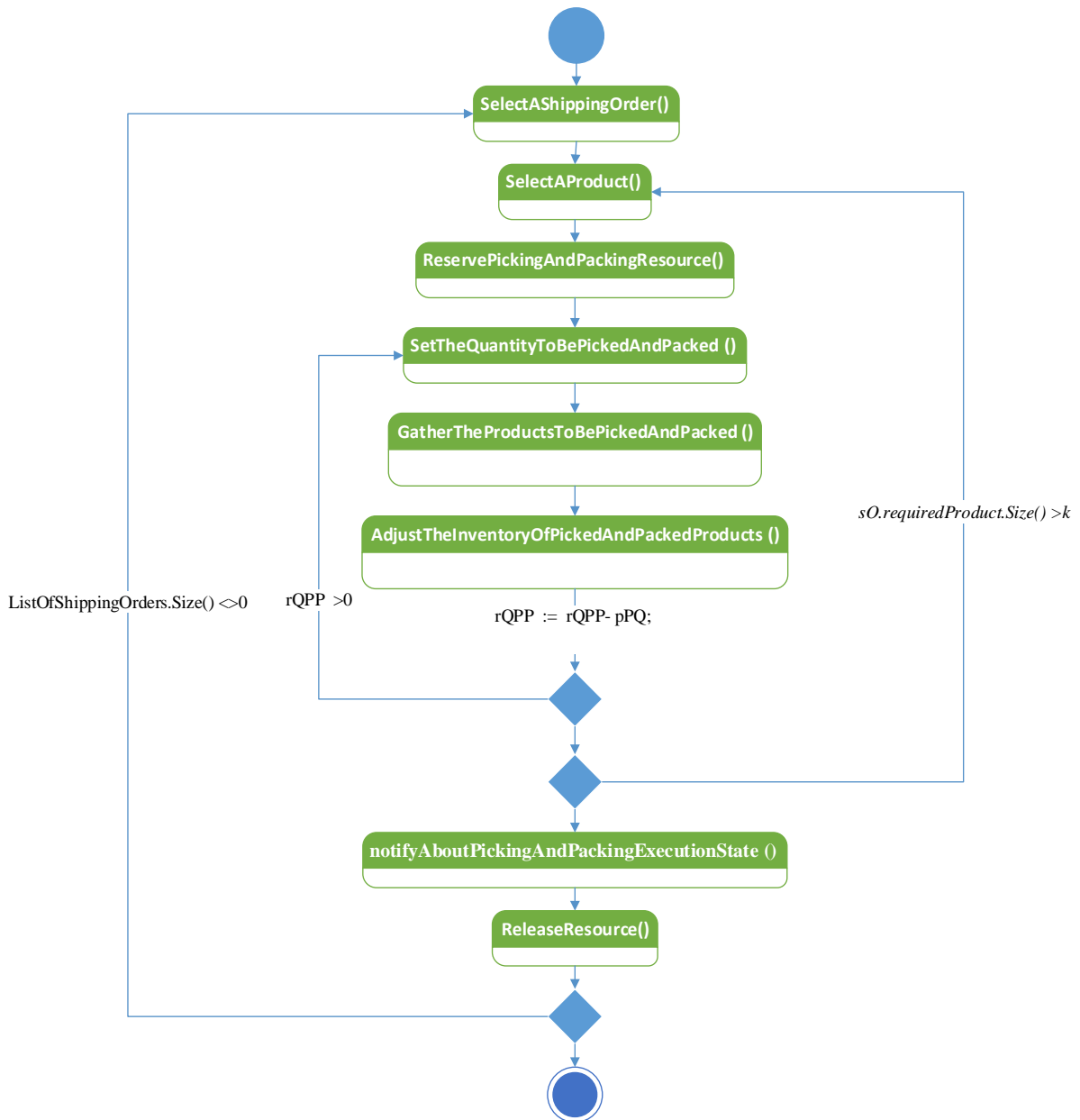


FIGURE A1.9: THE STATE MACHINE OF THE ALGORITHM OF THE PICKANDPACK OPERATION

## Internal variables

Aside from the variables already mentioned in Table A1.6, we need some internal variables for the algorithm of the **PICKANDPACK** Operation. In Table A1.6, we summarize those internal variables.

TABLE A1.6: THE INTERNAL VARIABLES USED IN THE ALGORITHM OF THE PICKANDPACK OPERATION

Internal variables	Designations
rQPP	remainingQuantitytoBePickedAndPacked.
pPQ	PickedAndPackedQuantity.
ListOfShippingOrders	List of stored received shipping orders.

## Methods

In the following tables, we provide the pseudo codes of the procedures (methods) that are used in the algorithm of the PICKANDPACK Operation (See figure A1. 9).

<b>Method 1:</b> Public void SelectAShippingOrder()
<b>Description</b> The method is responsible for selecting the shipping order to be picked and packed based on the first in first out rule and is also responsible for saving the information about it.
<b>Algorithm</b> Public void SelectAShippingOrder() { /Select a shipping order sO based on the first in first out rule. If(ShippingOrderIsReceived==true) then { ListOfShippingOrders.Add(ReceivedShippingOrder); } EndIF If (ListOfShippingOrders.size()==1&& ListOfShippingOrders[1].Status "PickedAndPacked") then {sO:= ListOfShippingOrders [1] ; } EndIF /Hold shipping orders until executing the current one/ Do { wait ; } while (sO.Status!= "PickedAndPacked") }EndWhile /Release a new shipping order/ If (sO.Status== "PickedAndPacked") then { sO:=ListOfShippingOrders.Next(); sO.Status := "ReleasedForPickingAndPacking"; k=0; } EndIf }

---

**Method 2:** Public void SelectAProduct()

**Description**

The method is responsible for selecting the Product to be picked and packed from the products mentioned in the shipping order and is also responsible for saving the information about it.

**Algorithm**

```
Public void SelectAProduct() {
ppRP:= sO.requiredProduct [k];
ppRQ := sO.RequiredQuantity[ppP];
eCT := ppR.CycleTime[ppP];
/ initialize/
rQPP := ppRQ;
pPQ:=0;
}
```

Public void ReservePickingAndPackingResource()

**Description**

This method is responsible for reserving the resource to be used for picking and packing products.

**Algorithm**

```
Public void ReservePickingAndPackingResource() {
Do{
If ( ppR.Available==true && ppR.Allocated==false) then
{ ppR.Allocated:= true; }
Else {Wait;}
EndIf;}
While (ppR.Available == false Or ppR.Allocated==false)
}EndWhile
}
```

Public void SetTheQuantityToBePickedAndPacked ()

**Description**

This method is responsible for the definition of the quantity to be picked and packed. The quantity to be picked and packed has to respect the available capacity of the used resource.

**Algorithm**

```
Public void SetTheQuantityToBePickedAndPacked () {
while ( rQPP pPIB.inventoryLevel( pPRP) ) Do { wait;};
If (tBPP = ppR.capacity × ppR.Number ) then { pPQ:= rQPP;}
Else {pPQ:= pPR.capacity × ppR.Number; }
EndIf
} EndWhile }
```

Public void GatherTheProductsToBePickedAndPacked ()

**Description**

This method is responsible for the gathering the quantities of products to be picked and packed from the input Buffer.



**Algorithm**

```

Public void GatherTheProductsToBePickedAndPacked () {
/ Retrieve the Product units from the input Buffer/
pPIB.InventoryLevel [pPRP]:= pPIB.InventoryLevel [ pPRP ]- pPQ ;
}

```

```

Public void AdjustTheInventoryOfPickedAndPackedProducts ()

```

**Description**

This method is responsible for adjusting the output Buffer inventory with the picked and packed products.

**Algorithm**

```

Public void AdjustTheInventoryOfPickedAndPackedProducts (){
/Advance time with picking and packing time./
currentTime:= currentTime+ eCT × pPQ;
/Increase the inventory level of the picked and packed products./
oB.InventoryLevel[pPRP ] := oB.InventoryLevel[ pPRP ]+ pPQ;
}

```

```

Public void EditInformationFeedbackAboutPickingAndPackingExecutionState ()

```

**Description**

This method is responsible for editing an information feedback that notifies about the state of the picking and packing execution.

**Algorithm**

```

Public void EditInformationFeedbackAboutPickingAndPackingExecutionState (){
iF.Identifier:= iF.GenerateId();
sO.status:= “PickedAndPacked”;
iF.NotifiedProduct:= pPRP;
iF.EditionTime:=currentTime;
iF.NotifiedOrder=sO;
iF.NotifiedBuffer:=oB;
iF.Reason=”ExecutionEnd”; }

```

```

Public void ReleaseTheResources()

```

**Description**

This method is responsible for releasing the resource used to pick and pack the products.

**Algorithm**

```

Public void ReleaseTheResource() {
/ Release the used resource and edit information feedback and modify production order/
ppR.Allocated:= false; }

```

## A1.4 THE LOADVEHICLE OPERATION (sDI.11 )

### Definition

The Operation is responsible for loading the TransportationResource with products based on a ShippingRequest. The Products are retrieved from a loading Buffer and put into a specific TransportationResource. This Operation is defined based on the SCOR Process element “sDI.11/ sD2.11 Load Vehicle and Generate Shipping Documents”.

### Inputs and outputs from the SCOR model

As shown in Table A1.7, SCOR defines one main input for this Process element which is the workflow. We capture the information mentioned in the workflow using a ShippingOrder. Furthermore, we assume that the workflow includes the information about the input Buffer, Buffer and the information about the resource to be used for loading Products and the Transportation Resource to be loaded. SCOR defines outputs that can be grouped into four sets. The first set of outputs (Shipping Documents, Load, Shipping, Verify, and Credit Information and Customer order) is relative to the information to be sent to the ship Process element. This set of outputs is covered through the ShippingOrder updated by the LOADVEHICLE Operation. The second set of outputs (Order Backlog, shipments) notifies planning about the loading realization. This set is covered with the proposed output InformationFeedback. The third set of outputs (Advance ship notice) concerns a notification to the receiver about the preparation of its shipping. We don't include this output since we suppose that the receiver is always ready to receive products. The last set of outputs is (Delivered End Items) which refers to the physical loaded products. This Process element output is modeled as a modification of the TransportationResource property *loadedQuantity*. Table A1.7 summarizes the inputs and the outputs we have retained for the LOADVEHICLE Operation from the SCOR model and the variable names that will be used to represent them in the model. Figure A1.10 describes the relation between the retained variables' blocks and the LOADVEHICLE Operation block. Figure A1.11 provides the elements of the Meta-model related to the inputs and outputs of the LOADVEHICLE Operation.

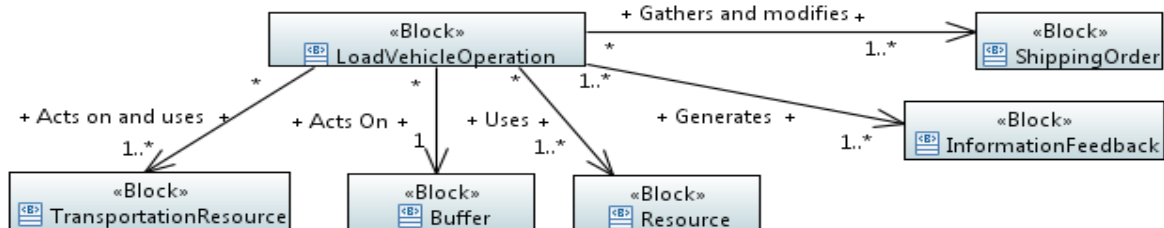
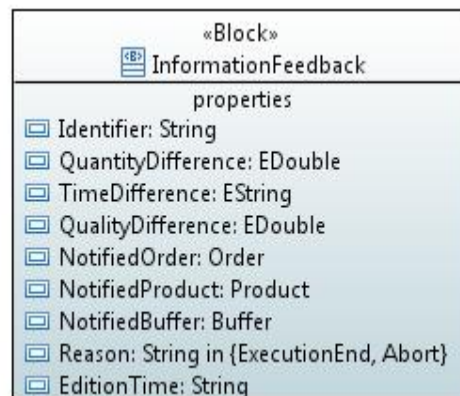
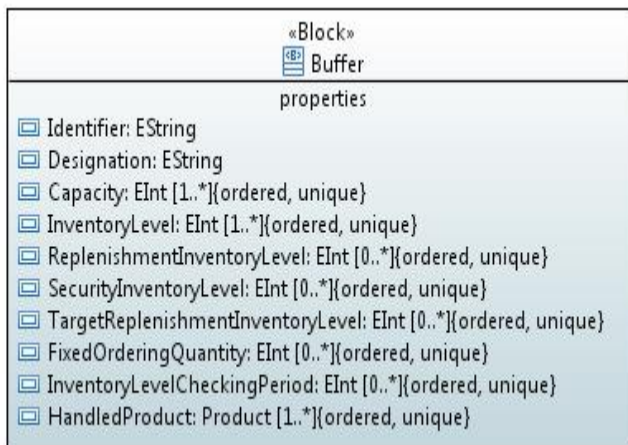
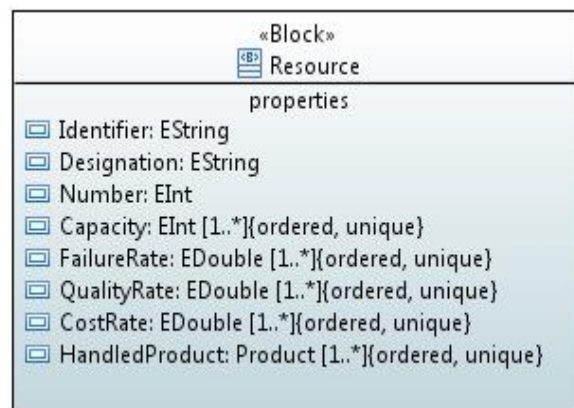
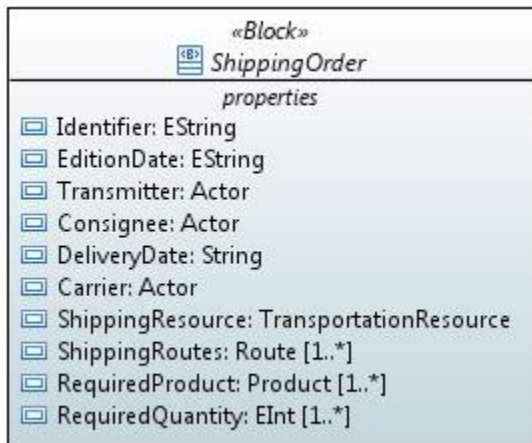


FIGURE A1.10: THE LOADVEHICLE OPERATION BLOCK

TABLE A1.7: SCOR RETAINED INPUTS AND OUTPUTS FOR THE LOADVEHICLE OPERATION

SCOR inputs and outputs	Retained inputs and outputs	Designations
<b>SCOR Inputs:</b>	<b>Inputs:</b>	
Workflow	sO [1..*]: shipping order [1..*]	It contains the information about what is to be loaded and where.
	iB:Buffer	It's the Buffer from where the products are picked to load vehicles.
	uR: Resource	It's the resource used to load vehicles with products.
	tR:TransportationResource	It's the transportation resource to be loaded with products.
<b>SCOR Outputs:</b>	<b>Outputs:</b>	
Shipping Documents Load, Shipping, Verify, and Credit Information Customer order Order Backlog Shipments Advance ship notice Delivered End Items	sO [1..*]: shipping order[1..*],	The shipping order with a modified status.
	iFd : Informationfeedback [1..*],	The notification about the execution state of the loading operation execution.



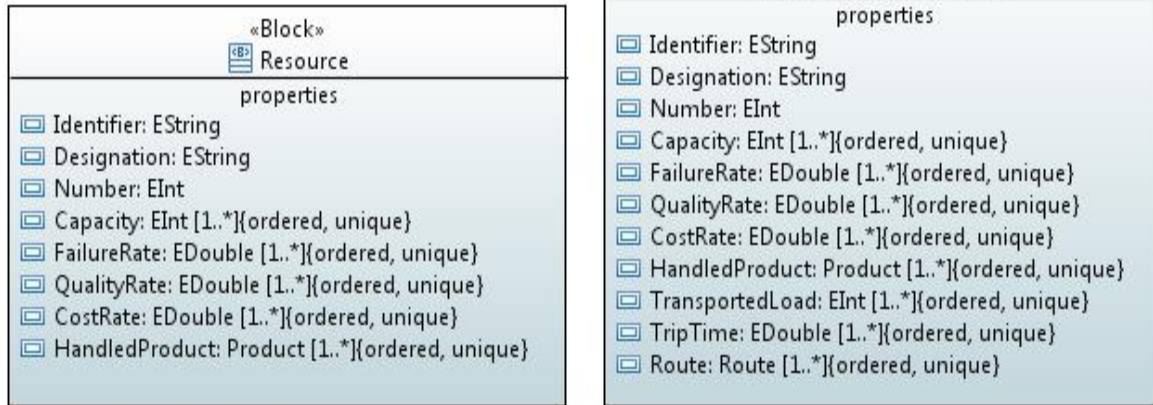


FIGURE A1.11: DETAILS OF THE USED INPUTS AND OUTPUTS FOR THE LOADVEHICLE OPERATION

## Assumptions

For the LOADVEHICLE Operation, we assume the following:

- The Operation has the ability to execute ShippingOrders containing more than one product.
- The ShippingOrders are executed by one with respect to the first in first out rule.
- The products are loaded as they are sorted in the ShippingOrders.
- The loading starts only when the required quantity is available.
- Every ShippingOrder is relative to a transportation resource.
- The Transportation Resource has to be in the loading location in order to start loading products

## Operation algorithm

The common standard mode of the LOADVEHICLE Operation can be described with the following algorithm. After receiving the ShippingOrder, a ShippingOrder is released and a Product is selected. The algorithm verifies if both the loading Resource and the Transportation Resource are available and allocable and if the required quantity is available in the loading Buffer. The required quantity is retrieved from the loading Buffer and put into the TransportationResource in several times to respect the capacity constraint of the loading Resource. The loading time depends on both the Resource used for loading and the Product type. The Operation algorithm is illustrated in the state machine of figure A1.12.

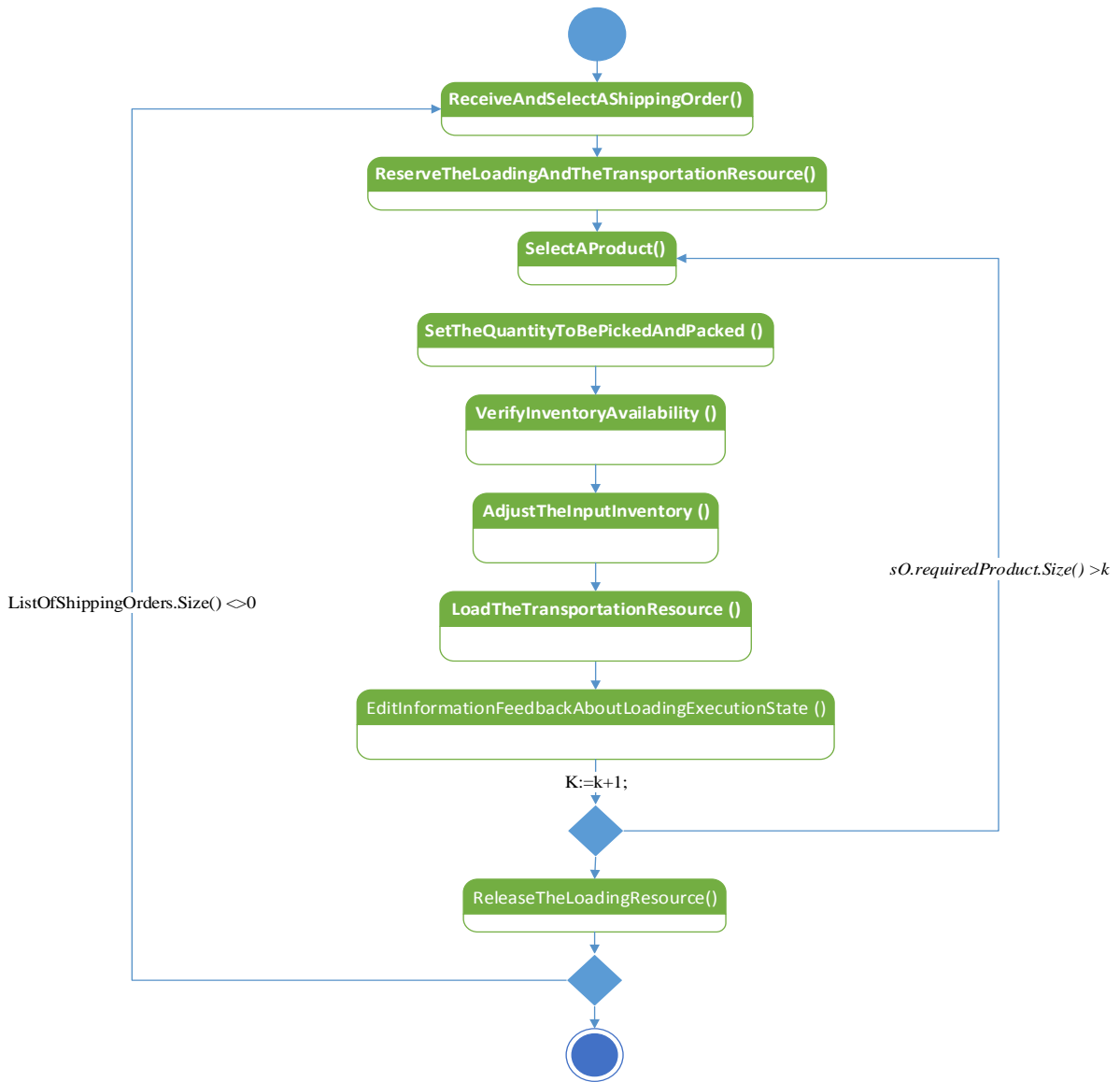


FIGURE A1.12: THE STATE MACHINE OF THE ALGORITHM OF THE LOADVEHICLE OPERATION

### Internal variables

Aside from the internal variables already mentioned in Table A1.8, we need some variables for the algorithm of the LOADVEHICLE Operation . In Table A1.8, we summarize these internal variables.

