A SERVICE ORIENTED COLLABORATIVE SUPPLY CHAIN PLANNING PROCESS DEFINITION AND EXECUTION PLATFORM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MEHMET OLDUZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

SEPTEMBER 2008

Approval of the thesis

"**A SERVICE ORIENTED COLLABORATIVE SUPPLY CHAIN PLANNING PROCESS DEFINITION AND EXECUTION PLATFORM**"

submitted by **Mehmet Olduz** in partial fullfillment of the requirements for the degree of **Master of Science in Computer Engineering, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, **Graduate School of Natural and Applied Sciences**

Prof. Dr. Volkan Atalay _____
Head of Department, **Computer Engineering**

Prof. Dr. Asuman Doğaç _____
Supervisor, **Department of Computer Engineering, METU**

Examining Committee Members:

Prof. Dr. İsmail Hakkı Toroslu _____
Department of Computer Engineering, METU

Prof. Dr. Asuman Doğaç _____
Department of Computer Engineering, METU

Assoc. Prof. Dr. Nihan Kesim Çiçekli _____
Department of Computer Engineering, METU

Assoc. Prof. Dr. Ahmet Coşar _____
Department of Computer Engineering, METU

Dr. Gökçe Banu Laleci Ertürkmen _____
SRDC Ltd.

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Mehmet Olduz

Signature :

ABSTRACT

A SERVICE ORIENTED COLLABORATIVE SUPPLY CHAIN PLANNING PROCESS DEFINITION AND EXECUTION PLATFORM

Olduz, Mehmet

M.S., Department of Computer Engineering

Supervisor: Prof. Dr. Asuman Doğaç

September 2008, 109 pages

Currently, there are many software applications handling planning, scheduling, material management, invoicing, workflow management within an organization. However, companies need to plan across a wider span of activities and need to collaborate with their partners to optimize the "overall" profitability. This requires collaborative planning within a supply chain and exchange of planning data. Collaborative Planning, Forecast and Replenishment (CPFR®) is one of the most prominent initiatives on Collaborative Planning. However, CPFR only provides guidelines, but does not mandate any technology for the definition and execution of planning process. Therefore, companies have difficulties to define and deploy CPFR solutions and there is a need for a Service Oriented, Open Platform for the definition and execution of collaborative planning processes involving many supply chain tiers.

In this work, first of all, the building blocks of the planning process have been defined as machine processable definitions in OASIS ebXML Business Specification Language (ebBP). CPFR Designer Tool developed provides the users to visually create CPFR Processes in ebBP and to convert this ebBP process definition automatically to an executable business process using OASIS Business Process Execution Language (WS-BPEL). In this way, the supply chain enterprises are able to create customized CPFR processes which are in integration with the underlying intra-enterprise planning processes. Moreover, in the thesis, a CPFR Process Execution Environment is prepared where the generated CPFR Process can be

executed.

The work presented in this thesis is realized as a part of IST-213031 iSURF project funded by European Commission under ICT FP7.

Keywords: CPFR®Guidelines, Supply Chain Management, Collaborative Planning, Business Processes, Web Services, WS-BPEL, ebXML

ÖZ

SERVİS TABANLI ORTAK TEDARİK ZİNCİRİ PLANLAMA SÜRECİ TANIMLAMA VE YÜRÜTME PLATOFORMU

Olduz, Mehmet

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Asuman Doğaç

Eylül 2008, 109 sayfa

Halihazırda şirketlerin planlama, zamanlama, malzeme yönetimi, faturalandırma ve iş akışı yönetimi gibi amaçlar için kullanmakta olduğu çok çeşitli uygulamalar mevcut. Fakat, karlılıklarını ve ortaklıklarını artırmak için şirketler çok daha geniş çerçevede planlama ve iş birliği yapma ihtiyacı duyuyorlar. Bu durum ise bir tedarik zinciri içinde ortak planlama ve planlama bilgisinin ortaklar arasında paylaşılmasını gerekli kılıyor. Bu ihtiyaca yönelik olarak, Ortak Planlama, Tahmin ve İkmal (CPFR®), ortak planlama alanındaki en ileri gelen girişimlerden biridir. Ancak, CPFR sadece yönergeler sağlıyorken, planlama sürecinin tanımlanması ve yürütülmesi için herhangi bir teknolojiyi zorunlu kılmıyor. Dolayısıyla, şirketler CPFR çözümleri tanımlamada ve konuşlandırmada sorunlar yaşıyorlar ve gerekli tedarik zinciri katmanlarını içeren ortak planlama süreçlerinin tanımlanması ve yürütülmesi için servise dayalı, açık bir platforma ihtiyaç var.

Bu çalışma kapsamında, öncelikle planlama süreçlerinin yapı taşları, OASIS ebXML İş Tanımlama Dili (ebBP) ile bilgisayarca işlenebilen tanımlar halinde ifade edildi. Geliştirilen CPFR Tasarı Yazılımı şirketlere görsel olarak CPFR Sürecini ebBP dilinde tanımlama ve bu tanımı otomatik olarak çalıştırılabilir bir iş süreci olan OASIS İş Süreci Yürütme Dili (WS-BPEL) tanımına çevirebilme olanağı sağlamaktadır. Bu sayede tedarik zinciri işletmeleri tabandaki şirket-içi planlama süreçleri ile entegre olmuş CPRF süreçlerini kolayca oluştura-

bileceklerdir. Ayrıca, yine tez kapsamında oluşturulan CPFR süreçlerinin çalıştırabilmesi için bir CPFR Süreç Yürütme Ortamı hazırlanmıştır.

Tezde sunulan bu çalışma bir Avrupa Komisyonu ICT FP7 projesi olan IST-213031 iSURF projesinin bir parçası olarak gerçekleştirilmiştir.

Anahtar Kelimeler: CPFR®Yönergeleri, Tedarik Zinciri Yönetimi, Ortak Planlama, İş Süreçleri, Web Servisleri, WS-BPEL, ebXML

ACKNOWLEDGMENTS

First of all, I am honored to express my sincere gratitude and special thanks to my supervisor Prof. Dr. Asuman Doğaç for all her guidance and support in this study.

I would also like to convey thanks to jury members for their valuable comments on this thesis.

I would also like to extend my deep appreciation to Dr. Gökçe Banu Laleci Ertürkmen for her continuous encouragement and enormous support throughout this thesis. My special thanks are extended to Alper Okcan, Mazhar Tekin and all other members at Software Research and Development Center team for their support and patience during this study.

I would like to specially thank to my friend Mustafa Yüksel for his help and support in the time of writing the thesis. My special thanks are extended to my friends Sercan Gök, Ahmet Topal and Kamil Tulum for their motivation and support during this study.

Last but not the least; I wish to present my deepest gratefulness to my parents and to my brother Ali Olduz for all their unconditional love, motivation and life-long support. Without them, this work could not have been completed.

To my family

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

APS	Advanced Planning and Scheduling
ASP	Active Server Pages
BPEL	Business Process Execution Language
BPM	Business Process Management
BTA	Business Transaction Activity
CORBA	Common Object Requesting Broker Architecture
CPFR	Collaborative Planning, Forecasting and Replenishment
CRM	Customer Relationship Management
DC	Distribution Center
DRP	Distribution Requirements Planning
EAN	European Article Number
ebBP	OASIS ebXML Business Process Specification Schema
ebXML	Electronic Business XML
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
FP7	Seventh Framework Programme
GDSSU	Global Data Synchronization Service Utility
GS1	Global Standards One
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
ICT	Information Communication Technologies
ISU	Interoperability Service Utility
iSURF	An Interoperability Service Utility for Collaborative Supply Chain Planning across Multiple Domains Supported by RFID Devices
JEITA	Japan Electronics and Information Technology Industries Association
JSP	Java Server Pages
MES	Message Exchange States
MRP	Material Requirements Planning
OASIS	Organization for the Advancement of Structured Information Standards
PDD	Process Deployment Descriptor
POS	Point-of-Sale
RFID	Radio Frequency Identification
SME	Small and Medium Enterprises
SOAP	Simple Object Access Protocol
SSL	Secure Sockets Layer
UCC	Uniform Commercial Code
UUID	Universally Unique Identifier
VICS	Voluntary Interindustry Commerce Solutions
VMI	Vendor Managed Inventory
W3C	World Wide Web Consortium

WMS Warehouse Management Systems

WS-BPEL Web Services Business Process Execution Language

WSDL Web Service Description Language

XML Extensible Markup Language

XSD XML Schema Definition

XSLT Extensible Stylesheet Language Transformations

CHAPTER 1

INTRODUCTION

In order to guarantee the survival in today's competitive and demanding digital world of business, the companies, especially SMEs, should be more agile, self-sustainable and responsive to the changes in the supply chain. Obtaining and maintaining a competitive edge in the supply chain is not only the concern of individual SMEs, but should be also addressed by the entire chain jointly. The supply chain partners should collaborate effectively so as to better align the supply and demand forecasts to have a joint strategy for handling the exceptions that will occur in the way of realizing the "the network is the business" vision.

The first prerequisite to have a collaborative supply chain planning is to share information on the supply chain visibility, individual sales and order forecast of companies, the current status of the products in the manufacturing and distribution process, and the exceptional events that may affect the forecasts in a secure and controlled way. Knowledge is the main instrument to drive and support collaboration: there needs to be a knowledge-oriented inter-enterprise collaboration between supply chain partners.

However, the collaborative inter-enterprise planning requires more than exchanging messages, there needs to be a joint planning process, defining when and how this information should be collected, and which application is responsible for assessing this information in order to create the joint supply chain forecasts, the replenishment and the exception management strategies.