TABLE A1.8: THE INTERNAL VARIABLES OF THE ALGORITHM OF THE STANDARD MODE OF THE LOADVEHICLE OPERATION

Internal variables	Designations
ListOfShippingOrders	A list where received shipping orders are stored.
ShippingOrderIsReceived	A Boolean variable that indicates if a new shipping order is received.
productsAreAvailable	A Boolean variable that takes “true” if the loading and transportation resources are available.
K	An integer counter that refers to the current loaded products.
rIQ	Remaining quantity to be loaded due to the differed loading of the required quantity to many times. This is to consider the loading resource capacity constraint.
plQ	A portion of the required quantity to be loaded.

## Methods

In the following tables, we provide the pseudo codes of the procedures (methods) that are used in the algorithm of the LOADVEHICLE Operation (see figure A1.12).

Public void ReceiveAndSelectAShippingOrder()
<b>Description</b> The method is responsible for receiving shipping orders and selecting the shipping order to be loaded based on the first in first out rule and is also responsible for saving the information about it.
<b>Algorithm</b> <b>Public void</b> ReceiveAndSelectAShippingOrder(){ Select a shipping order sO based on the first in first out rule. If(ShippingOrderIsReceived==true) then { ListOfShippingOrders.Add(ReceivedShippingOrder); EndIf If ( ListOfShippingOrders.size()==1&& ListOfShippingOrders[1].Status “Loaded”) then {sO:= ListOfShippingOrders [1] ; } EndIf /Hold shipping orders until loading the current one/ Do { wait ; } while (sO.Status!= “Loaded”) EndWhile /Release a new shipping order/ If (sO.Status== “PickedAndPacked”) then { sO:=ListOfShippingOrders.Next(); sO.Status := “ReleasedForLoading”; k=0; } EndIf }

Public void ReserveTheLoadingAndTheTransportationResource()
<b>Description</b>

This method is responsible for reserving the resource to be used for loading the products and the transportation resource where products are to be loaded.

**Algorithm**

```
Public void ReserveTheLoadingAndTheTransportationResource(){
Do{
    If ( IR.Available==true && IR.Allocated==false && tR.Available==true &&
        tR.Allocated==false) then {

        IR.Allocated:= true; tR.Allocated:=true; }

    Else {Wait;}
    EndIf
    While (IR.Available==false Or IR.Allocated==true Or tR.Available==false Or
        tR.Allocated==true) }EndWhile
```

**Method 3: Public void SelectAProduct()**

**Description**

The method is responsible for selecting the Product to be loaded from the products mentioned in the shipping order and is also responsible for saving the information about it.

**Algorithm**

```
Public void SelectAProduct() {
k:=k+1;
IP:= sO.requiredProduct [k];
IQ := sO.RequiredQuantity[IP];
eCT := IR.CycleTime[IP]× IR.Number; }
```

**Method 4: Public void VerifyInventoryAvailability ()**

Description: This method verifies the availability of the quantity to be loaded by verifying the condition (IB.inventoryLevel [IP ] IQ). If the required quantity is not available the operation waits until the products become available.

**Algorithm**

```
Public void VerifyInventoryAvailability (){
/Verify the availability of the products to be loaded /
while (IB.inventoryLevel [IP] IQ) do{ waits; productsAreAvailable:=false; } }
```

**Method 5: Public void SetTheQuantityToBeLoaded ()**

**Description**

This method is responsible for cutting the required quantity into smaller quantities that respect the available capacity of the loading resource. Furthermore, the quantity to be loaded needs to respect the transportation resource capacity.

**Algorithm**

```
Public void SetTheQuantityToBeLoaded() {
If (rlQ = tR.capacity × tR.Number ) then { plQ:= IQ;}
Else {plQ:= tR.capacity × tR.Number } ;
EndIf }
```

**Method 6: Public void AdjustTheInputInventory ()**



**Description**

The method adjusts the loading Buffer inventory ( IB.inventoryLevel[IP] ) by decreasing it with the quantity to be loaded plQ.

**Algorithm:**

```
Public void AdjustTheInputInventory () {
/ Adjust the input inventory
IB.inventoryLevel[IP]:= IB.inventoryLevel[IP] – plQ; }
```

**Method 7: Public void LoadTheTransportationResource ()****Description**

This method is responsible for loading the required quantity in the transportations resource. In fact, this method adjusts the stimulation current time by adding the loading cycle time. Furthermore, the method modifies the information about the transportation resource such as the ( tR.transportedLoad[k]) and the ( tR.handledProduct[k]).

**Algorithm**

```
Public void LoadTheTransportationResource () {
/ Inject loading delay into simulation time./
Simulation.currentTime:= Simulation.currentTime+ plQ × eCT ;
/Adjust the vehicle load and add the loaded Product to the list of handled products./
tR.transportedLoad[k]:= tR.transportedLoad+ plQ;
tR.handledProduct[k] := IP; }
```

**Method 8: Public void EditInformationFeedbackAboutLoadingExecutionState ()****Description**

This method is responsible for editing an information feedback that notifies about the state of the picking and packing execution.

**Algorithm**

```
Public void EditInformationFeedbackAboutLoadingExecutionState (){
sO.status:= “Loaded”;
iF.Identifier:= iF.GenerateId();
iF.EditionTime:=currentTime;
iF.NotifiedOrder=sO.identifier;
iF.NotifiedProduct:= IP;
iF.NotifiedBuffer= IB;
iF.Reason=”ExecutionEnd”; }
```

**Method 9: Public void ReleaseTheLoadingResource()****Description**

This method is responsible for releasing the resource to be used for loading the products.

**Algorithm**

```
Public void ReleaseTheLoadingResource(){
IR.Allocated:= false; }
```

## A1.5 THE SHIPPRODUCT OPERATION (sDi.12+A+sSRi.5)

### Definition

The Operation is responsible for the transportation of Products from an initial Facility (The loading facility) to another Facility (The reception facility) using TransportationResources according to a ShippingOrder.

### Inputs and outputs from the SCOR model

The SHIPPRODUCT Operation is defined mainly based on the SCOR Process element “sD1.12/sD2.12 Ship Product” and integrates also the Process element “sSRi.5 Return defective/MRO return/excess product”.

As shown in Table A1.9, SCOR defines inputs for both the Process element “sDi.12” and the Process element “sSRi.5”. Those inputs can be grouped into three sets. The first set (Shipping documents, customer order, scheduled Defective Product Return and load, Shipping, Verify, and Credit Information) expresses the required details to execute the shipping. We think that those inputs can be covered with the proposed flow input “shipping order” and the inputs the TransportationResource and the Route. The second set of inputs (return inventory availability) expresses the information about the products to be returned. This set is covered by the proposed input ShippingOrder. The third set of inputs (Returned Defective Product) refers to the Products to be shipped. This set is modeled through updating the property *TransportedLoad* and the property *loadedProduct* of the transportation resource construct. Also, SCOR defines outputs for both the Process element sDi.12 and the Process element sSRi.5. Those outputs can be grouped into two sets. The first set (Workflow, Customer order, Shipping Document) expresses the information to be communicated to the receiver Actor. That information is covered through the ShippingOrder that is communicated to the receiver Actor. The second set of inputs (Returned Defective Product) refers to the shipped Products. This set is modeled through a modification of the property *inventoryLevel* of the output Buffer construct. Table A1.9 summarizes the inputs and the outputs we have retained for the SHIPPRODUCT Operation from the SCOR model and the variable names that will be used to represent them in the model. Figure A1.13 describes the relation between the retained variables’ blocks and the SHIPPRODUCT Operation block. Figure A1.14 provides the elements of the Meta-model related to the inputs and outputs of the SHIPPRODUCT Operation.

TABLE A1.9: RETAINED INPUTS AND OUTUPUTS FROM THE SCOR MODEL FOR THE SHIPPRODUCT OPERATION

SCOR inputs and outputs	Retained inputs and outputs	Designations
<b>SCOR inputs:</b>	<b>Inputs:</b>	
Shipping documents, Customer order, Scheduled Defective Product Return, Load, Shipping, Verify, and Credit Information, Return Inventory Availability, Returned Defective Product	sO : Shipping order[1..*] {Ordered}.	It describes the shipping details.
	tR: transportationResource	It’s used to transport products.
	R: Route	The one followed to transport products from a facility to another.
<b>SCOR outputs:</b>	<b>Outputs:</b>	
Workflow, Customer order, Shipping Documents, Returned Defective Product	sO : Shipping order [1..*] {Ordered}.	The order with a modified status.

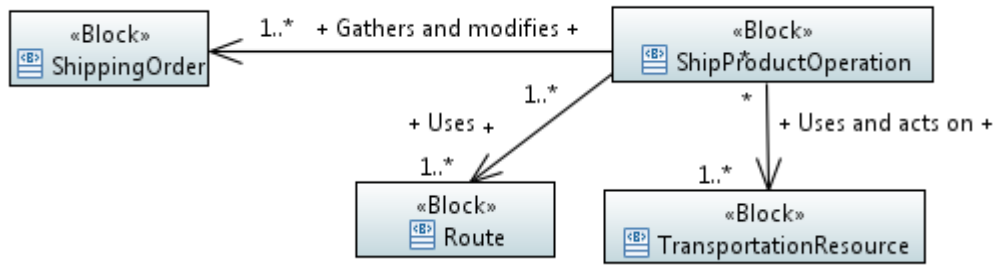


FIGURE A1.13: THE SHIPPRODUCT OPERATION BLOCK DEFINITION DIAGRAM

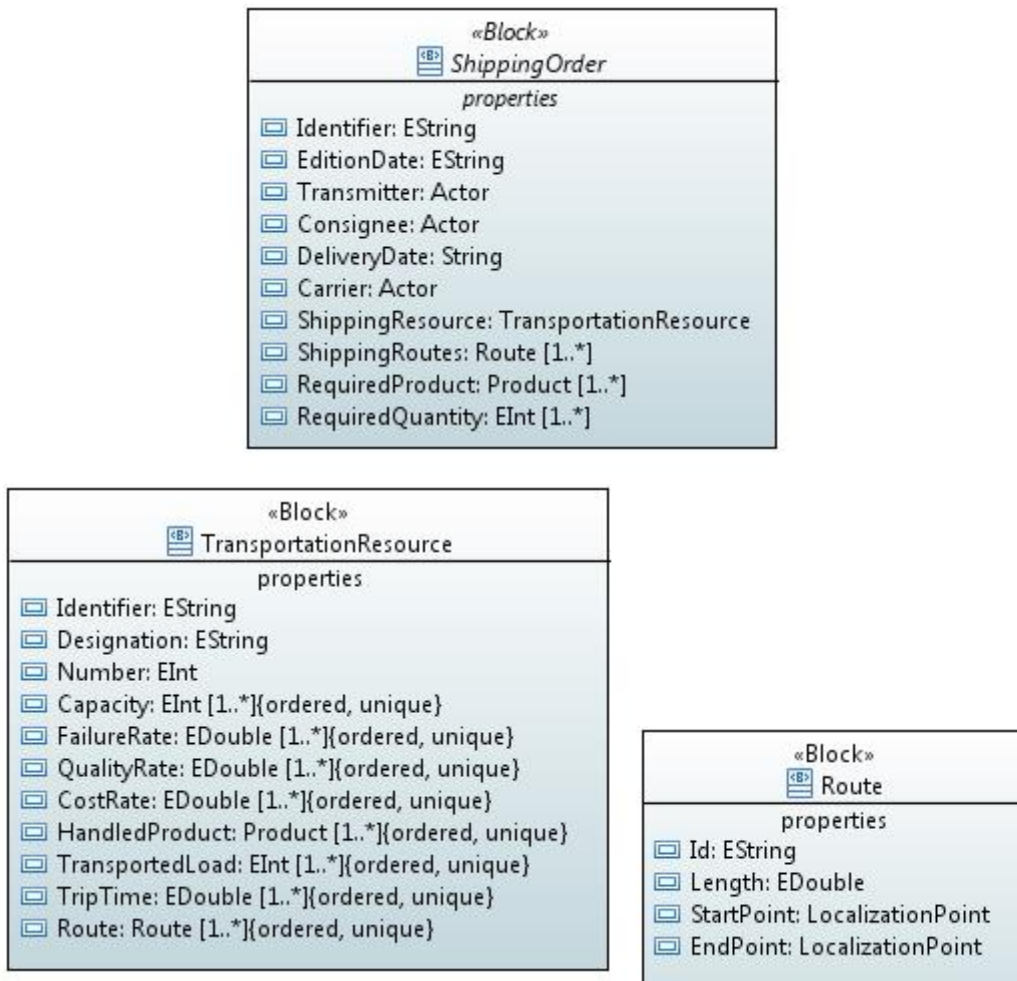


FIGURE A1.14: DETAILS OF THE USED INPUTS AND OUTPUTS FOR THE SHIPPRODUCT OPERATION

### Assumptions

For the SHIPPRODUCT Operation, we assume the following:

The time to return from the reception Facility is not considered for Transportation Resources since the return is usually done in hidden time.

The duration of the routing from the loading location to the unloading location is determined by the selected Route and the TransportationResource *trip time*.

The used vehicle (TransportationResource) may belong to either the supplier or the logistic provider.

**Operation algorithm**

The algorithm of the most common Operation Mode of the SHIPPRODUCT Operation is detailed in the state chart of figure A1.15. First, the received “shipping orders” are stored in a list. A “shipping order” is selected from the list. The transportation time is determined based on the *trip time* of the TransportationResource and the Route length. The Transportation Resource is moved to the unloading location.

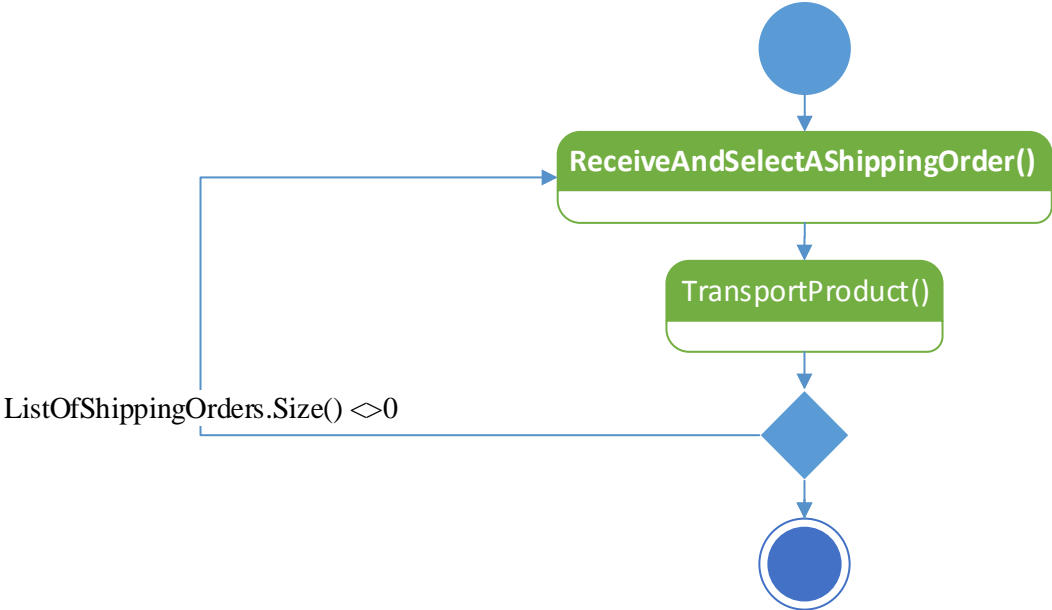


FIGURE A1.15: THE STATE MACHINE OF THE SHIPPRODUCT OPERATION

**Algorithm internal variables**

Aside from the variables already mentioned in Table A1.9, we need some internal variables for the algorithm of the SHIPPRODUCT Operation. In Table A1.10, we summarize those internal variables.

TABLE A1.10: INTERNAL VARIABLES USED IN THE ALGORITHM OF THE STANDARD MODE OF THE SHIPPRODUCT OPERATION

Internal variables	Designations
transportationTime	The time required for the vehicle to make a trip from the loading location to the unloading location.
ShippingOrderIsReceived	A Boolean variable which indicates if a new shipping order is received or not.
ListOfShippingOrders	A list where received shipping orders are stored.

**Methods**

In the following tables, we provide the pseudo codes of the procedures (methods) that are used in the algorithm of the SHIPPRODUCT Operation (See figure A1.15).

---

**Method 1: Public void ReceiveAndSelectAShippingOrder()****Description:**

The method is responsible for receiving shipping orders and selecting the shipping order to be loaded based on the first in first out rule and is also responsible for saving the information about it.

**Algorithm :**

```
Public void ReceiveAndSelectAShippingOrder(){
Select a shipping order sO based on the first in first out rule.
If(ShippingOrderIsReceived==true) then {
    ListOfShippingOrders.Add(ReceivedShippingOrder);
}EndIf
If ( ListOfShippingOrders.size()==1&& ListOfShippingOrders[1].Status “Shipped”) then
    {sO:= ListOfShippingOrders [1] ; }
EndIf
/Hold shipping orders until shipping the current one/
Do {wait ;} while (sO.Status!= “Shipped”) EndWhile
/Release a new shipping order/
If ( sO.Status== “Shipped”) then {
    sO:=ListOfShippingOrders.Next();
    sO.Status := “ReleasedForShipping”;
    uRo :=sO.ShippingRoutes ;
    k=0;
EndIf }
```

**Method 2: Public void TransportProducts()**

Description: This method is responsible for transporting products to the reception facility through the scheduled route.

**Algorithm:**

```
Public void TransportProducts() {
/ Determine Transportation time/
For i from 1 to uRo.size do {
    transportationTime := transportationTime+ tR.Speed * usedRoute[i].length;
}EndFor
/ Update simulation time/
Simulation.currentTime:= Simulation.currentTime+ transportationTime;}
```

**A1.6 The RECEIVEPRODUCT OPERATION (sSi.2+A+sDi.13+A+sDRi.3)****Definition**

The Operation is responsible of unloading the received products from the Transportation Resource and checking if the reception is done as it was scheduled to be or not ( received at the committed time).

**Inputs and outputs from SCOR model**

The Operation is defined based on the SCOR Source Process elements” **sS i.2 receive product**” integrated with a part of the Deliver Process element “**sD1.13/sD2.13 receive and verify Product by customer**” and a part of the deliver return Process element “**sDRi.3 Receive defective/MRO return/excess Product**”.