Being aware of this need, "Collaborative Planning, Forecasting, and Replenishment (CPFR®)" guidelines [9] have been produced by Voluntary Interindustry Commerce Standards (VICS) [41]. CPFR proposes steps to update the overall forecasts and the schedules based on conflicts with the internal plans and the schedules in an iterative manner between the two supply chain partners.

Promising an efficient collaboration in the supply chain, CPFR Guidelines have attracted

a considerable attention from the industry as it promises a significant amount of improvement on the efficiency of the supply chain management. According to a survey [10] performed by KJR Consulting in United States, 67% of the respondents (illustrated in the Figure 1.1) stated that they are currently involved in some CPFR Activity.

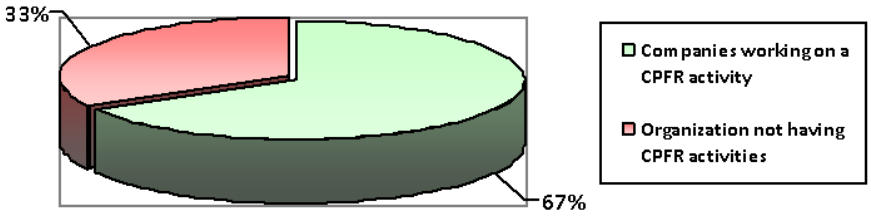

Figure 1.1: The status of CPFR Activities

The main reason behind this large participation in CPFR activities is possibly the expected benefits of implementing CPFR. In fact, the realized benefits of the CPFR reported by 21 companies participated in the survey of KJR Consulting is considerably promising as shown below in Figure 1.2.

Figure 1.2: Realized Benefits of CPFR

Among many pilot applications on CPFR, an example pilot application is the two phase

pilot between Intel Corporation and Shinko which was initiated to automate the forecast-to-cash procurement phase [35]. The second phase started in 2003 aims to automate the sharing of the forecasts through RosettaNet Standards, and the benefits have been reported as the reduction of days of inventory, the increased flexibility to meet the end customer's upside or downside requests, the quicker response, resulting in the forecast-related lead time reduction of up to 47%. The Vice-President, Intel Manufacturing Group, has reported that "*Making business decisions with good, hard data from the entire business environment instead of just local ERP (Enterprise Resource Planning) systems allows a business to anticipate changes in the business climate much sooner and to respond to those changes with greater agility*".

With the collaborative technologies first emerging in 2000, many early adopters from a variety of industries began pilot projects for the collaborative supply chain process. These companies were rewarded by the enhanced revenue, the greater cost efficiencies and improvements in inventory planning, however also reported that the design and the deployment processes were labor-intensive. Despite the hype of successful pilot projects, the collaborative forecasting remains in the early adoption stage. The SMEs typically have small IT budgets or crude process standards, with little visibility data, or data integrity to enable them to share and compare with the trading partners.

Moreover, in the survey of KJR Consulting mentioned above it is stated that:

"*The interviews highlight a common frustration with technology and the pressing need for technology vendors to create/support software that fully automates the CPFR process - a process that is interoperable among all players. Respondents agree that the key to scalability depends on achieving this technological vision. In interviews, the majority of companies noted a fear that their trading partners would specify the use of proprietary solutions and/or non-compatible systems. It is clear to respondents that the cost of participation in a non-interoperable world would out-weigh the apparent benefits promised by CPFR.*" [10].

Considering these responses from the industry, it can be concluded that there is a gap in the technology side of the CPFR adoption. As mentioned in [12] CPFR systems stand on their own, but must interact with the existing supply and demand chain applications in an enterprise and the points of contact depend upon whether a company is a retailer or supplier. For these reasons also, currently, the companies are facing difficulties in understanding and developing the underlying systems that will support the CPFR Process. Therefore, the effects of the envisioned significant effects of CPFR guidelines on the supply chain will be dramatic only when it is completely integrated with the demand plan of an enterprise, in which the production cycle is synchronized with CPFR order cycle.

In order to address these problems, in this thesis, CPFR building blocks have been defined in a standard, machine processable business process specification language - OASIS ebXML Business Specification Language (ebBP) [19]- a CPFR Designer Tool and a CPFR Process Execution Environment have been developed. The CPFR Designer Tool allows generating the joint inter-enterprise collaboration processes by customizing the CPFR Process templates which are composed of the CPFR building blocks defined in the ebBP Specification. Although the generated process definition of the collaboration is not an executable process definition, the collaboration definition can be further processed by the computer as it is machine processable. The CPFR Designer Tool developed within this thesis can further convert these definitions to an executable business process definition, Web Services Business Process Execution Language (WS-BPEL) [18], so that the CPFR process built can be enacted among the different supply chain partners. By automatically generating the deployment package of the collaborative business process definition for an open source BPEL engine, ActiveBPEL [3], the CPFR process specialized to the specific needs of the collaborating partners can be executed without involving a technologic burden on the companies, especially on the SMEs.

This thesis has been realized as a part of the iSURF project [29] (An Interoperability Service Utility for Collaborative Supply Chain Planning across Multiple Domains Supported by RFID Devices) supported by European Commission Information Communication Technologies (ICT) Seventh Framework Programme (FP7) [24]. The iSURF project aims to develop a collaborative supply chain planning environment based on CPFR guidelines addressing the interoperability challenges of deploying a CPFR process within a supply chain consortium. The iSURF general architecture is presented in Figure 1.3. The thesis research corresponds to the iSURF Service Oriented Supply Chain Planning Process Definition and Execution Environment which is highlighted in the figure below. The aim of this component within the iSURF project is to enable the definition and execution of inter-enterprise collaboration across a wide variety of business domains through a Service Oriented Collaborative Supply Chain Planning Process Definition and Execution Platform.

In the iSURF architecture, the interaction with legacy planning applications is achieved through the semantically enriched Web services which are implemented as legacy adapters. The interoperability of the business documents exchanged within the scope of this planning process is addressed by the iSURF Interoperability Service Utility (iSURF ISU). Finally, the supply chain visibility data is collected through a smart product architecture implemented based on the EPCGlobal [20] guidelines, and master data synchronization is achieved through

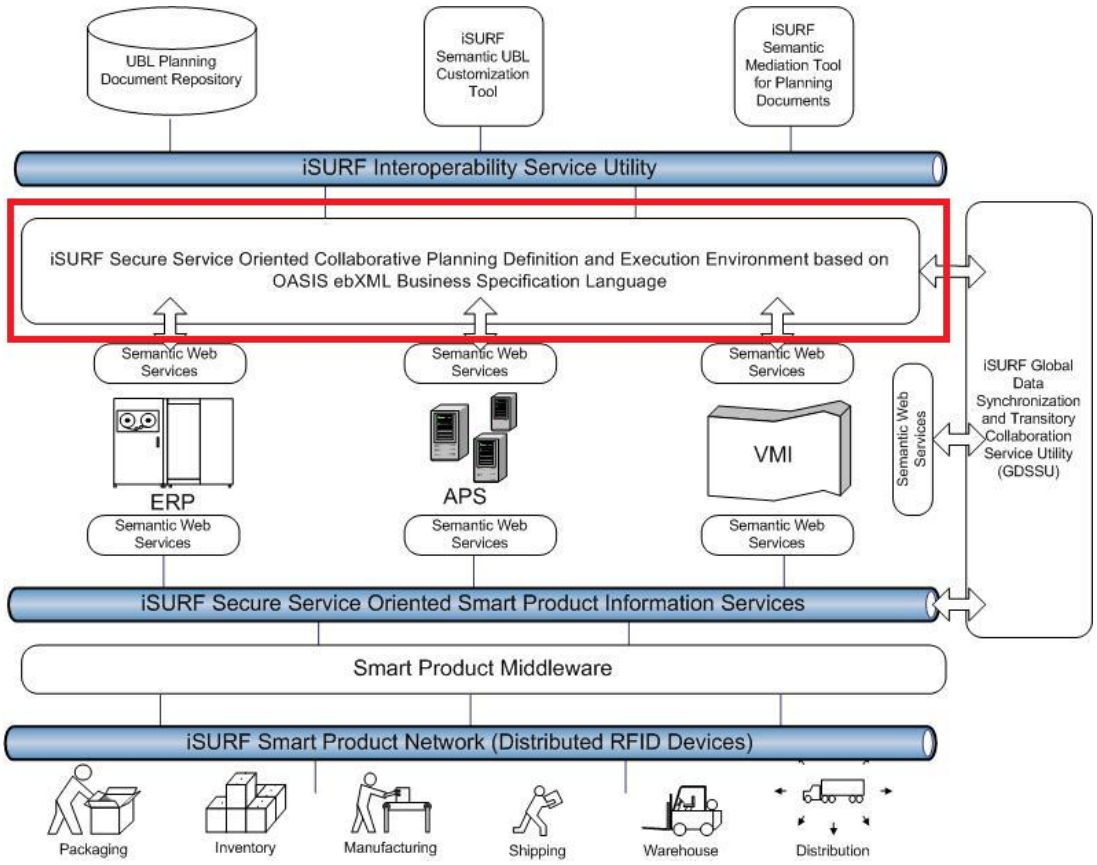

Figure 1.3: iSURF General Architecture

the Global Data Synchronization Service Utility (GDSSU).

As it can be seen in Figure 1.3, the Collaborative Planning Definition and Execution Environment has direct interface with the Semantic Web Services exposed over the systems of the companies, iSURF GDSSU and iSURF ISU. The web services are used for communicating with the existing applications of the partners like Enterprise Resource Planning (ERP), Advanced Planning and Scheduling (APS) or Vendor Managed Inventory (VMI). The objectives of the other two components are as follows:

- The iSURF ISU platform aims to enable the exchange of the planning data between enterprises and especially across the domains. In the iSURF project, the interoperability is regarded as a service that users can utilize anywhere and anytime they need. Therefore, iSURF ISU is projected to be a highly dynamic and flexible building block of the system which is envisioned to robustly respond to the user requests. The architecture of iSURF ISU is explained in [14].