As shown in Table A1.11, SCOR defines a set of inputs for the Process elements that defines the proposed Operation. The inputs can be grouped into three sets. The first inputs set (Shipping Documents) expresses the information transferred to the receiver by the deliverer that summarizes the content of the shipment. This set is covered by the proposed “shipping order” input. The second inputs set (Scheduled Receipts, Return Schedule instructions) expresses the internal information about the requirement that the received shipment need to respect. This set is covered by the ScheduledReception construct that specifies the requirements for a specific reception of the product. Furthermore, we suppose that those inputs contain the information about the resource to be used for reception, the information about the TransportationResource to be unloaded and the information about the Buffer where products need to be put. While the third set of SCOR inputs (Defective Products, Excess Products, MRO Products, Product, Returned Defective Product) refers to the received products. This set is modeled through changing the value of the property *transportedLoad* of the TransportationResource. SCOR defines also a set of outputs for the considered Process elements. The outputs can be grouped into two sets; the first set (Receipt Verification, Receipt Discrepancy Notification) expresses a notification about the state of the received Product. This outputs set is covered by both the ReceiptVerificationNotification and the modified ShippingOrder. The second set of outputs (Product, Returned Defective Product) refers to the received products. This set is modeled through the modification of the property *inventoryLevel* of the output Buffer. Table A1.11 summarizes the inputs and the outputs we have retained for the RECEIVEPRODUCT operation from the SCOR model and the variable names that will be used to represent them in the model. Figure A1.16 describes the relation between the retained variables’ blocks and the RECEIVEPRODUCT Operation block. Figure A1.17 provides the elements of the Meta-model related to the inputs and outputs of the RECEIVEPRODUCT Operation.

TABLE A1.11: RETAINED INPUTS AND OUTPUTS FROM THE SCOR MODEL FOR THE OPERATION RECEIVEPRODUCT

SCOR inputs and outputs	Retained inputs and outputs	Designations
<b>SCOR inputs:</b>	<b>Inputs:</b>	
Defective Products	sO: ShippingOrder [1..*] {Ordered}.	It describes what to receive.
Excess Products	rS: ReceiptSchedule [1..*] {Ordered}.	It describes the requirements for the expected reception.
MRO Products	oB: Buffer	The place where received products are put.
Product	rR: Resource	The resource used for unloading vehicles.
Returned Defective Product	tR: TransportationResource	The vehicle to be unloaded.
Scheduled Receipts		
Shipping Documents.		
<b>SCOR outputs:</b>	<b>Outputs:</b>	
ProductReturnedDefective Product	rVN: ReceiptVerificationNotification [1..*] {Ordered}.	The Generated notification to state about the respect of delivery time and quantity requirements
Receipt Verification	sO: ShippingOrder [1..*] {Ordered}.	The shipping order with a modified status.
Receipt Discrepancy Notification		



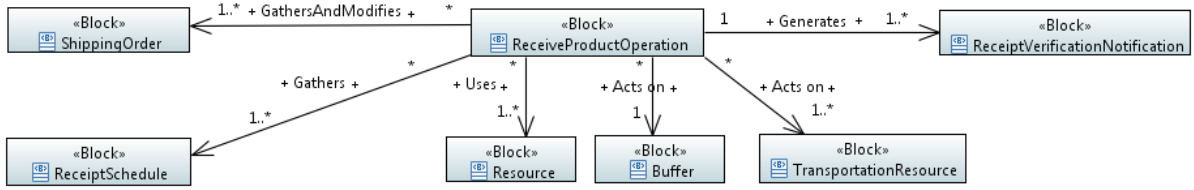


FIGURE A1.16: THE RECEIVEPRODUCT OPERATION BLOCK DEFINITION DIAGRAM

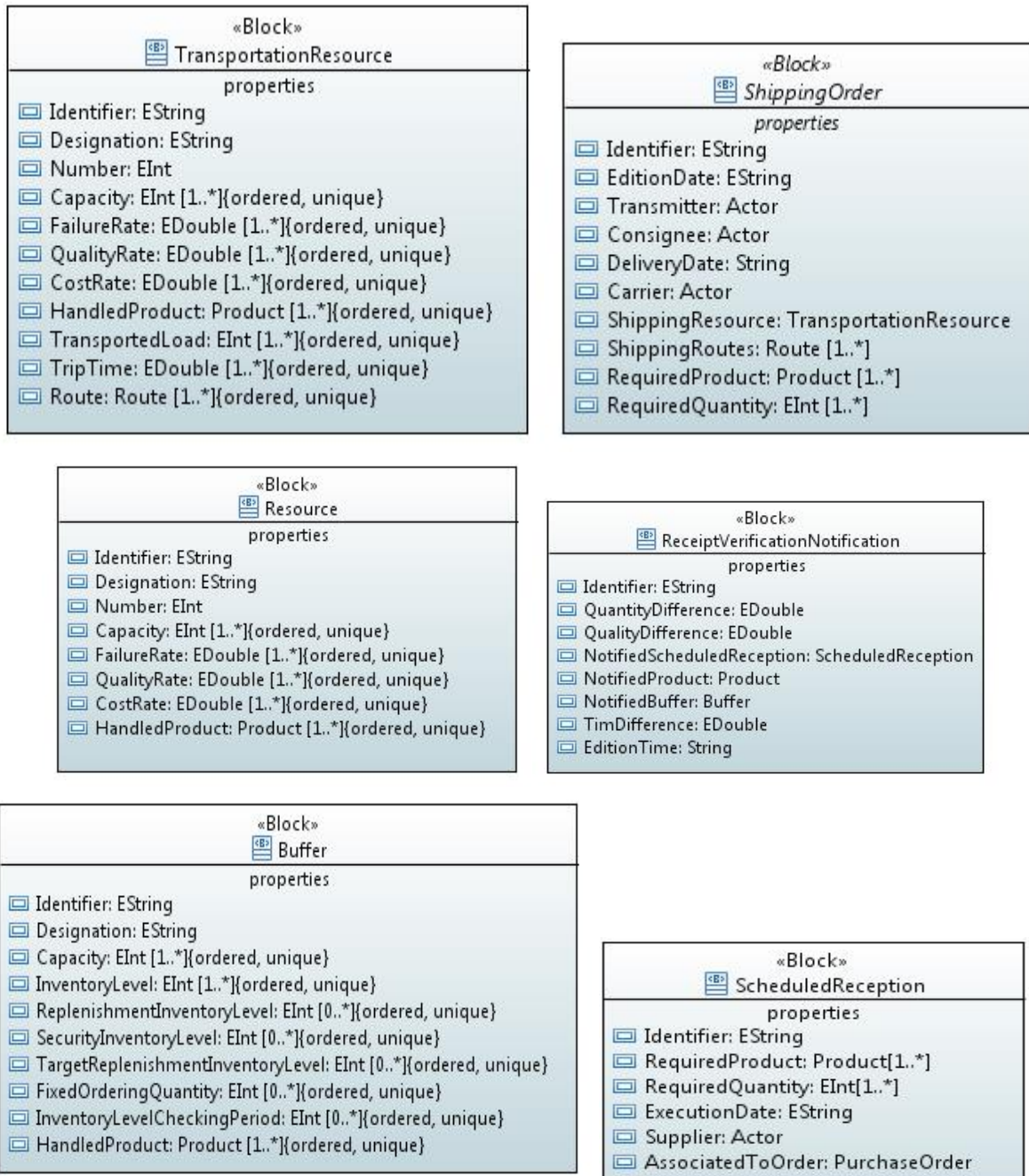


FIGURE A1.17: DETAILS OF THE USED INPUTS AND OUTPUTS FOR THE THE RECEIVEPRODUCT OPERATION

### Assumptions

For the RECEIVEPRODUCT Operation, we assume the following:

The reception of a ShippingOrder is equivalent to the reception of a shipment. The reception date of the ShippingOrder is the date of the shipment reception.



For each received ShippingOrder, a ScheduledReception object flow is supposed to be received.

The ShippingOrders are treated by one by following first in first out rule.

The TransportationResource that is reserved by the LOADVEHICLE Operation is released by the RECEIVEPRODUCT Operation.

**Operation algorithm**

The algorithm of the most common standard mode of the RECEIVEPRODUCT Operation is described as follow. First, the ShippingOrders and the ScheduledReceipts are stored in a list. A ShippingOrder is selected and its associated ScheduledReception is gathered. The reception Resource is reserved. The Products are unloaded from the TransportationResource as they are ordered in the list of transported Products. A notification about the compliance to time and quantity requirements is generated. Finally, both the TransportationResource and the reception Resource are released. The Operation algorithm is illustrated in the state machine of figure A1.18.

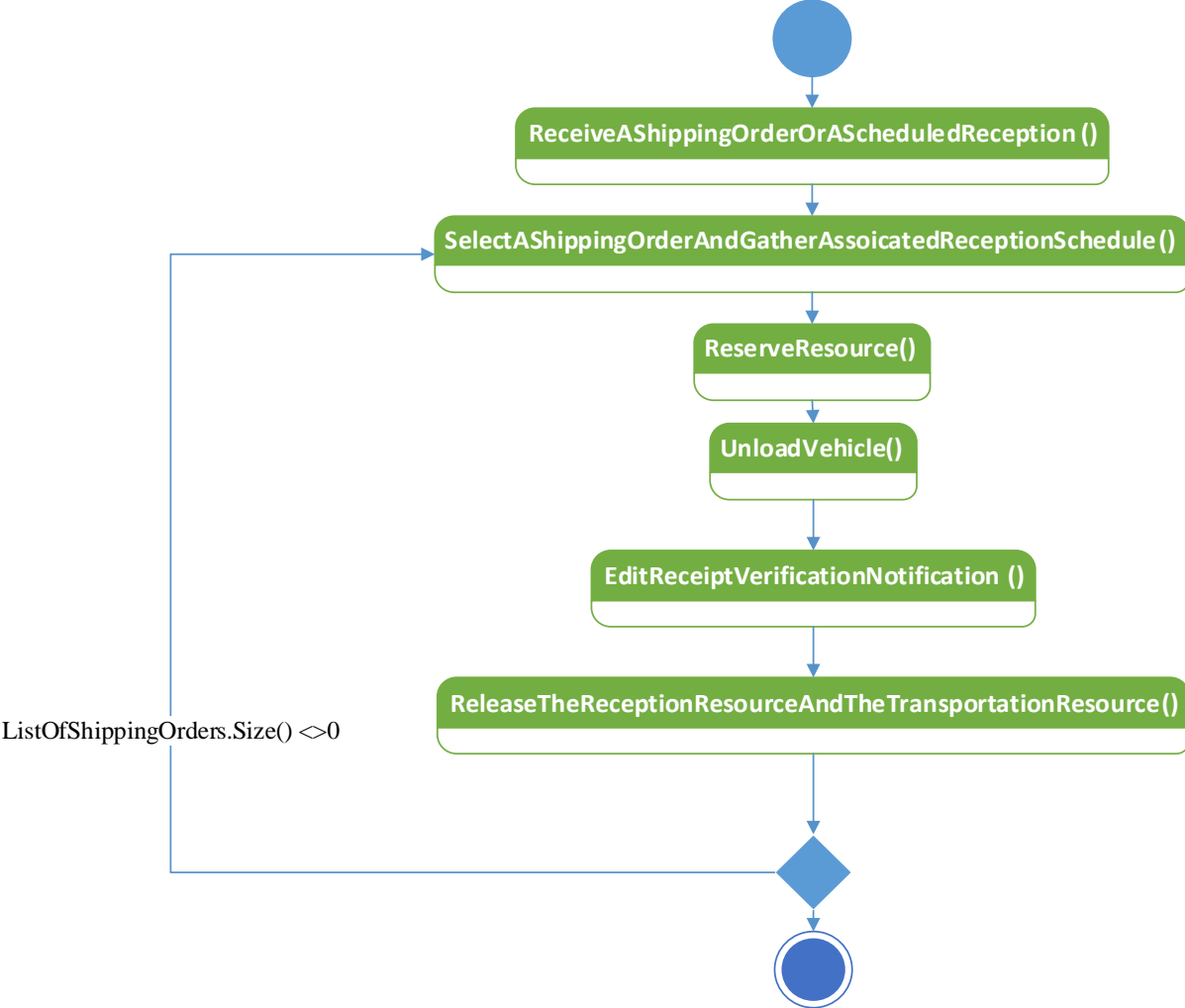


FIGURE A1.18: THE STATE MACHINE OF THE RECEIVEPRODUCT OPERATION

**Internal variables**

Aside from the variables already mentioned in Table A1.11, we need some internal variables for the algorithm of the RECEIVEPRODUCT Operation. In Table A1.12, we summarize these internal variables.

TABLE A1.12: INTERNAL VARIABLES USED IN THE ALGORITHM OF THE STANDARD MODE OF THE RECEIVEPRODUCT OPERATION

Internal variables	Designations
receivedProduct :Product [1..*] /	List of received products.
receivedQuantity:	List of received quantity.
ReceptionDate: Double[1..*]/	List of reception dates relative to shipping orders.
ShippingOrderIsReceived: Boolean	A Boolean variable that takes 1 if a scheduledReception object flow is received.
ListOfShippingOrders : shippingOrder[1..*]	List of received shipping orders objects flows.
ListOfScheduledReception : scheduledReception [1..*]	List of received scheduledReception object flows.

## Methods

In the following tables, we provide the pseudo codes of the algorithms (methods) that are used in the algorithm of the RECEIVEPRODUCT Operation (see figure A1.18).

<b>Method 1 :</b> public void ReceiveAShippingOrderOrAScheduledReception ()
<b>Description</b> This method receives a set of shippingOrder object flows and a set of scheduledReception object flows and stores them in a list. The method selects a ship to be treated. The method stores the information about it and then gathers the associated scheduledReception.
<b>Algorithm</b> Public void ReceiveAScheduledReceptionOrAShippingOrder(){ If ( ShippingOrderIsReceived==True) then { ListOfShippingOrders.Add(ReceivedShippingOrder); ReceptionDate[ReceivedShippingOrder]:=currentTime; } EndIf  If (ScheduledReceptionIsReceived == True) then { ListOfScheduledReception.Add(ReceivedScheduledReception); } } EndIf }

<b>Method 2:</b> public void SelectAShippingOrderAndGatherAssociatedScheduledReception ()
<b>Description</b> This method selects a shipping order relative to the received delivery and the related scheduled reception.
<b>Algorithm</b> Method 2: public void SelectAShippingOrderAndGatherAssociatedScheduledReception () { Select a shipping order based on the first in first out rule. If (ListOfShippingOrders.size()==1&& ListOfShippingOrders[1].Status “Received”) then {sO:= ListOfShippingOrders [1] ; } ENDIF / Holds shipping orders until receiving the current one/ Do {wait ;} while (sO.Status!= “Received”) EndWhile /Release a new shipping order/

```

If (sO.Status== “Received”) then {
  sO:=ListOfShippingOrders.Next();
  sO.Status := “ReleasedForReception”;
  For k from 1 to ListOfScheduledReception.size() {
    If (ListOfScheduledReception[k].associatedOrder==sO) then
      {sR:= ListOfScheduledReception[k];}
    EndIf
  }EndFor
}EndIF }}

```

**Method 3:** Public void ReserveResource()

**Description**

This method is responsible for reserving the resource to be used for unloading the products.

**Algorithm**

```

Public void ReserveResource(){
Do{
If ( rR.Available==true && rR.Allocated==false) then
{rR.Allocated:= true; rR.Allocated:=true; }
Else {Wait;}
EndIf
While (rR.Available==false Or rR.Allocated==true)
}EndWhile }

```

**Method 4:** Public void unloadVehicle()

**Description**

This method is responsible for unloading the vehicle and putting products in the reception Buffer of the receiver. This is by adjusting the transportedLoad of the transportation resource and by adjusting the inventory level of the reception Buffer.

**Algorithm**

```

public void unloadVehicle() {
For i from 1 to tR.loadedProduct.size() {
  unloadingTime:= rR.cycleTime[tR. handledProduct[i]] ×
  FloorFunction (sO.requiredQuantity [tR. handledProduct[i]] / (rR.Capacity[tR.
  handledProduct[i]] × rR.Number));
  SimulationTime.currentTime:= SimulationTime.currentTime+ unloadingTime;
  tR.TransportedLoad[tR. handledProduct[i]]:= tR.TransportedLoad[tR.
  handledProduct[i]] - sO.requiredQuantity[tR. handledProduct[i]] ;

  receivedProduct[i]:= tR. handledProduct[i];
  receivedQuantity[[tR. handledProduct[i]] := sO.requiredQuantity [tR. handledProduct[i]
  ];

  oB.InventoryLevel[tR. handledProduct[i]] := oB.InventoryLevel[tR.
  handledProduct[i]] + sO.requiredQuantity[tR. handledProduct[i]] ;
}EndFor}

```

**Method 5: Public void EditReceiptVerificationNotification ()****Description**

This method is responsible for editing an information feedback that notifies about the state of the picking and packing execution.

**Algorithm**

```
Public void EditReceiptVerificationNotification (){
  For i from 1 to receivedProduct.size() {
    rVN.Identifier:= rVN.GenerateId();
    rVN.EditionTime=currentTime;
    rVN.NotifiedOrder:=sO;
    rVN.NotifiedBuffer:=oB;

    rVN.NotifiedProduct:= receivedProduct[i];
    rVN.QuantityDifference:= receivedQuantity[receivedProduct[i]]-
    sR.requiredQuantity[receivedProduct[i]];
    rVN.TimeDifference:= receptionDate[sO]- sR.ExecutionDate;

    If (rVN.QuantityDifference 0) then
      { rVN.QuantityReport:=” non-compliance”;}
    Else { rVN.QuantityReport:=” compliance “;}
    EndIf
    if (rVN.TimeDifference 0 ) then
      { rVN.TimeReport:=” non-compliance”;}
    Else { rVN.TimeReport:=” compliance “;}
    EndIF
  }EndFor
}
```

**Method 6: Public void ReleaseTheReceptionResourceAndTheTransportationResource()****Description**

This method is responsible for releasing the resource used to pick and pack the products.

**Algorithm**

```
Public void ReleaseTheReceptionResourceAndTheTransportationResource(){
  rR.Allocated:= false;
  tR.Allocated:= false; }
```

**A1.7 THE VERIFY OPERATION (sSi.3+A+sDi.13+A+sDRi.3)****Definition**

The Operation is responsible of separating conforming Products from non-conforming ones and filling the notification about it.

**Inputs and outputs From SCOR model**

The Operation is defined based on the source Process element “sS 1.3/sS2.3 Verify Product” of SCOR integrated with the parts responsible for verification relative to the Deliver Process element “sDi.13: receive and verify Product by customer” and relative to the Deliver Return Process element “sDRi.3 Receive defective/MRO return/excess Product (includes verify)”.

As shown in Table A1.13, SCOR defines a set of inputs for the Process elements that define the VERIFY operation. The inputs can be grouped into two sets. The first set of SCOR inputs (Defective Products, Excess Products, MRO Products, Product and Returned Defective Product) refers to the received Products. This set is modeled through a modification in the property *inventoryLevel* of the input Buffer. The second set of SCOR inputs (Receipt Verification) refers to the notification that has to be completed by the VERIFY Operation in order to notify about the conformity state of the received products.

SCOR defines also a set of outputs for the considered Process elements. The outputs can be grouped into two sets: The first set (Receipt Verification, Receipt Discrepancy Notification) expresses a notification about the state of the received product. This output set is covered by the ReceiptVerificationNotification. In fact, the ReceiptVerificationNotification produced by the RECEIVE Operation is filled with the information about the quantity of non-conforming Products. The second set of outputs (Product, Returned Defective Product) refers to the verified Products. This set is modeled through the modification of the property *inventoryLevel* of the Buffers of good products and the Buffer of non-conforming products. Table A1.13 summarizes the inputs and the outputs we have retained for the VERIFY Operation from the SCOR model and the variable names that will be used to represent them in the model. Figure A1.19 describes the relation between the retained variables' blocks and the VERIFY Operation block. Figure A1.20 provides the elements of the Meta-model related to the inputs and outputs of the VERIFY Operation.