- The iSURF GDSSU ensures the accuracy and reliability of master data used in the

supply chain by developing standard based open platform for the SMEs. While developing the system, the Global Standards One (GS1) [26] Global Data Synchronization Network (GDSN) [25] standards are used. GDSN is a network that connects data pools, which are regional sources of manufacturer and retailer data, to the GS1 Global Registry. The GS1 Global Registry lets companies locate the source or the recipient data pools so that data is standardized and synchronized for trading partners on a near real-time basis.

The CPFR Designer Tool and the CPFR Process Execution Environment will be in communication with these other iSURF components, namely: iSURF ISU, iSURF GDSSU and iSURF Web Services wrapping legacy enterprise systems.

During the execution of the CPFR process, the CPFR Process Execution Environment will access to iSURF ISU for the translation of the documents between the formats of the participating companies if they are different. iSURF GDSSU will be used if an irresolvable exception is generated by one of the collaborating companies. In this case, another partner will be searched and suggested to the company which wants to continue on business. Finally, the CPFR Designer tool accesses the list of services during the design of CPFR and CPFR Process Execution Environment will call those web services during the execution of CPFR Process.

This thesis is organized as follows: Chapter 2 summarizes the background on the enabling technologies and standards. In Chapter 3, the building blocks we propose for machine processable representation of CPFR Processes are presented. The implementation of the CPFR Designer Tool is presented in Chapter 4. The generation of the WS-BPEL definition and a CPFR Process Execution Environment is described in Chapter 5. In Chapter 6, the related work is presented on CPFR and on the business processes. Finally, Chapter 7 concludes this thesis and presents the future work.

CHAPTER 2

BACKGROUNG ON ENABLING TECHNOLOGIES AND STANDARDS

2.1 Collaborative Planning, Forecasting, and Replenishment - CPFR®

2.1.1 Introduction

Collaborative Planning, Forecasting, and Replenishment (CPFR®) guidelines formalize the processes between two trading partners used to agree upon a joint plan and forecast, monitor success through replenishment, and recognize and respond to any exceptions [2].

The main idea behind performing the planning and forecast jointly comes from the fact that trading partners have different competencies based on their strategies and investments. Also, the trading partners have different sources of information and different views of the market. By sharing plan and forecast data, the intelligence of multiple trading partners can be combined in the planning and fulfillment of customer demand. The main objective of CPFR is to increase the accuracy of demand forecasts and replenishment plans, necessary to lower inventories across the supply chain and attain high service levels of the right products in right locations [1]. As a result, trading partners would benefit from increased sales.

The main benefits CPFR guidelines propose to provide to its implementers in the collaboration can be listed as [39]:

- Greater visibility to improve replenishment accuracy
- Out-of-stock reduction
- Overstock reduction

- Production capacity aligned to meet customer demand
- Presentation stock
- Assortment optimization

Voluntary Interindustry Commerce Solutions (VICS) [41] reported that since the publication of CPFR guidelines in 1998, over 300 companies have implemented the process. Numerous case studies of CPFR projects document in-stock percentage improvements of from 2-8% for products in stores, accompanied by inventory reductions of 10-40% across the supply chain [39].

Syncra Systems and Industry Directions also conducted a survey of manufacturers, retailers, distributors, logistics providers and others [1]. According to this survey the report of this survey, the following benefits have been observed:

- An 80% increase in business with a CPFR partner
- $9M increase in sales
- Simultaneous sales growth and inventory reductions of at least 10%
- Improved fill rates with less inventory
- 100% service level with almost 40 inventory turns a year

2.1.2 Reference Model

CPFR guideline provides a reference model for the collaborative aspects of planning, forecasting and replenishment processes. This generic reference model is illustrated in Figure 2.1 which can be applied to many industries.

According to the model, the seller and the buyer employ four main activities in order to improve the overall performance of the supply chain [39]:

- *Strategy and Planning phase* establishes the ground rules for the collaborative relationship. Trading partners exchange information about their corporate strategies and business plans in order to collaborate on developing a joint business plan. The Joint Business Plan identifies the significant events that affect supply and demand in the planning period, such as promotions, inventory policy changes, store openings/closings, and product introductions.

Figure 2.1: The CPFR Reference Model

- *Demand and Supply Management phase* involves the development of a shared plan on the consumer demand. The consumer demand at the point of sale is categorized as sales forecasting and the future product ordering based on the sales forecast is referred as order forecast.

- *Execution* involves the order generation which transitions forecasts to firm demand, and order fulfillment, the process of producing, shipping, delivering, and stocking products for consumer purchase.