TABLE A1.13 : RETAINED INPUTS AND OUTPUTS FOR THE VERIFY OPERATION FROM THE SCOR MODEL

SCOR inputs and outputs	Retained inputs and outputs	Designations
<b>SCOR inputs:</b>	<b>Inputs:</b>	
Receipt Verification	rVN: ReceiptVerificationNotification [1..*] {Ordered}.	The received notification that includes only the information about the satisfaction of the time and quantity requirements
Defective Products		
Excess Products	iB: Buffer	The Buffer from where the products to be verified are picked
MRO Products	gB:Buffer	The Buffer where the verified good products are put
Product	dB: Buffer	The Buffer where the verified defective products are put
Returned Defective Product	uR: Resource	The resource used to verify the received products
<b>SCOR outputs:</b>	<b>Outputs:</b>	
Receipt Verification	rVN: ReceiptVerificationNotification [1..*] {Ordered}.	The notification completed with the information about the satisfaction of the quality requirements.
Receipt DiscrepancyNotification		
Product		
Returned Defective Product		

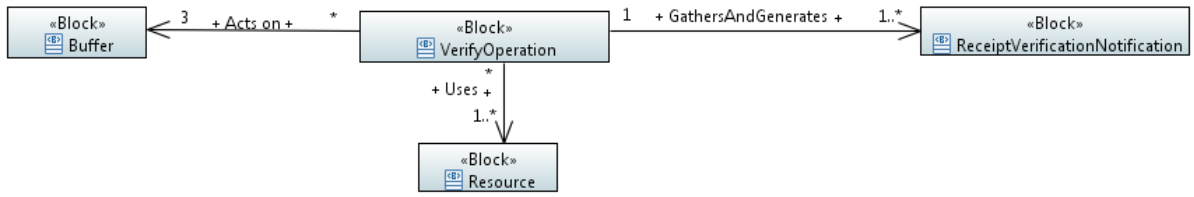


FIGURE A1.19: THE VERIFY OPERATION BLOCK DEFINITION DIAGRAM

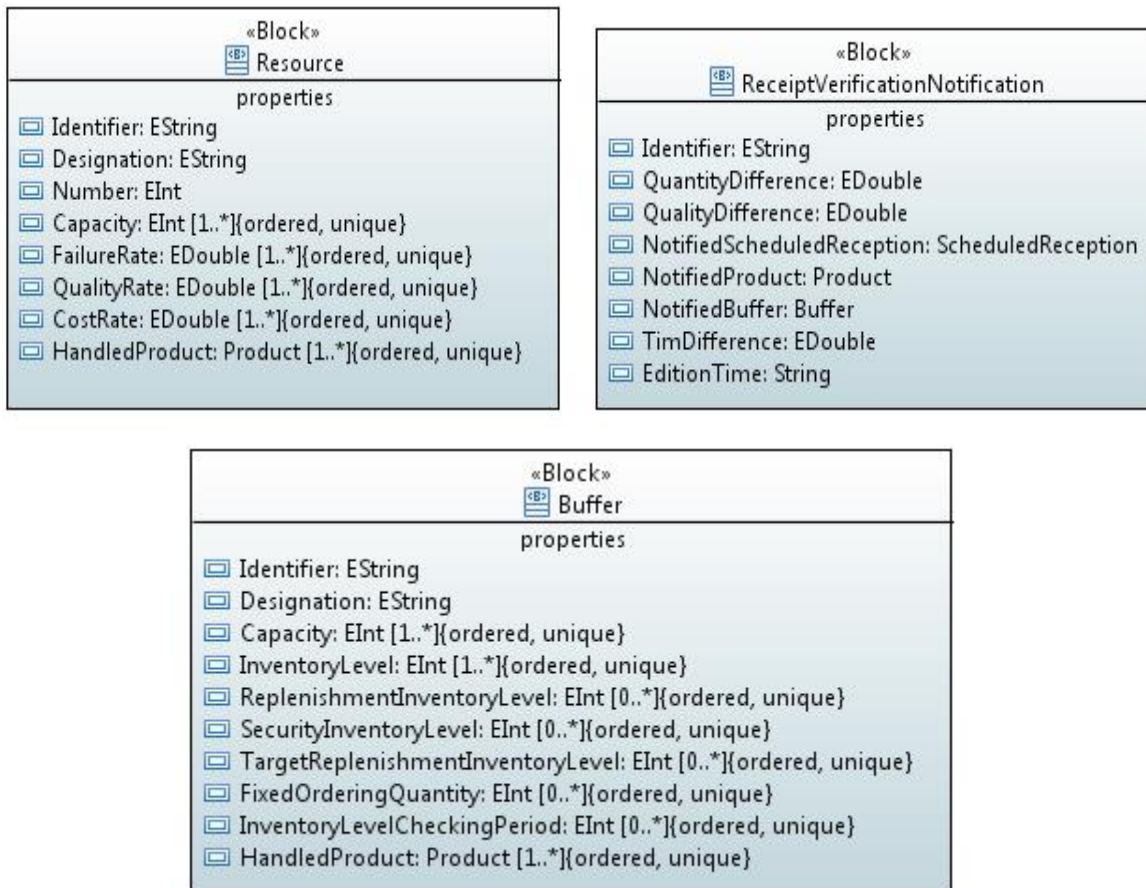


FIGURE A1.20: DETAILS OF THE USED INPUTS AND OUTPUTS FOR THE VERIFY OPERATION

### Assumptions

For the VERIFY Operation, we assume the following:

The verification starts when receiving a ReceiptVerificationNotification.

The ReceiptVerificationNotification are treated by one, following first in first out rule.

### The operation algorithm

The common standard Mode of the VERIFY Operation is as follows. The received ReceiptVerificationNotification are stored in a list. A ReceiptVerificationNotification object flow is selected. The verification Resource is reserved. The products are checked and the defective units are separated from the good ones. The received ReceiptVerificationNotification is filled by the information about the compliance to quality requirements. Finally, the verification Resource is released. The Operation algorithm is illustrated in the state machine shown in figure A1.21.

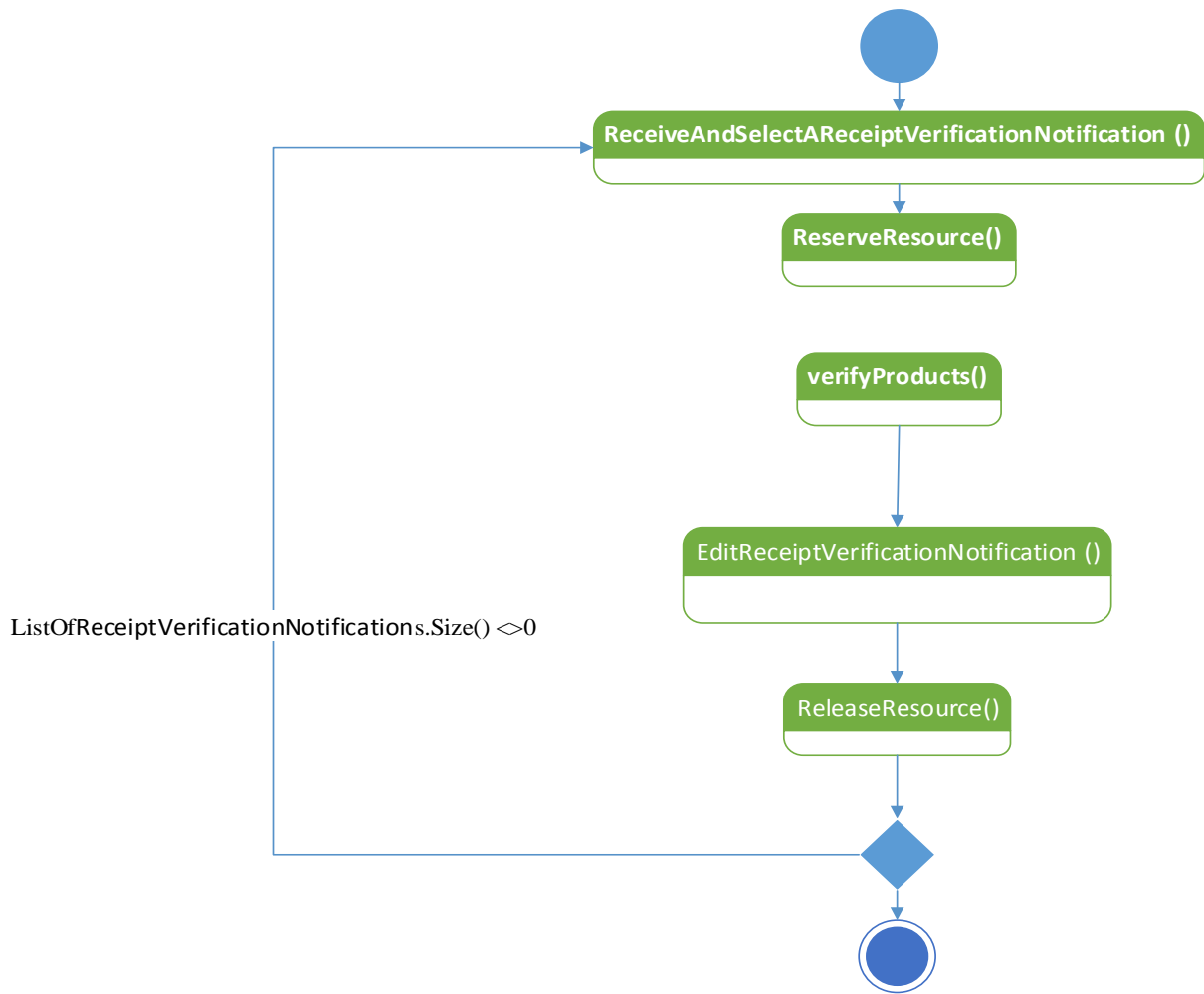


FIGURE A1.21: THE STATE MACHINE OF THE VERIFY OPERATION

### Internal variables

Aside from the variables already mentioned in table A1.13, we need some internal variables for the algorithm of the VERIFY Operation. In table A1.14, we summarize these internal variables.

TABLE A1.14: INTERNAL VARIABLES USED IN THE ALGORITHM OF THE STANDARD MODE OF THE VERIFY OPERATION

Internal variables	Designations
ReceiptVerificationNotificationIsReceived	A boolean variable that takes 1 if a ReceiptVerificationNotification object flow is received.
ListOfReceiptVerificationNotifications	List of received ReceiptVerificationNotification object flows.
EndOfCurrentVerification	A boolean variable that takes true if the current verification is executed.

### Methods



In the following tables, we provide the pseudo codes of the procedures (methods) that are used in the algorithm of the VERIFY Operation (See figure A1.21).

<b>Method 1:</b> public void ReceiveAndSelectAReceiptVerificationNotification ()
<b>Description</b> This method receives a set of ReceiptVerificationNotification object flows and selects one to be treated.
<b>Algorithm</b> Method 1: public void ReceiveAndSelectAReceiptVerificationNotification () If (AReceiptVerificationNotificationIsReceived==True) then { ListOfReceiptVerificationNotification.Add(ReceivedReceiptVerificationNotification); } EndIf  /Select a ReceiptVerificationNotification based on the first in first out rule./ If ( ListOfReceiptVerificationNotifications.size()==1) then {EndOfCurrentVerification:= true; } EndIF / Holds ReceiptVerificationNotifications until receiving the current one/ Do { wait ; } while (EndOfCurrentVerification == false) EndWhile /Release a new ReceiptVerificationNotification/ If (EndOfCurrentVerification== true) then { rVN:= ListOfReceiptVerificationNotifications.Next(); EndOfCurrentVerification:= false; } EndIF }

<b>Method 2:</b> Public void ReserveResource()
<b>Description</b> This method is responsible for reserving the resource to be used for verifying the products.
<b>Algorithm</b> Public void ReserveResource(){ Do{ If (vR.Available==true && vR.Allocated==false) then {vR.Allocated:= true; vR.Allocated:=true; } Else { Wait; } EndIf While (vR.Available==false Or vR.Allocated==true) EndWhile}

---

**Method 3: Public void verifyProducts()****Description**

This method is responsible for verifying the received products. For each product, the method adjusts the inventory level of the Buffer of received products, the inventory level of the Buffer of defective products and the inventory level of the Buffer of good products.

**Algorithm**

```
Public void VerifyProducts() {  
  
For i from 1 to rVN. notifiedProduct.size() {  
    iB.InventoryLevel[rVN. notifiedProduct[i]] := iB.InventoryLevel[rVN.  
notifiedProduct[i]] - rVN.notifiedQuantity[rVN. notifiedProduct[i]] ;  
    verificationTime:= vR.cycleTime[rVN. notifiedProduct[i]] × FloorFunction  
(rVN.notifiedQuantity [rVN. notifiedProduct [i]] / (vR.Capacity[rVN. notifiedProduct  
[i]] × vR.Number)) ;  
    SimulationTime.currentTime:= SimulationTime.currentTime+ verificationTime;  
    defectiveQuantity[rVN.notifiedQuantity[rVN. notifiedProduct[i]]]:=  
determineDefectiveQuantity(rVN.notifiedQuantity[rVN. notifiedProduct[i]]);  
    dB.InventoryLevel[rVN. notifiedProduct[i]] := dB.InventoryLevel[rVN.  
notifiedProduct[i]] + DefectiveQuantity;v  
    gB.InventoryLevel[rVN. notifiedProduct[i]] := gB.InventoryLevel[rVN.  
notifiedProduct[i]] + rVN.notifiedQuantity [rVN. notifiedProduct [i]] -  
DefectiveQuantity;  
} EndFor  
EndOfCurrentVerification:= true; }
```

**Method 4: Public void EditReceiptVerificationNotification ()****Description**

This method is responsible for updating the receiptVerificationNotification to notify about the quality of the received products.

**Algorithm**

```
Public void EditReceiptVerificationNotification () {  
For i from 1 to rVN. notifiedProduct.size() {  
    rVN.QualityDifference:= defectiveQuantity[rVN.notifiedQuantity[rVN.  
notifiedProduct[i]]]; }  
If (rVN.QualityDifference 0) then {  
    { rVN.qualityReport:=” non-compliance”;}  
Else { rVN. qualityReport:=” compliance “;}  
EndIf  
}EndFor }
```

**Method 5: Public void ReleaseTheVerifyResource()****Description**

This method is responsible for releasing the resource used to verify products.

**Algorithm**

```
Public void ReleaseTheVerifyResource() {  
vR.Allocated:= false; }
```

---

## A1.8 THE TRANSFER OPERATION (sSi.4+A+sDRi.4+A+ sD1.8)

### Definition

This Operation is responsible for transferring the products, including the returns, from one location to another and informing the requesting Operations about the new availability of the required Products.

### Input and output from SCOR model

The TRANSFER Operation is defined based on the SCOR Process element “**sSi.4 Transfer Product**” and integrates also the Process element “**sDRi.4 transfer defective/MRO return/excess product**” and the deliver Process element “**sD1.8 Receive Product from Source or Make**”.

As shown in Table A1.15, SCOR specifies many inputs for the Process elements used to form the TRANSFER Operation. Those inputs can be grouped into three sets. The first set (Replenishment Signal, Production Schedule, and the Scheduled Receipts) refers to the three types of transfer orders that trigger the Operation Mode execution. In fact, this setting specifies the Product types, the quantities to be transferred and determine the input Buffers. Those orders are captured respectively through the inputs (ReplenishmentSignal, ProductionOrder and ScheduledReception). The orders are executed only when the required inventory is already available in the input Buffer. So the TRANSFER Operation has to wait for a signal that indicates that the inventory is available for the case of the received ProductionOrder or the received ScheduledReception. SCOR specifies a set of signals for this purpose (Inventory Availability and ReceiptVerification) those inputs are associated respectively to (Production Schedule and the scheduled Receipts). We capture those inventory readiness signal through respectively (inventoryAvailabilityNotification and ReceiptVerificationNotification). The third set of inputs (Returned Defective Product and finished Product Release ) refers to the Products to be transferred. This set is modeled by adjusting the *inventory level* property of the input Buffer.

SCOR defines also outputs that can be regrouped into two sets: The first set regroup (Inventory Availability and the Return Inventory Transfer Data) while the second set regroup (Loaded Retail Cart or Pallet and the Transferred Product and Defective Products).

The first set notifies that the products are already transferred. This set is captured through an InventoryAvailabilityNotification that informs about the accomplishment of the received order. The second set is modeled by adjusting the *inventoryLevel* property of the output Buffer. Table A1.15 summarizes the inputs and the outputs we have retained for the TRANSFER Operation from the SCOR model and the variable names that will be used to represent them in the model. Figure A1.22 describes the relation between the retained variables' blocks and the TRANSFER Operation block. Figure A1.23 provides the elements of the Meta-model related to the inputs and outputs of the TRANSFER Operation.

TABLE A1.15 : RETAINED INPUTS AND OUTPUTS FROM THE SCOR MODEL FOR THE TRANSFER OPERATION

<b>SCOR inputs and outputs</b>	<b>Retained Inputs and outputs:</b>	<b>Designations</b>
<b>SCOR inputs:</b>	<b>Inputs:</b>	
Replenishment Signal Production Schedule, Scheduled Receipts, Inventory Availability, Finished Product Release, Receipt Verification, Returned Defective Product,	rS: ReplenishmentSignal [1..*] {Ordered},	A request to transfer products to a given Buffer.
	pO: productionOrder [1..*] {Ordered},	It contains the information about what to be transferred.
	sR: scheduledReception [1..*] {Ordered},	It contains the information about the received products to be transferred.
	rVN: ReceiptVerificationNotification [1..*] {Ordered},	It's a notification that the reception was executed.
	iAN: inventoryAvailabilityNotification [1..*] {Ordered},	It's a notification that the products are ready to be transferred.
	iB: Buffer	The Buffer from where the products to be transferred are picked
	oB: Buffer	The Buffer where the products are transferred.
	tR: TransferResource	The resource used to transfer products.
	pT: Path	The path to be followed to transfer products between the input and output Buffers.
<b>SCOR outputs:</b>	<b>Outputs:</b>	
Inventory Availability, Return Inventory Transfer Data. Loaded Retail Cart or Pallet, Transferred Product, Defective Products.	<b>iAN:</b> InventoryAvailabilityNotification [1..*] {Ordered},	A notification to state that the products were already transferred to the output Buffer.

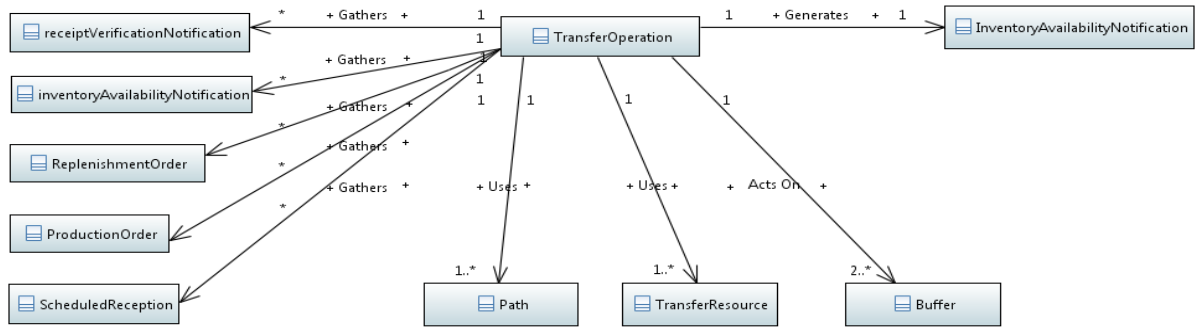


FIGURE A1.22: TRANSFER OPERATION BLOCK DEFINITION DIAGRAM

The list of references is shown in figure A1.23.