- *Analysis phase* involves monitoring the execution of activities for exceptions that are identified during strategy and planning phase. Calculation of key performance metrics, plan adjustments for improving results are also take place in this phase.

While these Collaboration Activities are presented in logical order, most companies are involved in all of them at any moment in time. There is no predefined sequence of steps. Execution issues can impact strategy, and analysis can lead to adjustments in forecasts. The details of the four phases are given in the nine-step process model of CPFR presented in next section.

2.1.3 CPFR Process Model

The process model of CPFR consists of nine primary process activities. It is possible to deploy different scenarios to the same process model based on the relationships and competencies of the trading partners. CPFR does not fix the manager of the process, instead provides alternative lead roles for forecasting. It should be noted that these alternatives only show the leader of the forecasting process since the main point of CPFR is developing a shared forecast.

The model is segmented into stages and illustrated in Figure 2.2.

The first two steps belong to the strategy and planning phase of the reference model where trading partners exchange information about their corporate strategies and business plans in order to collaborate on developing a joint business plan. The first step targets to develop a collaboration agreement which addresses each party's expectations and the actions and resources necessary for success. In this step, the buyer and seller co-develop a general business arrangement that includes the overall understanding and objective of the collaboration, confidentiality agreements, data to be shared, and the empowerment of resources to be employed throughout the CPFR process. Second step, involves the development of the joint business plan which improves the overall quality of forecasting by including data from both partners. Partners should share data to clarify the item management profiles such as order minimums and multiples, lead times, order intervals, etc.

Steps 3-8 belong to demand and supply management phase of the reference model and involve the development of a shared plan on the consumer demand. In the third step, consumption data such as point-of-sale data, historical shipments, and withdrawals are shared to create a joint sales forecast.

In the fourth step, the items that do not conform to the sales forecast constraints agreed in the collaboration agreement. In the fifth step the guideline recommends that the identified exceptions in the previous step are resolved by querying shared data, email, telephone conversations and meetings; and submitting any change to the sales forecast.

In step six, the sales forecast, causal information, and inventory strategies are combined to generate a specific order forecast that supports the shared sales forecast and the joint business plan. The result of Step 6 is a time-phased, netted order forecast. The order forecast allows the seller to allocate production capacity against demand while minimizing safety stock. It also gives buyers increased confidence that orders will be delivered. The collaboration made in this phase reduces uncertainty between trading partners and leads to