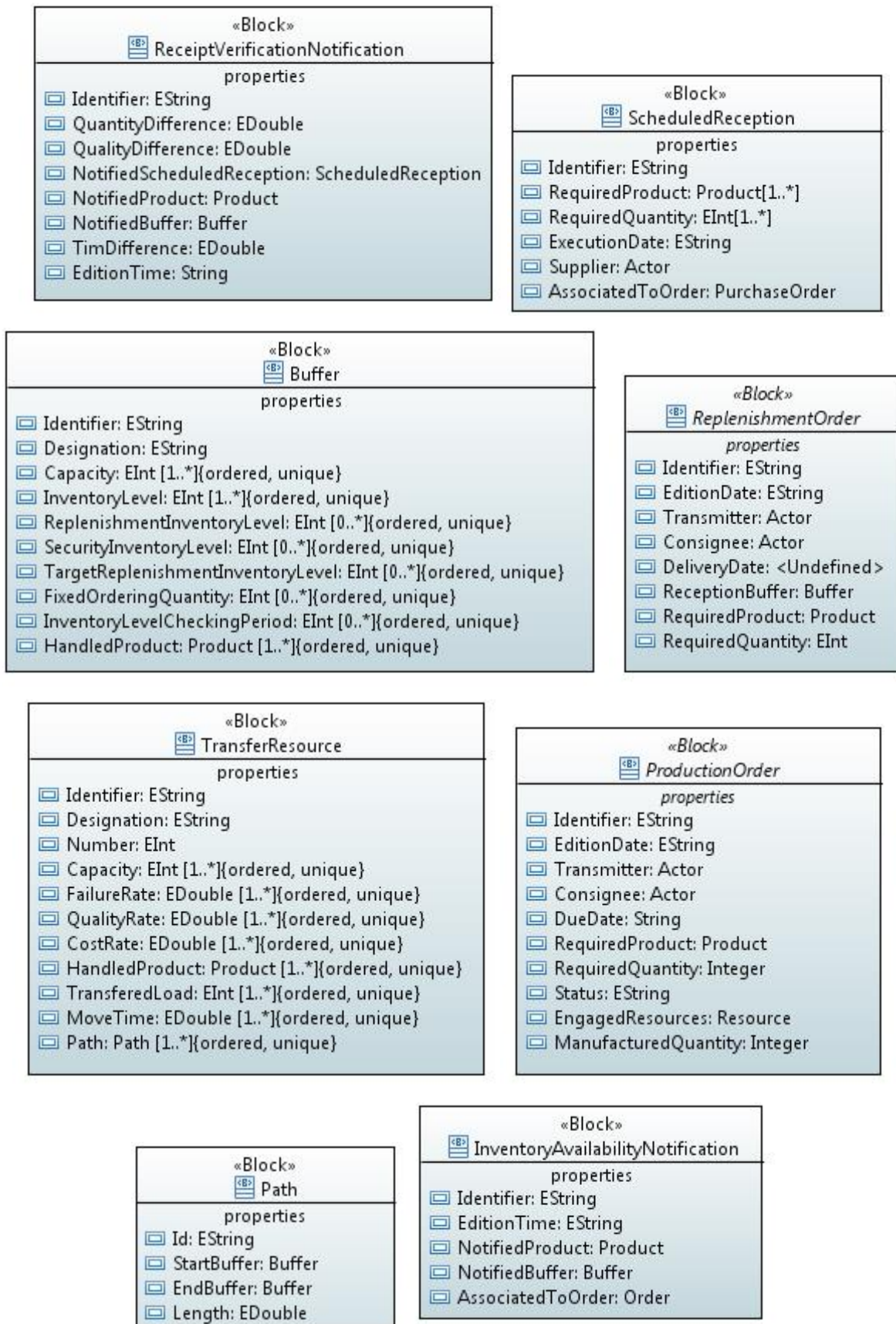


FIGURE A1.23: DETAILS OF THE USED INPUTS AND OUTPUTS FOR THE TRANSFER OPERATION

---

**Assumptions:**

For the TRANSFER Operation, we assume the following:

- The Operation handles a set of ReplenishmentSignals, a set of ProductionOrders and a set of ScheduledReception information flows;
- The ReplenishmentSignals are treated by one, following first in first out rule.
- The ReplenishmentSignal is executed only if the required inventory is available in the input Buffer.
- The quantity mentioned in the ProductionOrder is transferred only if the associated InventoryAvailabilityNotification is received,
- The quantity mentioned in the ScheduledReception is transferred only if the associated ReceiptVerificationNotification is received.
- Different input Buffers and output Buffers may be considered.

**The operation algorithm:**

The algorithm of the common standard mode of the TRANSFER Operation is as follows. The Operation algorithm starts by storing the set of received orders (ReplenishmentSignal, ProductionOrder and ScheduledReception). The Orders are selected by one by following first in first out rule. Only the orders that respect the launching conditions are released. In fact, a ReplenishmentSignal is released only if the required quantity can be satisfied by the available inventory. Also, a ProductionOrder is released only if a notification about the inventory availability is received. Furthermore, a ScheduledReception is released only after receiving a notification about products reception.

The TransferResource is allocated and moved to the input Buffer; it picks the Product units with respect to the transfer Capacity. The TransferResource moves the Product units to the output Buffer and returns to the input Buffer position for the remaining quantities. After finishing the transfer execution, a Notification is sent to inform about the availability of products in the output Buffer. The Operation execution method algorithm is explained by a state machine in figure A1.24.



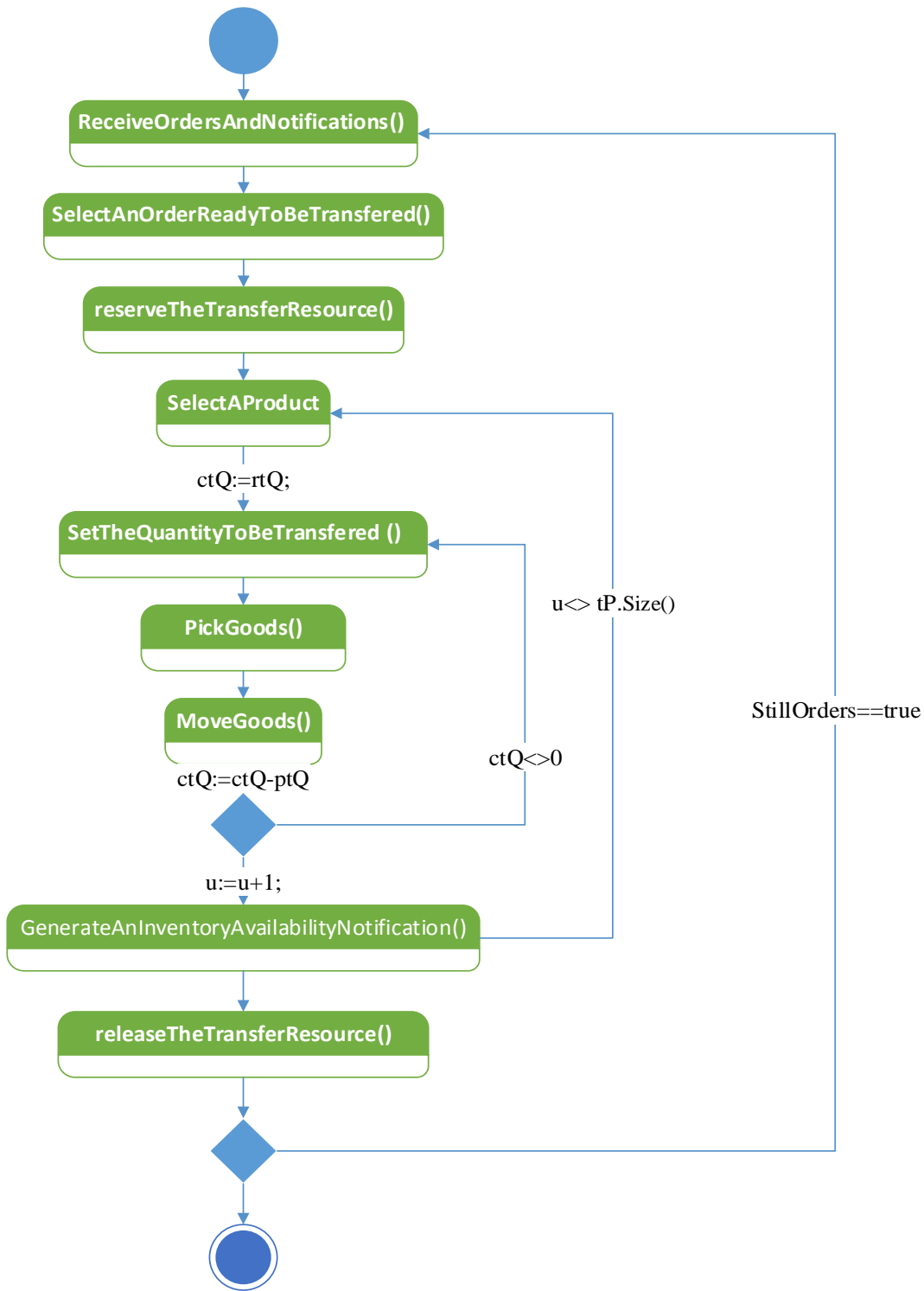


FIGURE A1.24: THE STATE MACHINE OF THE TRANSFER OPERATION

## Internal variables

Aside from the internal variables already mentioned in Table A1.15, we need some internal variables for the algorithm of the TRANSFER Operation. In A1.16, we summarize these internal variables.

TABLE A1.16: INTERNAL VARIABLES USED IN THE ALGORITHM OF THE STANDARD MODE OF THE TRANSFER OPERATION

Internal variables	Designations
ListOfNewOrders	The list of received orders.
ExecutionStatus	The execution status of the current treated order.
ReceptionOfANewOrder	A boolean variable which indicates if a new order is received.
ListOfInventoryAvailabilityNotifications	The list of inventory availability notifications.
ListOfReceiptVerificationNotifications	The list of receipt verification notifications.
currentOrderRank:	The selected order rank in the list of received orders.
selectedOrderType	The type of the selected order
pN	a number of Product types in the currently selected order.
tQ:	the list of quantities to be transferred by Product type.
tP:	The list of Product types to be transferred.
ctQ:	The remaining current quantity to be transferred;
ctP :	The current Product to be transferred;
rtQ :	the remaining quantity to be transferred to the current Product type.
ptQ:	the Quantity portion of the current Product to be transferred that respects the transfer resource capacity.

## Methods:

In the following tables, we provide the pseudo code of the procedures (methods) that are used in the algorithm of the TRANSFER Operation (see figure A1.24).

<b>Method 1:</b> Public void ReceiveOrdersAndNotifications()
<p><b>Description</b> The method is responsible for receiving replenishment signals, production orders and scheduled receptions and the notification (receiptVerificationNotification and inventoryAvailabilityNotification ), saving them in lists.</p>
<p><b>Algorithm</b> Public void ReceiveOrdersAndNotifications{ / receiveOrdersAndSaveThemInAList/ If (ReceptionOfANewOrder==true) then {     ListOfNewOrders.Add(receivedOrder);}     ExecutionStatus[ListOfNewOrders.size()]= “waiting”;} EndIf If (ReceptionOfANewReceiptVerificationNotification == true ) then{     ListOfReceiptVerificationNotifications.Add(ReceiptVerificationNotification);} EndIF If (ReceptionOfANewInventoryAvailabilityNotification== true ) then {     ListOfInventoryAvailabilityNotifications.Add(inventoryAvailabilityNotification);} EndIF}</p>

**Method 2: Public void SelectAnOrderReadyToBeTransferred()****Description**

The method is responsible for selecting an order (replenishment signal, a production order or a scheduled reception) based on the first in first out rule. An order is selected if the conditions of its treatment are true. The information about the selected order is saved and the order is released.

**Algorithm**

```
Public void SelectAnOrderReadyToBeTransferred() {
num:=0;
stillOrders:=true;
For k from 1 to ListOfNewOrders.size() do{
  If (ExecutionStatus[k]== "waiting") then{
    num:=num+1;
    If (num==1) then{
      currentOrderRank:=k;
      selectedOrderType:= ListOfNewOrders[k].type();
      If (selectedOrderType==replenishmentSignal){
        pN:= 1;
        If (inventoryLevel(replenishmentSignal.requiredProduct)
replenishmentSignal.requiredQuantity) then {
          rS:= ListOfNewOrders[k];
          tQ.initialize();
          tP.initialize();
          tQ[1]:= rS.requiredQuantity;
          tP[1]:= rS.requiredProduct;
          ExecutionStatus[k]:= "Released" ;
          stillOrders:=true;
        }EndIf
      }EndIf
      If (selectedOrderType ==productionOrder){
        pN:= 1;
        For j from 1 to ListOfInventoryAvailabilityNotifications.size() {
          If (ListOfInventoryAvailabilityNotifications[j].associatedToOrder==
ListOfNewOrders[k]) {
            pO:= ListOfNewOrders[k];
            tQ.initialize();
            tP.initialize();
            tQ[1]:= pO.requiredQuantity;
            tP[1]:= pO.requiredProduct;
            ExecutionStatus[k]:= "Released" ;
            stillOrders:=true;
          }EndIf
        }EndFor
      }EndIf
      If (selectedOrderType ==scheduledReception){
        For j from 1 to ListOfReceiptVerificationNotifications.size() {
          If (ListOfReceiptVerificationNotifications[j].associatedToOrder==
ListOfNewOrders[k]) {
            sR:= ListOfNewOrders[k];
            pN:= sR.requiredProduct.size();
```

```

        For i from 1 to sR.requiredProduct.size(){
            tQ.initialize();
            tP.initialize();
            tQ[i]:= sR.requiredQuantity[i];
            tP[i]:= sR.requiredProduct[i];
        } EndFor
        ExecutionStatus[k]:= “Released”;
        stillOrders:=true;
    }EndIf
}EndFor
}EndIf
}EndIf
}EndIf
}EndFor }

```

**Method 3:** Public void reserveTheTransferResource ()

**Description**

This method is responsible for reserving the transfer resource to be used to transfer products from the input Buffer to the output Buffer. If the resource is not available, the method waits until the transfer resource becomes available.

**Algorithm**

```

Public void reserveTheTransferResource () {
While (tR.available== false) {wait;} EndWhile
tR.available:= false;
}

```

**Method 4:** Public void SelectAProductType ()

**Description**

This method is responsible for selecting the Product type to be treated.

**Algorithm**

```

Public void SelectAProductType () {
ctQ:= tQ[u];
ctP:=tP[u];
}

```

**Method 5:** Public void SetTheQuantityToBeTransferred ()

**Description**

This method is responsible for cutting the required quantity into smaller quantities that respect the available capacity of the transfer resource.

**Algorithm**

```

Public void SetTheQuantityToBeTransferred (){
If (rtQ = tR.capacity[ctP] × tR.Number[ctP] ) then
    { ptQ:= rtQ;}
Else {ptQ:= tR.capacity[ctP] × tR.Number[ctP] ;}
EndIF}

```

---

**Method 6: Public Void PickGoods()****Description**

This method picks the goods to be transferred from the input Buffer (iB).

**Algorithm**

```
Public Void PickGoods() {  
iB.inventoryLevel(ctP):=iB.inventoryLevel(ctP)- ptQ;  
tR.transferredLoad(ctP):= tR.transferredLoad(ctP)+ ptQ;  
}
```

**Method 7: Public Void moveGoods ()****Description**

This method adjusts the inventory level of the output Buffer by adding the unloaded quantity to the current inventoryLevel and modifies the load size (tR.transferredLoad )of the transfer resource.

**Algorithm**Public Void dropGoods () {

```
simulation.currentTime:= simulation.currentTime+ MoveTime(Path(iB,oB)) ;  
oB.inventoryLevel(ctP):=oB.inventoryLevel(ctP)+ tR.TransferredLoad(ctP);  
tR.transferredLoad(ctP):= tR.transferredLoad(ctP)- unloadedQuantity;
```

}

**Method 8: Public Void GenerateAnInventoryAvailabilityNotification ()****Description**

When all the required quantity for a given order is transferred an inventoryAvailabilityNotification is generated to inform about the new current inventory level of transferred products.

**Algorithm**

```
Public Void GenerateAnInventoryAvailabilityNotification () {  
/Notify about the inventory level change to concerned operation./  
ExecutionStatus[currentOrderRank]:= “Released”; / modify the status of the current order/  
NotifiedProduct:= ctP;  
NotifiedBuffer:= oB;  
AssociatedToOrder:= ListOfNewOrders[currentOrderRank];  
ListOfNewOrders.DeleteLink(AssociatedToOrder);}
```

**Method 9: Public void releaseTheTransferResource ()****Description**

This method is responsible for releasing the transfer resource, after finishing treating the current order.

**Algorithm**

```
Public void releaseTheTransferResource () {  
tR.available:= true; }
```

## A2 THE LIBRARY OF ROLES

- The Buyer role

TABLE A2.1: BUYER ROLE PROCESS ELEMENTS

Buyer Role Process Elements	SCOR Process
sP1.1: Identify, Prioritize and Aggregate Supply Chain Requirements	Plan supply chain (SP1)
sP1.2: Identify, Prioritize and Aggregate Supply-Chain Resources	
sP1.3: Balance Supply Chain Resources with SC Requirements	
sP1.4: Establish & Communicate Supply-Chain Plans	
sP2.1: Identify, Prioritize and Aggregate Product Requirements	Plan source (SP2)
sP2.2: Identify, Assess and Aggregate Product Resources	
sP2.3 : Balance Product Resources with Product Requirements	
sP2.4: Establish Sourcing Plans.	
sS1.1/sS2.1 : Schedule Product Deliveries	Source Process (sS)
sS1.2/ sS2.2 : Receive Product	
sS1.3/ sS1.3 : Verify Product	
sS1.5/ sS1.5 : Authorize Supplier Payment	
sSR1.1: Identify Defective Product Condition	Source Return Defective Product (sSR1)
sSR1.2: Disposition Defective Product	
sSR2.1: Identify Defective Product Condition	Source Return MRO Product (sSR2)
sSR2.2: Disposition Defective Product	
sSR3.1: Identify Excess Product Condition	Source Return Excess Product (sSR3)
sSR3.2: Disposition Excess Product	
sD4.2 Receive Product at store	Deliver retail Process (sD4)
sDR1.2 : Schedule Defective Return Receipt.	Deliver Return defective Product (sDR1)
sDR1.1 : Authorize defective Product Return	
sDR1.2 : Schedule MRO Return Receipt.	Deliver Return MRO Product(sDR2)
sDR1.1 : Authorize MRO Product Return	
sDR3.1 Authorize Excess Product Return	Deliver Return Excess Product (sDR3)
sDR3.2 Schedule Excess Return Receipt	
sDR1.3 Receive Defective Product	Deliver Return Defective Product (sDR1)
sDR2.3 Receive Defective Product	Deliver Return MRO Product (sDR2)
sDR3.3 Receive Excess Product	Deliver Return Excess Product (sDR3)

- The Deliverer role

TABLE A2.2: DELIVERER ROLE PROCESS ELEMENTS

<b>Deliverer role Process elements</b>	<b>Process categories</b>
sP4.1 Identify, Prioritize and Aggregate Delivery Requirements	Plan Deliver (sP4)
sP4.2 Identify, Assess and Aggregate Delivery Resources	
sP4.3 Balance Delivery Resources and Capabilities with Delivery Requirements	
sP4.4 Establish Delivery Plans	sD.Deliver Process
sD1.4/sD2.4 Consolidate orders	
sD1.5/sD2.5 Build Loads	
sD1.6/sD2.6 Route Shipments	
sD1.7/sD2.7 Select Carriers and Rate Shipments	
sD1.11/sD2.11 Load Vehicle & Generate Shipping Docs	
sD1.12/sD2.12 Ship (finished for DTO)Product	
sD1.13/sD2.13 Receive and verify Product by Customer	
sD1.14/sD2.14 Install Product	
sD4.7 Deliver and/or install	
sSR3.5 Return Excess Product	Source Return Excess Product (sSR3)
sSR1.5 Return Defective Product	Source Return Defective Product (sSR1)

- The Storer role

TABLE A2.3: STORER ROLE PROCESS ELEMENTS

<b>Storer role Process elements</b>	<b>Process categories</b>
sP1.1. Identify, Prioritize and Aggregate supply chain (SC) Requirements	Supply chain plan Process (sP1)
sP1.2. Identify, Assess and Aggregate supply chain Resources	
sP1.3. Balance SC Resources with SC Requirements	
sP1.4. Establish and Communicate Supply Chain Plans.	
sS1.4/sS2.4: Transfer product	Source Process (sS)
sM1.2/ sM2.2 : Issue Material/ Issue Sourced or In-Process Product	Make Process (sM)
sM1.5/ sM2.5: Stage product	
sM1.6/ sM2.6 Release Product to Deliver	
sD1.8/ sD2.8 Receive Product from Source or Make	Deliver Process (sD)
sD1.9/ sD1.10 Pick Product and Pack Product	Deliver retail Product Process (sD4)
sD4.1: Generate Stocking Schedule	
sD4.3 Pick Product from backroom	
sD4.4 Stock Shelf	
sD4.5 Fill Shopping Cart	
sDR1.4 Transfer Defective Product	Return Process (sDR)
sDR3.4 Transfer Excess Product	
sDR2.4 Transfer MRO Product	



- The Maker role

TABLE A2.4: MAKER ROLE PROCESS ELEMENTS

Maker role Process elements	Process categories
SP1.1: Identify, Prioritize and Aggregate Supply Chain Requirements	Plan supply chain Process (sP1)
SP1.2: Identify, Prioritize and Aggregate Supply-Chain Resources	
SP1.3: Balance Supply Chain Resources with SC Requirements	
SP1.4: Establish & Communicate Supply-Chain Plans	
SP3.1: Identify, Prioritize and Aggregate Production Requirements	Plan make Process (sP3)
SP3.2: Identify, Assess and Aggregate Production Resources	
SP3.3: Balance Production Resources with Production Requirements	
SP3.4 Establish Production Plans.	
sM1.1/ sM2.1: Schedule Production Activities	Make Process (sM)
sM1.3/ sM2.3: Produce and Test	
sM1.4 /sM2.4: Package	
sM1.7/ sM2.7: Waste Disposal	

- The Vendor Role

TABLE A2.5: VENDOR ROLE PROCESS ELEMENTS

Vendor role SCOR Process elements	SCOR Processes
sP1.1: Identify, Prioritize and Aggregate Supply Chain Requirements	Plan supply chain (SP1)
sP1.2 : Identify, Prioritize and Aggregate Supply Chain Resources	
sP1.3 : Balance Supply Chain Resources with SC Requirements	
sP1.4 : Establish & Communicate Supply-Chain Plans	
sP4.1 : Identify, Prioritize and Aggregate Delivery Requirements	Plan Deliver (SP2)
sP4.2: Identify, Assess and Aggregate Delivery Resources	
sP4.3: Balance Product Resources with Delivery Requirements	
sP4.4: Establish & Communicate Delivery Plans	
sD1.1 / sD2.1: Process inquiry and quotes	Deliver Process (sD)
sD1.2/sD2.2 : Receive, Enter and Validate Order	
sD1.3/sD2.3 : Reserve Inventory and Determine Delivery Date	
sD1.15/ sD2.15 : Invoice	
sD4.6 Checkout	Deliver Retailer products (sD.4 )
sSR1.3 Request Defective Product Return Authorization	Source Return Defective Product (sSR1)
sSR1.4: Schedule Defective Product Shipment	
sSR2.3 Request Defective Product Return Authorization	Source Return MRO Product (sSR2)
sSR2.4: Schedule Defective Product Shipment	
sSR3.3 Request Excess Product Return Authorization	Source Return Excess Product (sSR3)
sSR3.4: Schedule Excess Product Shipment	

## A3 SC RISK COUNTERMEASURES STRATEGIES PROPOSED IN LITERATURE

TABLE A3.1: SC RISK COUNTERMEASURES STRATEGIES (PROPOSED BY JÜTTNER ET AL.(2003))

Strategies	Designations
Postponement.	It entails delaying the actual commitment of resources to maintain flexibility and delay incurring costs. Two types of postponement exist: Form postponement: includes labeling, packaging, assembly, and manufacturing. Time postponement refers to the movement of goods from manufacturing plants only after customer orders are received.
Speculation (Selective risk taking)	The principle of speculation holds that changes in form, and the movement of goods to forwarding inventories, should be made at the earliest possible time in the marketing flow in order to reduce the costs of the marketing system. Is demand-side risk management strategy.
Hedging	Hedging is undertaken by having a globally dispersed portfolio of suppliers and facilities such that a single event (like currency fluctuations or a natural disaster) will not affect all the entities at the same time and/or in the same magnitude. Is a supply side risk management strategy
Security	Is aimed at increasing a supply chain's ability to sort out what is moving, and identify unusual or suspicious elements. Security strategy also encompasses working closely with government and port officials to proactively comply with regulations.
Control/share/transfer.	Control, share, or transfer of risks takes the form of vertical integration, contracts, and agreements. Control can also be obtained through virtual supply chain integration and supply chain collaboration. Sharing or transferring risks takes place through outsourcing and/or writing flexible contracts with clauses that account for possible changes in the environment and associated risks. Sharing and transferring risk may take place in supply chains with either a short-term or a long-term focus.

TABLE A3.2: SC RISK COUNTERMEASURES STRATEGIES (PROPOSED BY MANUJ ET AL.(2008))

Strategies	Designations
Avoidance	Dropping specific products=geographical markets=supplier and=or customer organizations
Control	Vertical integration Increased stockpiling and the use of Buffer inventory Maintaining excess capacity in productions, storage, handling and=or transport Imposing contractual obligations on suppliers
Co-operation	Joint efforts to improve supply chain visibility and understanding Joint efforts to share risk-related information Joint efforts to prepare supply chain continuity plans Flexibility _ Postponement Multiple sourcing Localized sourcing1

TABLE A3.3: SC DISRUPTIONS MITIGATION COUNTERMEASURES STRATEGIES (PROPOSED BY TANG ET AL. (2006))

<b>Robust supply chain strategies</b>	<b>Main objectives</b>	<b>Benefit(s) under normal circumstances</b>	<b>Benefit(s) after a major disruption</b>
Postponement	Increases Product flexibility	Improves capability to manage supply	Enables a firm to change the configurations of different products quickly
Strategic Stock Increases	Product availability		Enables a firm to respond to market demand quickly during a major disruption
Flexible supply base	Increases supply flexibility		Enables a firm to shift production among suppliers promptly
Make-and-buy			Enables a firm to shift production between in-house production facility and suppliers rapidly
Economic supply incentives	Increases Product availability		Enables a firm to adjust order quantities quickly
Flexible transportation	Increases flexibility		Enables a firm to change the mode of transportation rapidly
Revenue management	Increases control of Product demand	Improves capability to manage demand	Enables a firm to influence the customer Product selection dynamically
Dynamic assortment planning			Improves capability to manage demand Enables a firm to influence the demands of different products quickly
Silent Product rollover	Increases control of Product exposure to customers	Improves capability to manage supply and demand	Enables a firm to manage the demands of different products swiftly

TABLE A3.4: SUPPLY DISRUPTION REACTIVE COUNTERMEASURES STRATEGIES FOR LOW-VALUE-PRODUCT (PROPOSED BY SHAO.(2012))

<b>Strategies</b>	<b>Designations</b>
Backordering	The manufacturer passively accepts the disruption and back orders customers' orders until the supplier recovers from the disruption.
Compensation	The manufacturer pays a penalty to the customers for late delivery of the product.
Mixed	The manufacturer offers customers a menu of choices when the supply of the low-value component is disrupted. Each arriving potential customer has a menu of choices, i.e., buying the high-value product, buying an upgraded version of the low-value product, ordering the low-value Product and getting a compensation for late delivery, or leaving without buying anything.
Downgrading	The customers who arrive for the high-value Product B would move to the downgraded version of Product B, and some would move to the low-value product.

## A4 MODEL OF THE SC CASE STUDY

### A4.1 MODELING THE SC STRUCTURE

#### A4.1.1 MODELING THE PRODUCT VIEW OF THE SC

The Product view model that describes the bill of material of the Products handled within the SC is shown in figure A4.1.

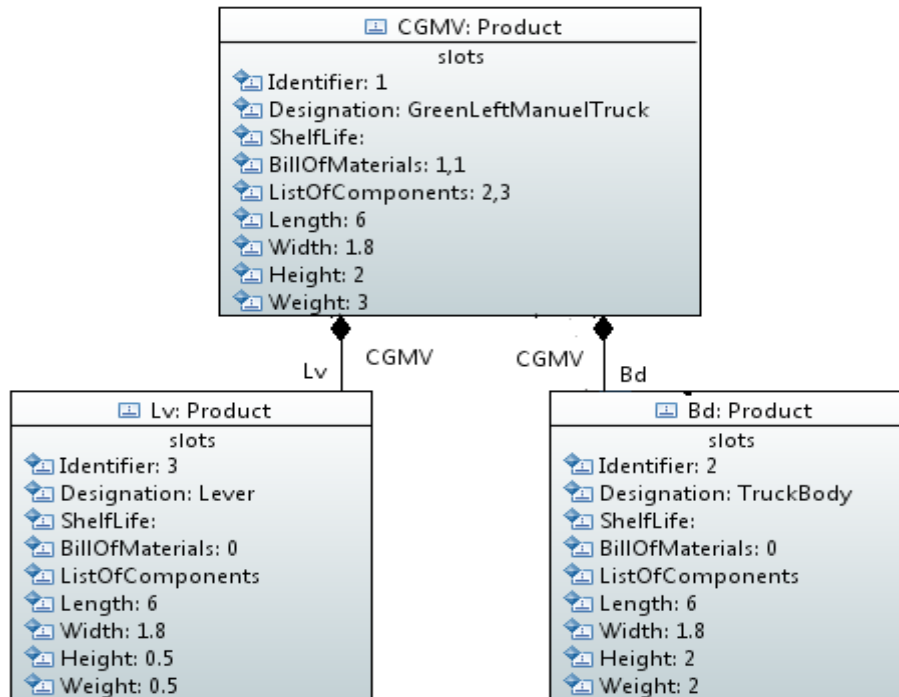


FIGURE A4.1: THE OBJECT DIAGRAM OF THE PRODUCTS CGMV, BD AND LV

#### A4.1.2 MODELING THE ACTOR'S NETWORK VIEW OF THE SC

The clauses if the Actor's relationships are defined through the Contract construct. Hence the declared contract instances for the relationships between TruckMuch and Supplier 1, between TruckMuch and Supplier2, between TruckMuch and DistC1 and between TruckMuch and DistC2 are shown respectively in the figures A4.2, A4.3, A4.4, and A4.5.

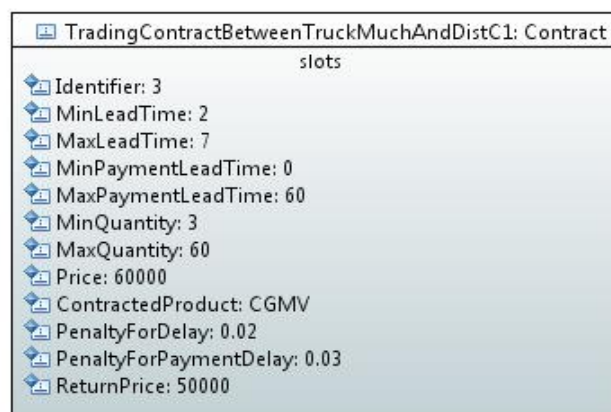


FIGURE A4.2: THE INSTANCE DIAGRAM OF THE CONTRACT BETWEEN TRUCKMUCH AND DISTC1

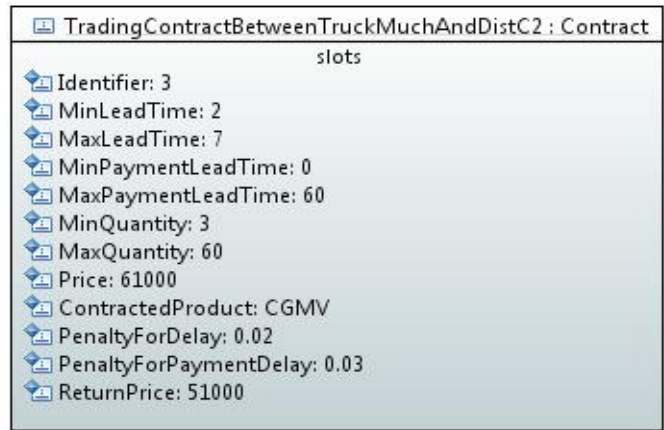


FIGURE A4.3: THE INSTANCE DIAGRAM OF THE CONTRACT BETWEEN TRUCKMUCH AND DISTC2

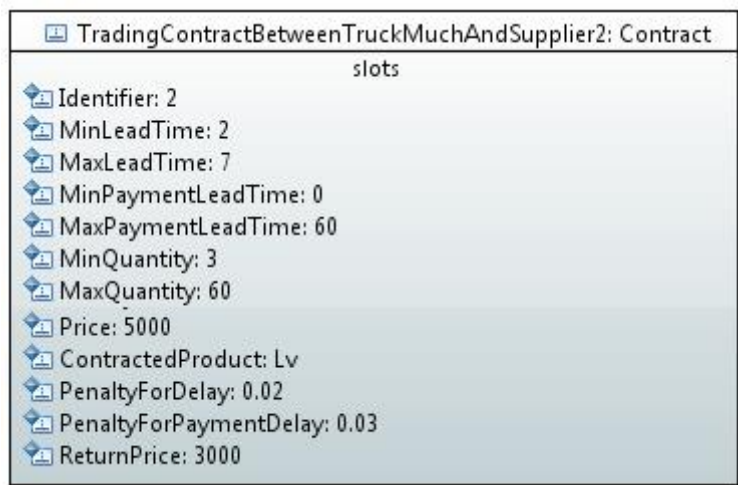


FIGURE A4.4: THE INSTANCE DIAGRAM OF THE CONTRACT BETWEEN TRUCKMUCH AND SUPPLIER2

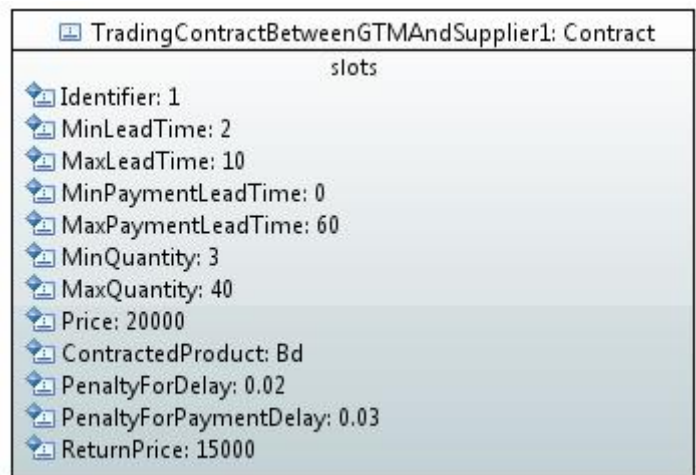


FIGURE A4.5: THE INSTANCE DIAGRAM OF THE CONTRACT BETWEEN TRUCKMUCH SUPPLIER1

### A4.1.3 MODELING THE INFRASTRUCTURE VIEW OF THE SC

#### A4.1.3.1 MODELING THE SC FACILITIES

The SC infrastructure is modeled using the Facility construct. Hence, the GTM factory, the DC1 distribution center 1, the DC2 distribution center are modeled respectively using the Facility diagrams shown in figure A4.6, A4.7, A4.8, A4.9 and A4.10.

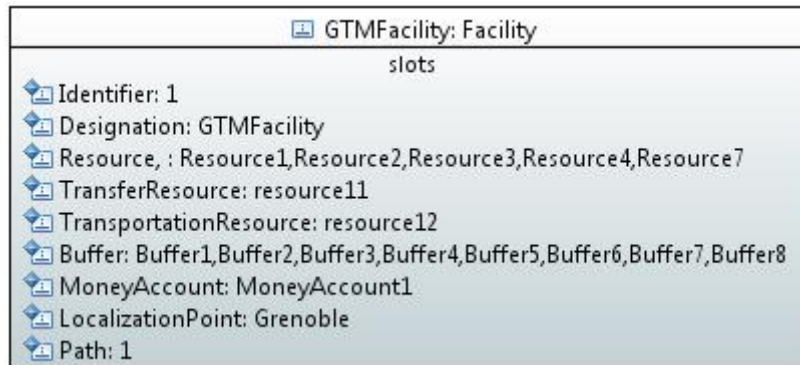


FIGURE A4.6: THE INSTANCE DIAGRAM OF THE GTM FACILITY

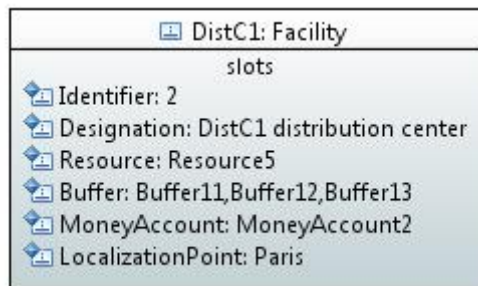


FIGURE A4.7: THE INSTANCE DIAGRAM OF THE DISTC1 FACILITY

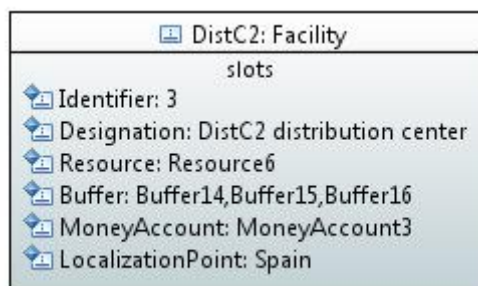


FIGURE A4.8: THE INSTANCE DIAGRAM OF THE DISTC2 FACILITY

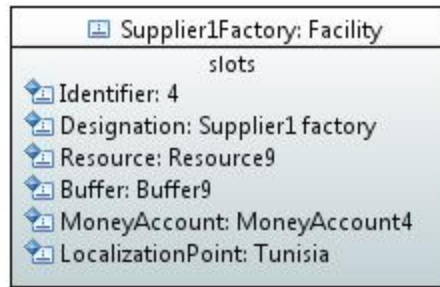


FIGURE A4.9: THE INSTANCE DIAGRAM OF THE SUPPLIER1

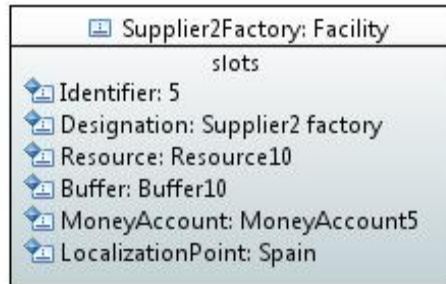


FIGURE A4.10: THE INSTANCE DIAGRAM OF THE SUPPLIER2

#### A4.1.3.2 MODELING THE SC RESOURCES

The declared Resources for the Facility GTM shown in Figure A4.11 are respectively: The Resource1 used for the reception and the verification of subassemblies, the Resource 2 used for producing the trucks, the Resource3 used for testing the manufactured trucks and the Resource7 used for loading vehicle with trucks.

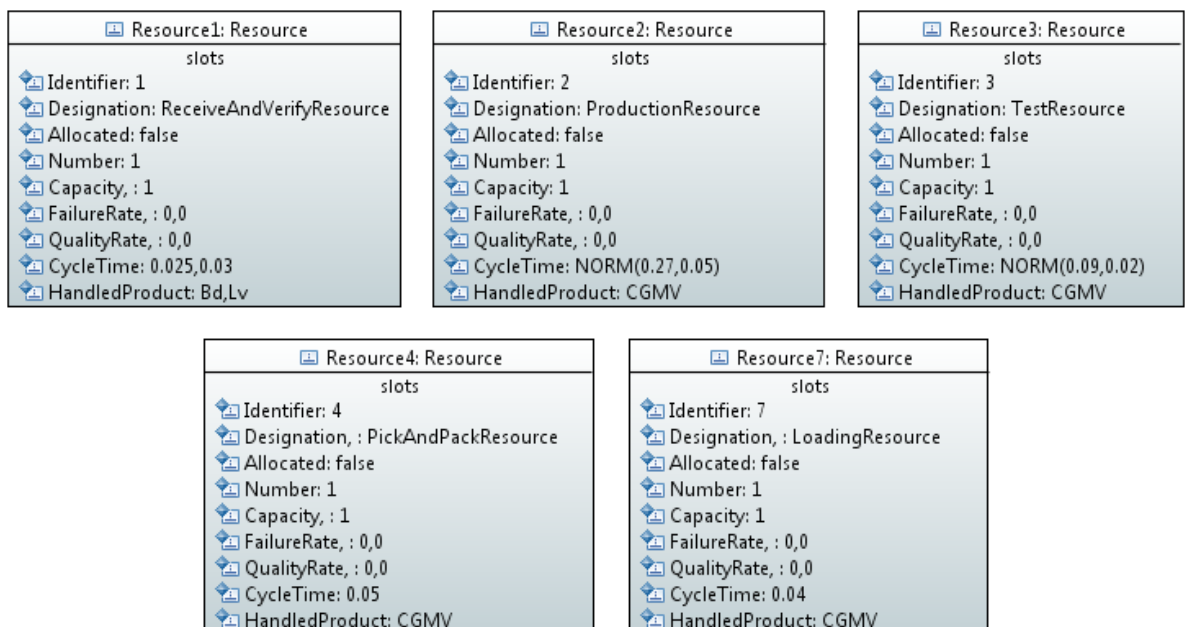


FIGURE A4.11: THE RESOURCE INSTANCES DIAGRAM BELONGING TO GTM FACILITY

The declared Resource for the Facility DistC1 shown in Figure A4.12 is the one used for the reception and the verification of sourced trucks.