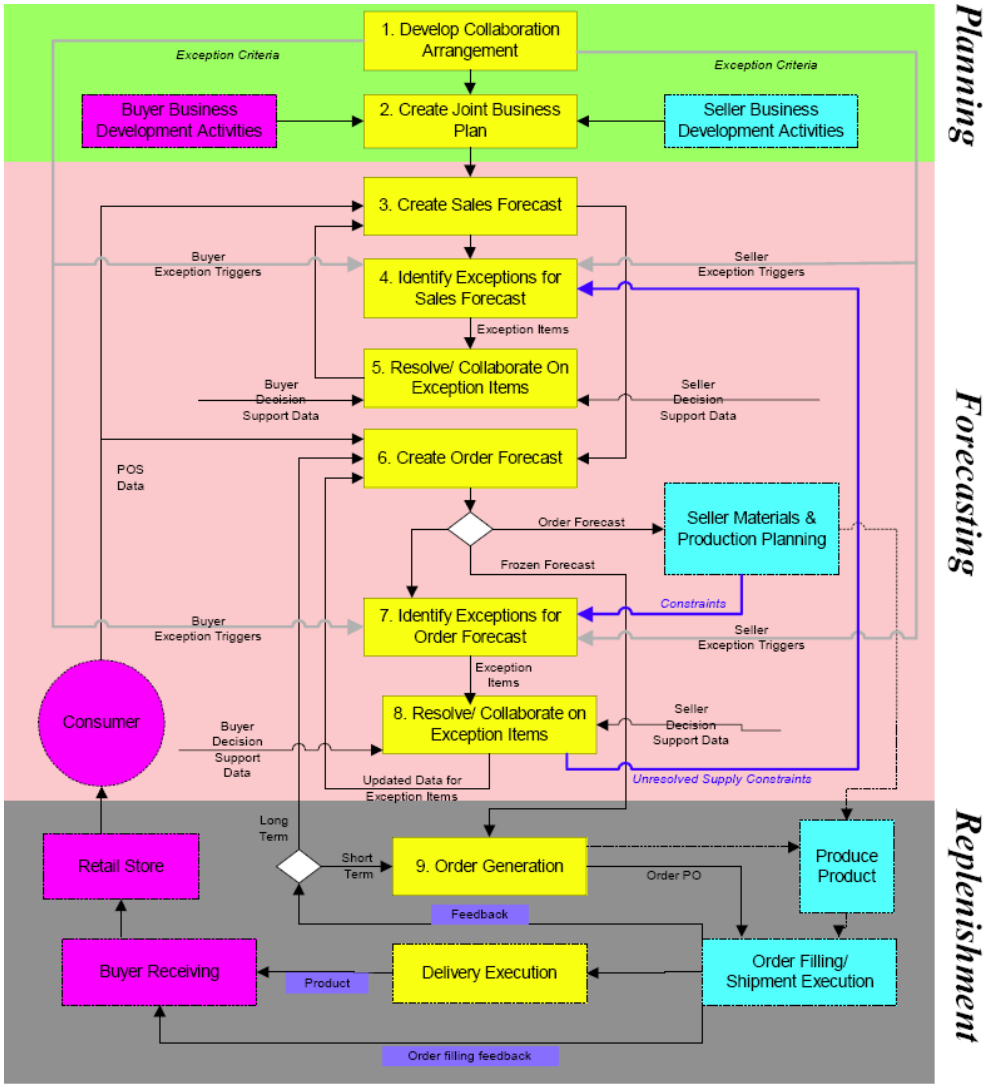

Figure 2.2: The CPFR Process Model

consolidated supply chain inventories.

Steps 7-8 are the steps to identify and resolve any exceptions for order forecast which are similarly performed for the sales forecast. The last step belongs to the execution phase of the reference model and involves the order generation. As a result, a committed order is generated from the frozen period of the order forecast.

2.1.4 CPFR Integration in Collaborative Environments

In order to facilitate CPFR, the trading partners should have the necessary infrastructure to build, share and adjust online forecasts and plans. For some companies, achieving internal collaboration can pose a bigger challenge than working with customers or suppliers. CPFR

does not mandate any technology. However, it should be integrated with the enterprise systems of record that produce and consume supply chain data. Figure 2.3 illustrates where CPFR fits in. Although it does not mandate anything in the internal process of enterprises, the trading partners should have necessary infrastructure to share information with CPFR processes.

A core CPFR objective is to establish a common process that can be used not only between two trading partners, but across an entire marketplace. To achieve this objective, CPFR builds upon GS1 XML [27] standards for item identification, location identification, and electronic commerce message interchange. However, since it does not mandate and any messaging standard to be used in the collaboration process, trading partners need to agree on a set of standards. Another issue for trading partners is the integration of their legacy applications. The trading partners cannot easily leave their existing systems should have necessary infrastructure to share information with CPFR processes or they should extend their systems for this.

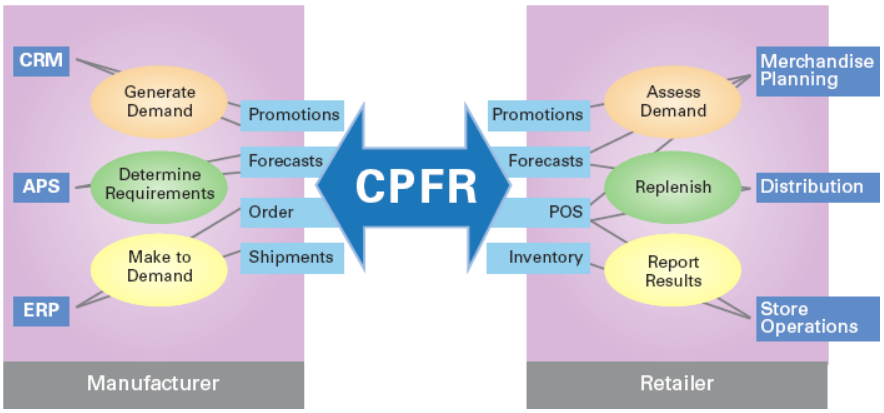

Figure 2.3: The Role of CPFR

In our architecture, CPFR constitutes the core of the planning process semantics; the requirements of the business process, the behavior of the template business processes, and the type of the exchangeable messages were all built primarily considering the CPFR guidelines.

2.2 Global Standards One XML - GS1 XML

Global Standards One (GS1) [26] is a family of standards focusing on different aspects of supply chain integration such as electronic products codes, product information synchroniza-

tion and the electronic document standards. GS1 was formed in early 2005 by the European Article Number (EAN) [15] and the Uniform Commercial Code (UCC) [40] organizations when they joined together. EAN and UCC were two organizations that heavily contributed to the adoption and proliferation of barcodes. The part addressing the electronic document interoperability in this family of standards is GS1 eCom. In GS1 eCom, there are two distinct categories: the earlier eCom standards that are based on Electronic Document Interchange (EDI), called EANcom [16] and the newer generation GS1 XML [27] which is defined using eXtensible Markup Language (XML) Schema.

As shown in Figure 2.3, a GS1 XML document is represented with a StandardBusiness-Document, which contains a StandardBusinessDocumentHeader (SBDH) and a Message. StandardBusinessDocumentHeader provides information about the routing and processing of the XML instance document contained in the GS1 XML Message. In other words the SBDH contains the configuration information for the transport software to send the message to its destination.