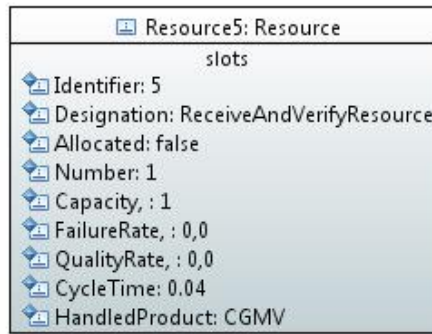


FIGURE A4.12: THE RESOURCE INSTANCES DIAGRAM BELONGING TO DISTC1 FACILITY

The declared Resource for the Facility DistC2 shown in Figure A4.13 is the one used for the reception and the verification of sourced trucks.



FIGURE A4.13: THE RESOURCE INSTANCES DIAGRAM BELONGING TO DISTC2 FACILITY

The declared Resource for the Facility of the Supplier1 shown in Figure A4.14 is the one used for loading vehicles with manufactured trucks.

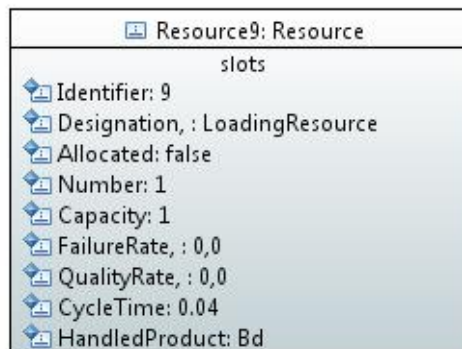


FIGURE A4.14: THE RESOURCE INSTANCES DIAGRAM BELONGING TO SUPPLIER1 FACILITY

The declared Resource for the Facility of the Supplier2 shown in Figure A4.15 is the one used for loading vehicles with manufactured trucks.



FIGURE A4.15: THE RESOURCE INSTANCES DIAGRAM BELONGING TO SUPPLIER2 FACILITY

### A4.1.3.3 MODELING THE SC BUFFERS

The declared Buffer's instances for the GTM Facility, shown in Figure A4.16, are as follow: The Buffer1 is used for storing the received subassemblies, the Buffer2 is used for storing the verified good subassemblies, the Buffer3 is used for storing the verified defective subassemblies, the Buffer4 is used for storing the issued subassemblies for production, , the Buffer5 is used for storing the manufactured trucks, the Buffer 6 is used for storing the good trucks, the Buffer7 is used for storing the defective Trucks and the Buffer8 is used for storing the picked and packed trucks.



FIGURE A4.16: THE BUFFER INSTANCE BELONGING TO GTM FACILITY

The declared Buffer's instances for the DistC1 Facility, shown in Figure A4.17, are as follow: The Buffer11 is used for storing the received trucks, the Buffer12 is used for storing the verified Trucks qualified as good and the Buffer13 is used for storing the verified trucks qualified as defective.

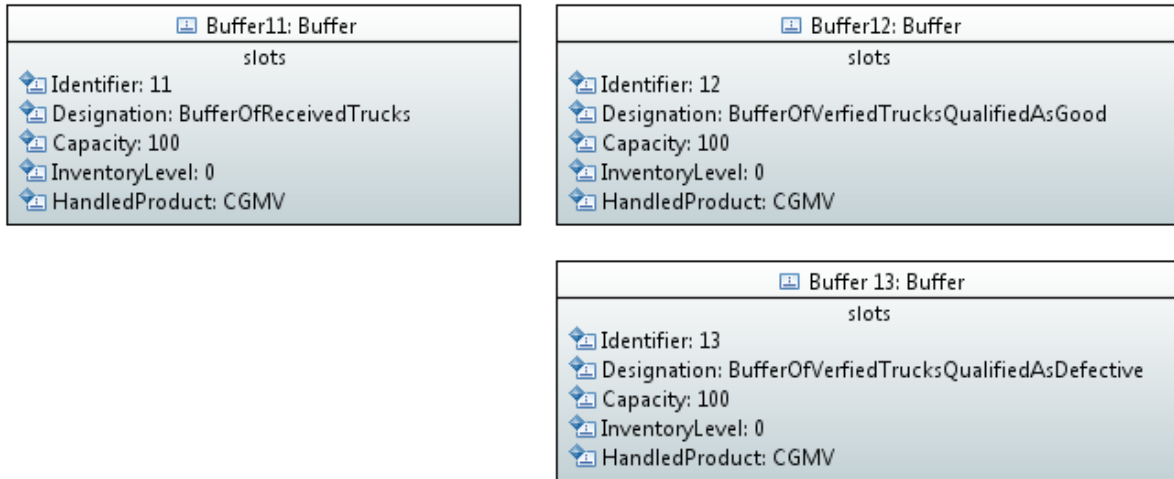


FIGURE A4.17: THE BUFFER INSTANCE BELONGING TO DISTC1 FACILITY

The declared Buffer's instances for the DistC2 Facility, shown in Figure A4.18, are as follow: The Buffer14 is used for storing the received trucks, the Buffer15 is used for storing the verified Trucks qualified as good and the Buffer16 is used for storing the verified trucks qualified as defective.

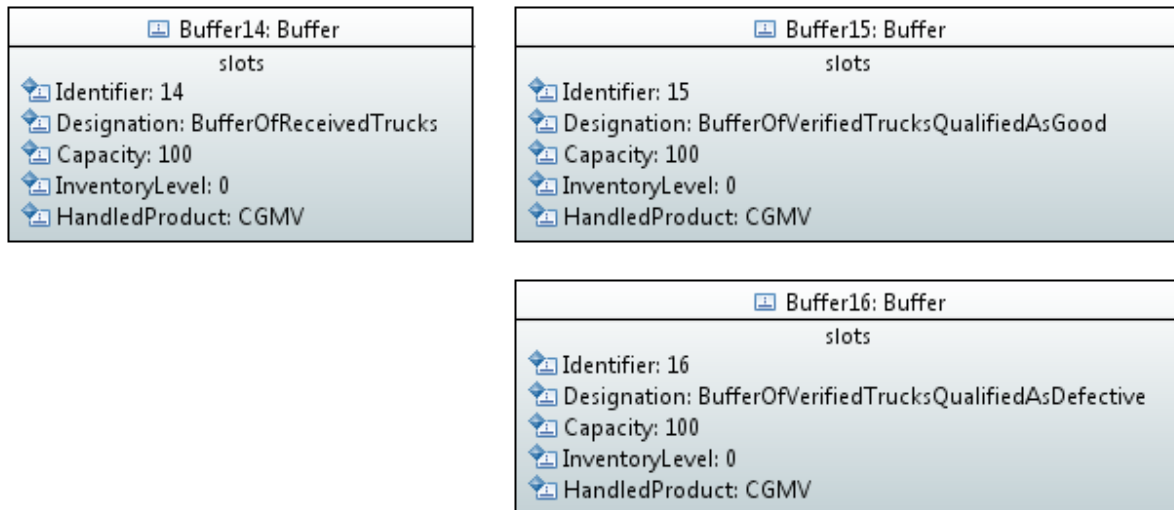


FIGURE A4.18: THE BUFFER INSTANCE BELONGING TO DISTC2 FACILITY

The declared Buffer instances Buffer9 for the supplier1 Facility, shown in Figure A4.19 is used for storing the subassemblies that are going to be loaded in vehicles.

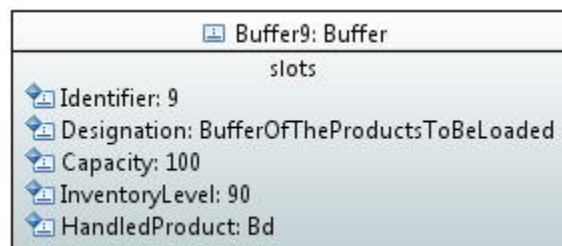


FIGURE A4.19: THE BUFFER INSTANCE BELONGING TO SUPPLIER1 FACILITY

The declared Buffer instances “Buffer10” for the supplier1 facility, shown in Figure A4.20 is used for storing the subassemblies that are going to be loaded in vehicles.

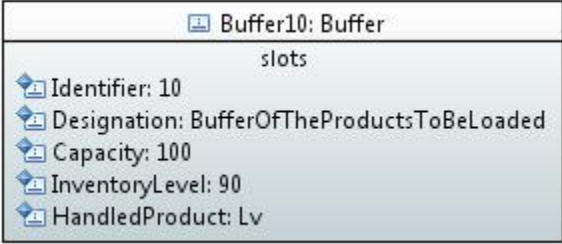


FIGURE A4.20: THE BUFFER INSTANCE BELONGING TO SUPPLIER2 FACILITY

A4.1.4 MODELING THE TRANSPORTATION VIEW OF THE SC

A4.1.4.1 MODELING THE SC ROUTES AND PATHS

The declared Route instances used for transporting manufactured trucks and subassemblies from and to the Facility GTM are shown in Figure A4.21. Namely, the Route1 instance is the one used for transporting the manufactured trucks from Grenoble to Paris, the Route2 is the one used for transporting the manufactured trucks from Grenoble to Spain, the Route3 instance is the one used for transporting the subassembly Bd from Tunisia to Grenoble, the Route4 instance is the one used for transporting the subassembly Lv from Morocco to Grenoble.

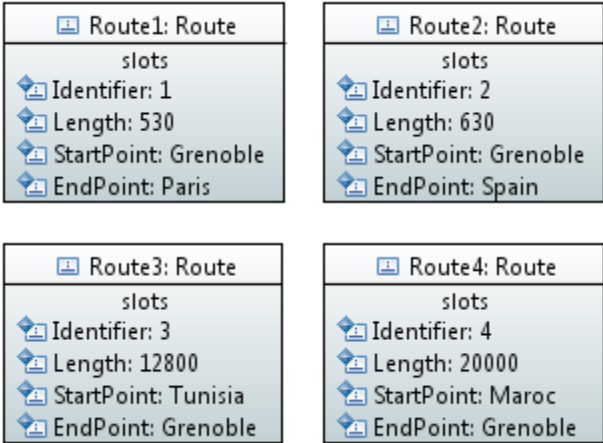


FIGURE A4.21: THE SC ROUTE INSTANCES

The declared path is used for transferring the sub-assemblies from the Buffer of the verified sub-assemblies to the input Buffer of the assembly Operation. The Path instance is shown in Figure A4.22.

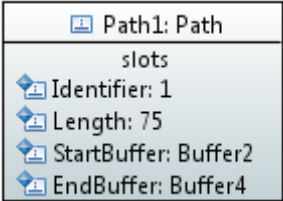


FIGURE A4.22: THE PATH INSTANCE BELONGING TO GTM

#### A4.1.4.2 MODELING THE SC TRANSPORTATIONRESOURCES AND TRANSFERRESOURCES

The declared Transportation Resource instances used for modeling the shipping vehicles are shown in Figure A4.23. Namely, the Resource14 is the Transportation Resource used for shipping the sub-assembly Bd from Tunisia to Grenoble. The Resource15 is the Transportation Resource used for shipping the sub-assembly Lv from Morocco to Grenoble. The Resource12 is the Transportation Resource used for shipping the trucks CGMV from Grenoble to Paris and from Grenoble to Spain.

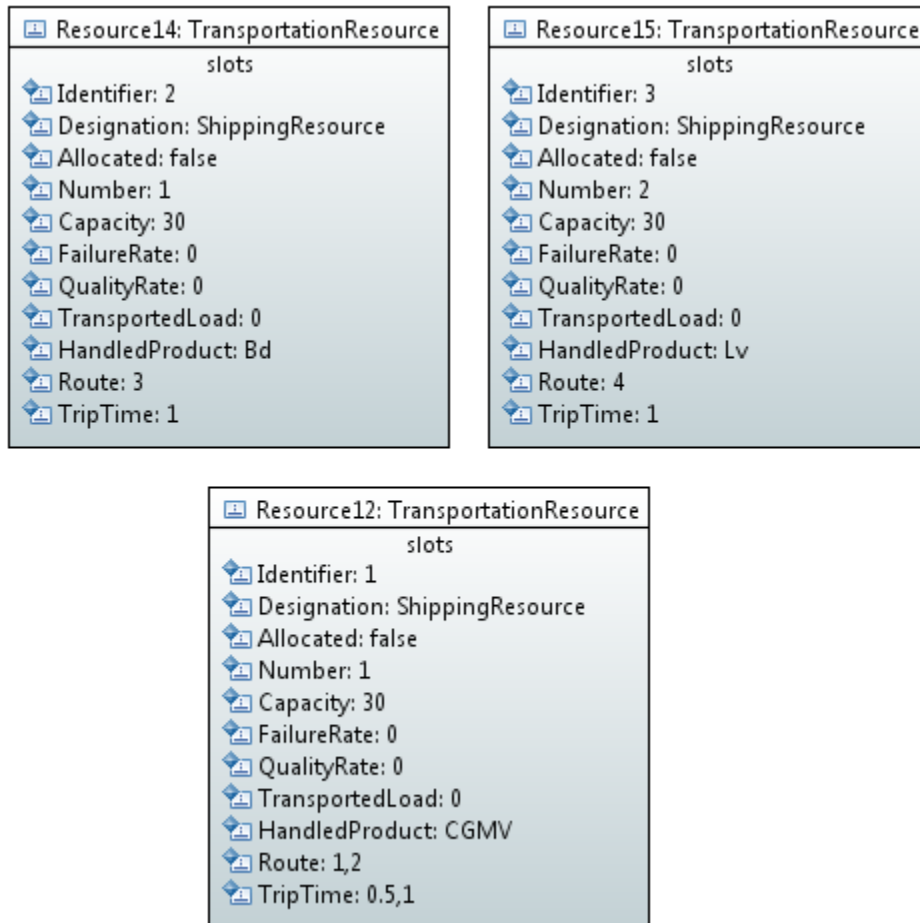


FIGURE A4.23: THE SC TRANSPORTATIONRESOURCE INSTANCES

The declared TransferResource instance used for modeling the unique transfer station of the SC is shown in Figure A4.24. Namely, the Resource14 is the Transfer Resource used for issuing the sub-assembly Bd and Lv from the Buffer of verified good subassemblies to the input Buffer of the production Operation.



FIGURE A4.24: THE SC TRANSFERRESOURCE INSTANCES

## A4.2 MODELING THE SC BEHAVIOR

### A4.2.1 MODELING THE SC ACTIVITIES

To create the model of the SC activities, we first start by instantiating the SC Roles and then instantiating the Operation instances that fit with the SC functions. Hence, we instantiate the Roles shown in Figure A4.25 for Truck-Much.

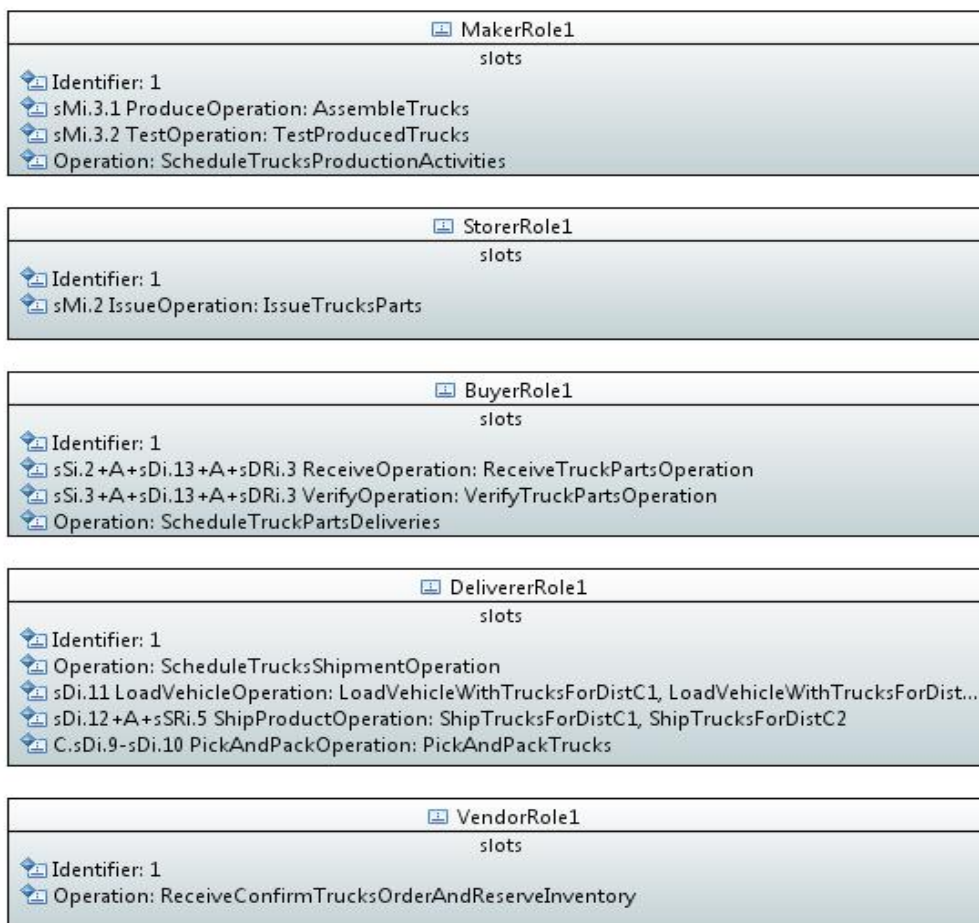


FIGURE A4.25: THE INSTANCES OF THE ROLES CONSTRUCTS RELATIVE TO TRUCKMUCH

The instantiated Roles for Supplier1 are shown in Figure A4.26. We are only interested in the reception of “Purchase Order”, the preparation and the shipping of trucks bodies to final customers.

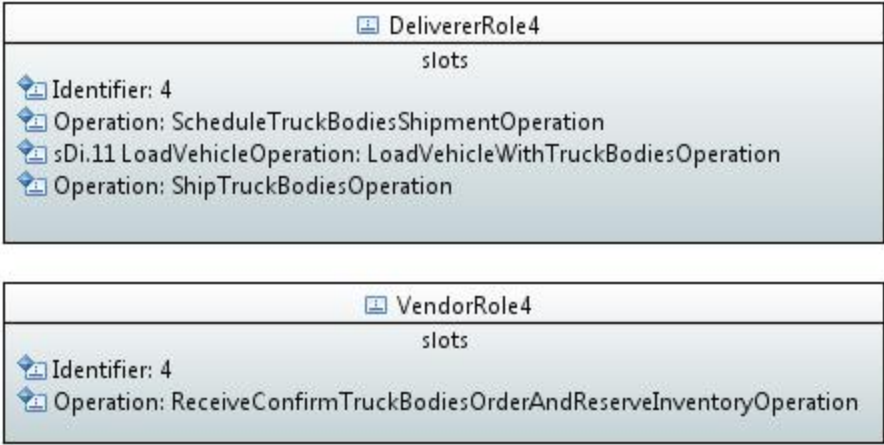


FIGURE A4.26: THE INSTANCES OF THE ROLES CONSTRUCTS RELATIVE TO SUPPLIER1

The instantiated Roles for Supplier2 are shown in Figure A4.27. We are only interested in the reception of “Purchase Order”, the preparation and the shipping of trucks levers to final customers.

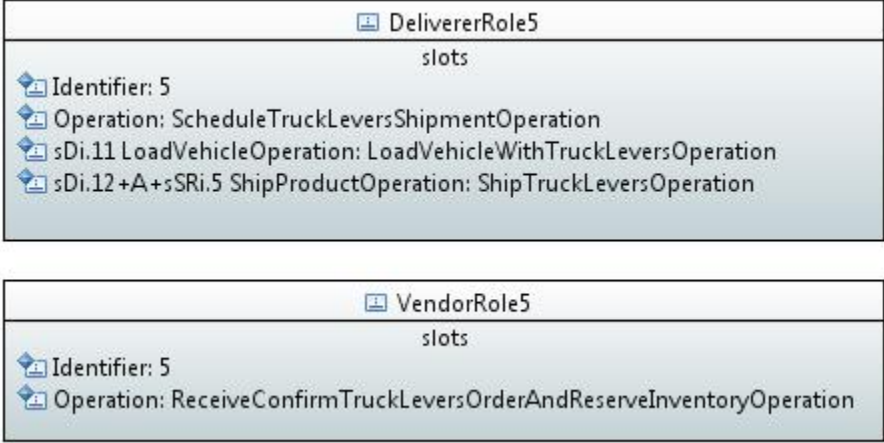


FIGURE A4.27: THE INSTANCES OF THE ROLES CONSTRUCTS RELATIVE TO SUPPLIER2

The instantiated Roles for DistC1 are shown in Figure A4.28.



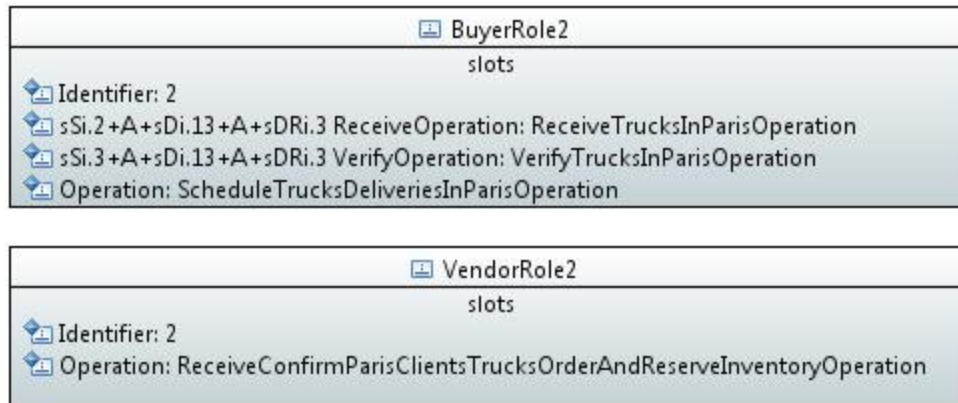


FIGURE A4.28: THE INSTANCES OF THE ROLES CONSTRUCTS RELATIVE TO DISTC1

The instantiated Roles for DistC2 are shown in Figure A4.29.

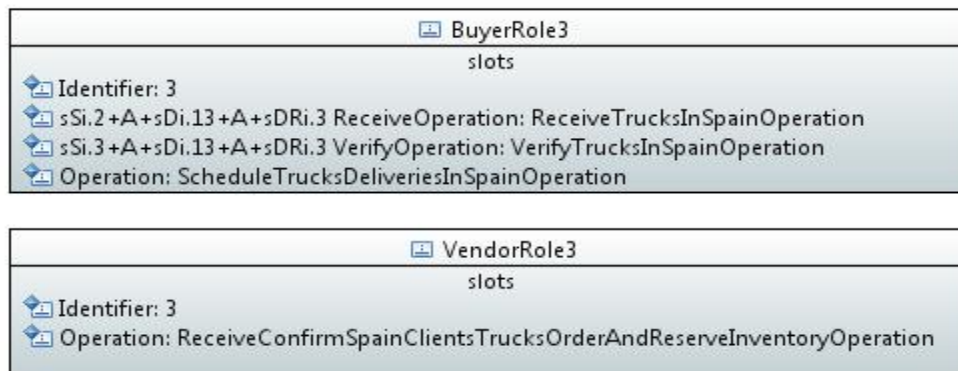


FIGURE A4.29: THE INSTANCES OF THE ROLES CONSTRUCTS RELATIVE TO DISTC2

After naming the Operation instances for each Role of the SC member, the next thing to do is to customize each Operation instance by setting its parameters with the values of the variables representing the SC structure elements.

#### A4.2.2 MODELING THE SC PROCESSES

In this step, we link the instantiated Operation instances of the SC Actors to form the SC Process. This is done through creating an activity diagram that specifies the Operations organization and the exchanged Flows.

Hence, we create an aggregated view of the SC Process that is shown in Figure A4.30.

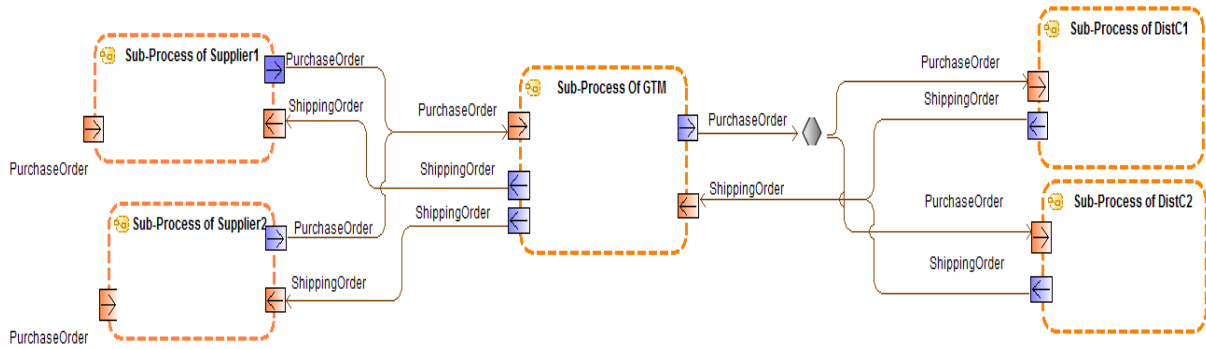


FIGURE A4.30: THE ACTIVITY DIAGRAM MODELING THE AGGREGATED SC PROCESS OF THE CASE STUDY

The sub-process of each Facility is presented using an activity node in the activity diagram of the figure A4.30.

The sub-process of each Facility is presented in a separated activity diagram. Hence, the activity diagrams for the Facilities of GTM, Supplier1, Supplier2, DistC1, and DistC2 are shown respectively in the figure A4.31, the figure A4.32, the figure A4.33, the figure A4.34 and the figure A4.35.

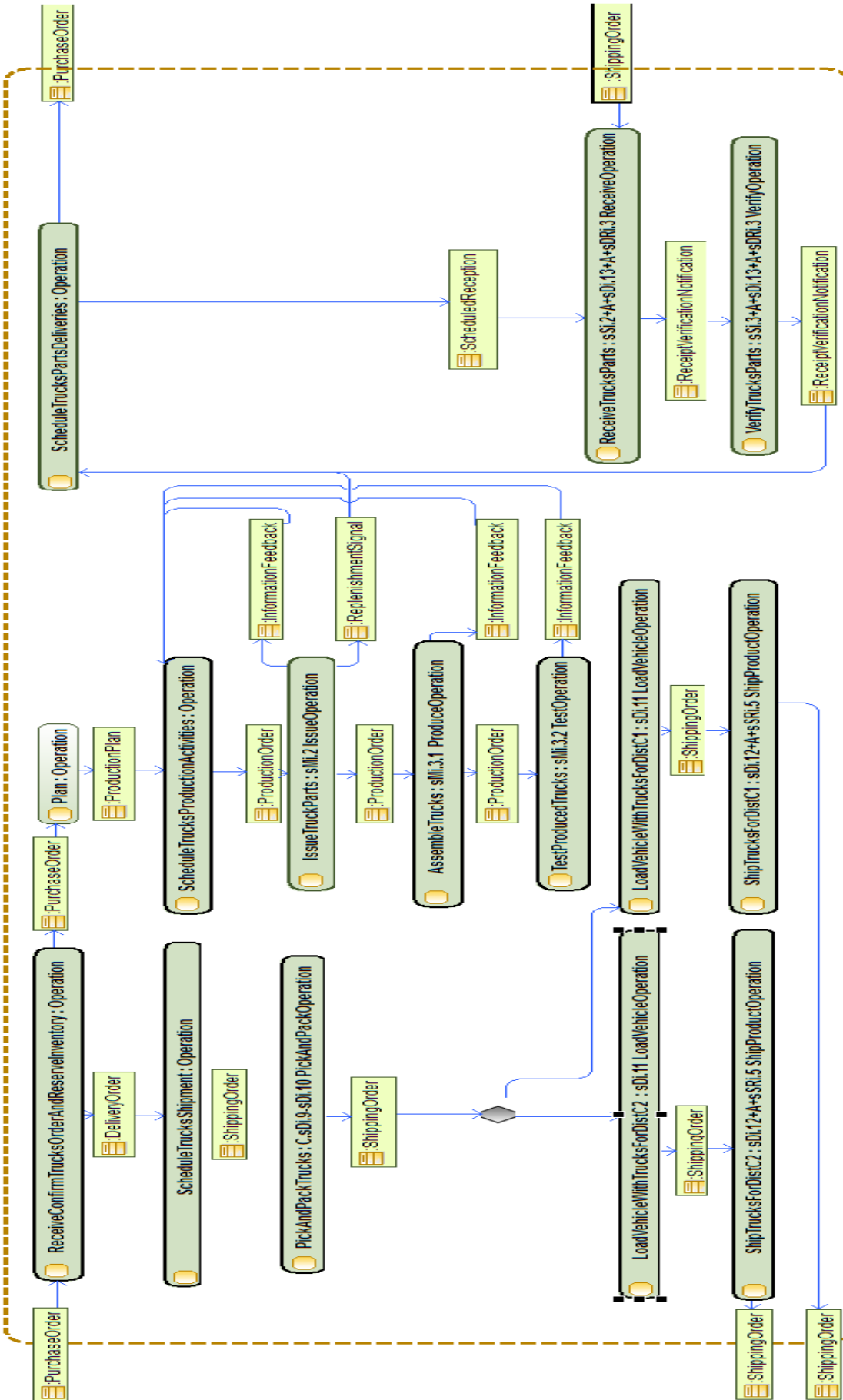


FIGURE A4.31: THE ACTIVITY DIAGRAM MODELING THE SUB-PROCESS OF GTM

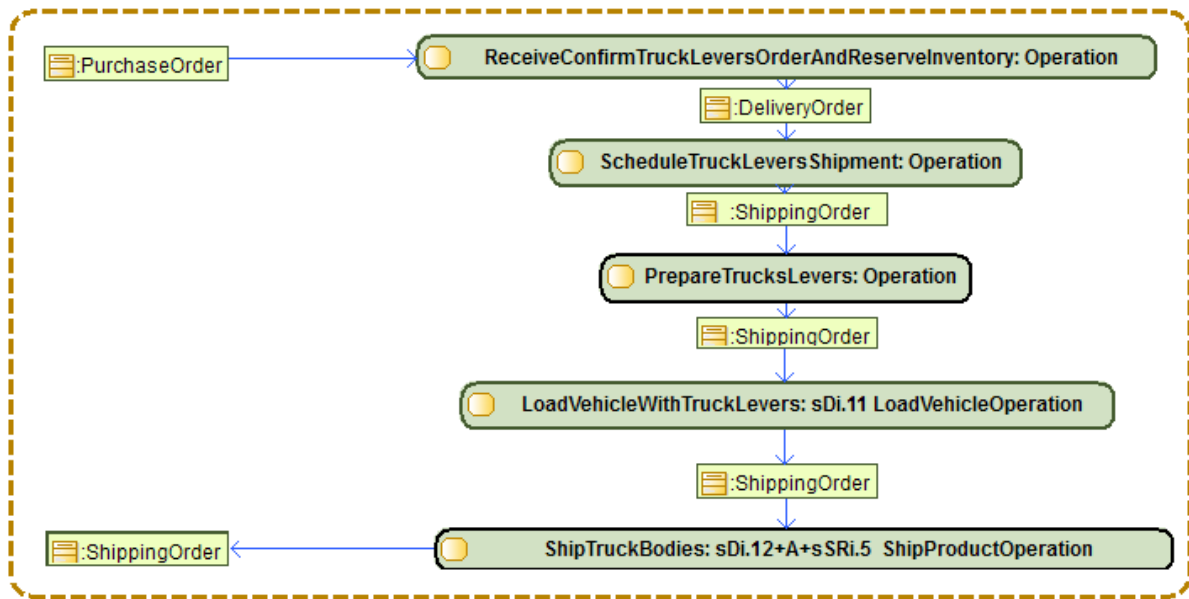


FIGURE A4.32: THE ACTIVITY DIAGRAM MODELING THE SUB-PROCESS OF SUPPLIER1

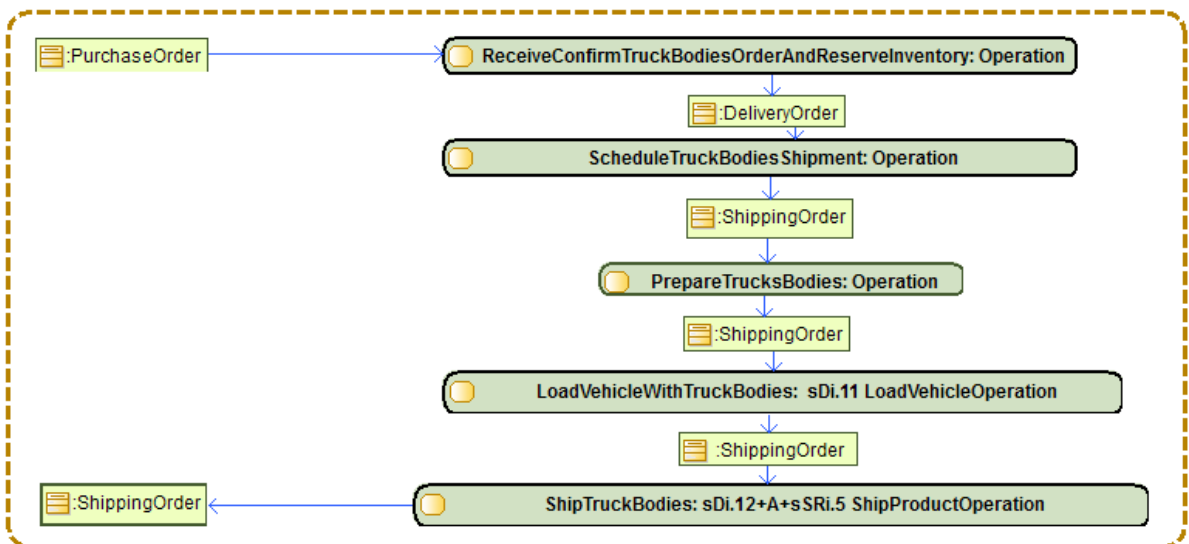


FIGURE A4.33: THE ACTIVITY DIAGRAM MODELING THE SUB-PROCESS OF SUPPLIER2

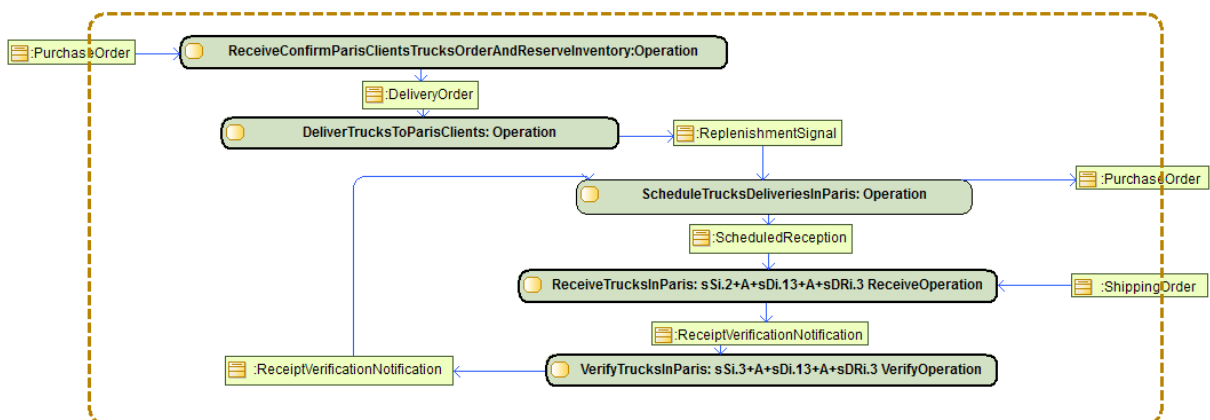


FIGURE A4.34: THE ACTIVITY DIAGRAM MODELING THE SUB-PROCESS OF DISTC1

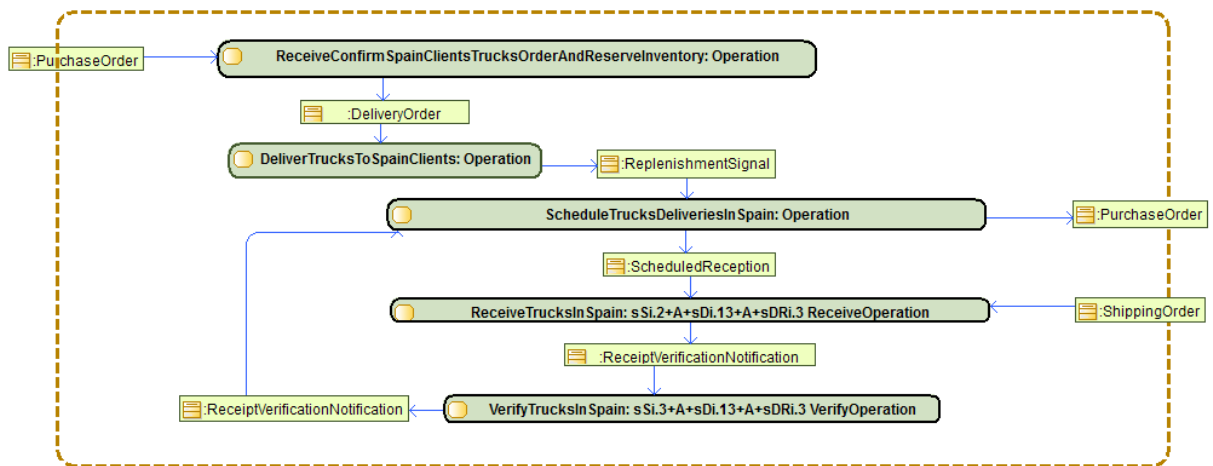


FIGURE A4.35: THE ACTIVITY DIAGRAM MODELING THE SUB-PROCESS OF DISTC2

---

## RESUME

---

La maîtrise des risques est un enjeu majeur pour les entreprises. Loin d'être l'apanage des seules catastrophes naturelles, les perturbations des chaînes logistiques actuelles peuvent parfois être causées par des événements mineurs amplifiés par les failles d'organisations industrielles de plus en plus complexes. Nombreux sont les exemples de ces perturbations avec des conséquences économiques graves.

La gestion des risques dans les chaînes logistiques est un thème récent et les méthodes et outils actuels ne répondent pas encore totalement aux préoccupations des gestionnaires de ces chaînes logistiques. Une grande aide peut être apportée par la simulation des événements affectant les chaînes. Cependant malgré son efficacité pour couvrir la complexité de la chaîne, la simulation reste encore difficile à mettre en œuvre, notamment dans les phases de création et d'exploitation des modèles.

Le but de cette thèse est de faciliter l'utilisation de la simulation pour l'analyse des risques dans les chaînes logistiques. Ainsi, nous avons développé un référentiel de modélisation pour la simulation qui permet d'assurer une construction facile des modèles de la structure, du comportement et des risques inhérents aux chaînes logistiques. Ce référentiel est bâti sur un ensemble de métamodèles et de bibliothèques adaptés à la définition de chaînes logistiques et définis sur la base du référentiel SCOR. Ajouté à cela, nous avons proposé un guide de traduction permettant le passage d'un modèle conceptuel de chaîne logistique vers un modèle de simulation permettant de tester les scénarios de risque. Une bibliothèque de modules de simulation a été proposée pour accompagner ce passage. Une étude de cas a été menée pour tester et valider partiellement l'approche proposée.

Mots clés: Gestion des risques, Analyse des chaînes logistiques, Méta-modélisation, Simulation

## ABSTRACT

---

Controlling risks is an important issue for companies. Far from being only the prerogative of natural disasters, the disruptions of today's supply chains can sometimes be caused by minor events amplified by the flaws of increasingly complex industrial organizations, causing severe economic losses.

Risk management in supply chains is a recent theme and the proposed solutions are not yet able to meet the needs of practitioners. One of the solutions to analyse risks is using simulation. But, despite its effectiveness to cover the complexity of the chain, it still presents a major weakness which is the difficulty of implementation.

The aim of this thesis is to facilitate and to adapt the simulation for risk analysis of supply chains. Thus, we have developed a modeling framework for simulation which enables an easy construction of models of supply chain structure, behavior and if the associated risks. This is done through the proposition of a set of meta-models and libraries, defined on the basis of the SCOR reference model. In addition, we proposed a translation guide for the translation of the conceptual model of supply chains into a simulation model and enabling testing risk scenario. Additionally, we developed a library of simulation modules.

A case study was conducted and the results show the relevance of the proposed approach.

Key words: Risk management, Supply Chain analysis, Meta-modeling, Simulation.