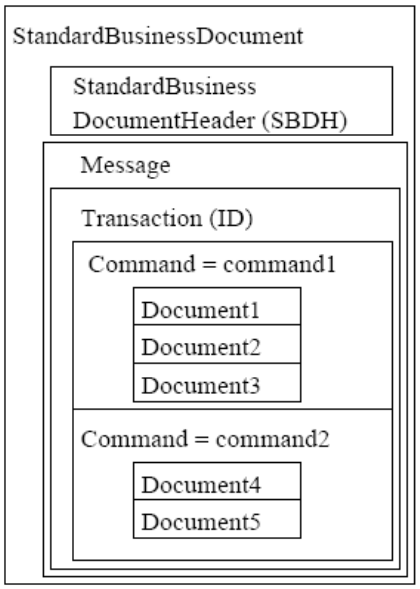

Figure 2.4: The Structure of GS1 XML

The message part residing in the StandardBusinessDocument is the part containing the specific information about the document. In the CPFR XML Messaging Model [11], the following eight messages are listed as the documents that can be exchanged during the

planning process:

- *Forecast:* Projected demand for an item sourced at a seller location and consumed at a buyer location
- *Forecast revision:* A set of proposed changes to a forecast, as the result of, for example, promotional activity, weather, distribution or transportation issues, or re-planning
- *Product activity:* Actual product movement observed, which may include on-hand quantities, distribution center (DC) withdrawals, or Point-of-Sale (POS) Data
- *Performance history:* A collection of values gathered for key performance metrics in the trading partner relationship, such as forecast error, sales growth, or the number of emergency orders
- *Exception notification:* Indication of a variance from trading partner guidelines for changes in a forecast, differences between partner forecasts, or key performance metrics (e.g., forecast error, overstock)
- *Exception criteria:* Definition of the threshold for variances beyond which an exception message should be triggered
- *Event:* Description of a promotion, inventory policy change, or other planned event, along with its expected and actual impacts on the supply chain
- *Item information request:* One trading partner's request to the other to send product activity, forecast, or performance data, when the partner does not automatically send data

On the 2.0.2 version of GS1 XML messages there are two additional messages listed:

- *Item location profile:* Parameters to guide the collaborative replenishment process for an item sourced at a seller location and consumed at a buyer location
- *Retail Event:* This document type defines the business need, lifecycle, structure and content of messages that communicate information about retail events (such as promotions, product introductions, community or environmental events) that affect supply or demand.

In our architecture, this list of messages has been used as the messages exchanged between the collaborating partners. Among the messages stated above, a *Retail Event* message

(also along with some additional information) can represent all the information that can be expressed in an *Event* message. Therefore, the "Event" message can be discarded. We have included two other messages (*"Inventory Activity Or Inventory Status"* and *"Purchase Conditions"*- included in 2.1.1 version of GS1 XML) into the messages that can be used during collaborative planning process differently from the list above. Please see the Section 3.2.1 for the details.

2.3 ebXML Business Process Specification Schema - ebBP

Electronic Business XML (ebXML) is an initiative from and United Nations Centre for Trade Facilitation and Electronic Business. ebXML aims to provide the exchange of electronic business data in Business-to-Business and Business-to-Customer environments.

Within this initiative, the ebXML Business Process Specification Schema (ebBP) technical specification [17] defines a standard language by which business systems may be configured to support execution of Business Collaborations consisting of Business Transactions between collaborating parties or business partners.

The specification schema supports the specification of business transactions and the choreography of business transactions into business collaborations. These patterns determine the actual exchange of business documents and business signals between the partners to achieve the required electronic commerce transaction. Business Collaboration consists of a set of specific roles collaborating through a set of choreographed Business Transactions. Each Business Transaction is conducted between two parties playing complementary abstract roles namely, the Requesting Role and the Responding Role.

Choreography is an ordering of Business Activities within a Business Collaboration. The purpose of a Choreography is to specify which Business Transaction Activity and/or Collaboration Activity should (are expected to) happen.

A Business Transaction is atomic; it cannot be decomposed into lower level message exchanges that could be reused independently unlike a Business Collaboration. Furthermore, one or more Business Signals may additionally be exchanged as part of a Business Transaction to ensure state alignment of the involved parties. There are a number of States that facilitate the choreographing of Business Activities. These include a Start state and a Completion state (which comes in a Success and Failure flavor) as well as a series of gateways: a Fork gateway, a Join gateway and a Decision gateway. In ebBP, the Business Transactions are defined as an abstract super class with eight concrete business patterns as follows [17]:

- *Commercial Transaction (or Business Transaction):* Typically this pattern defines a formal obligation between parties. The semantics defined by commercial transaction is as follows:
  - A response is required.
  - Either a request receipt acknowledgment or exception is required.
  - Either response receipt acknowledgment or exception is required.
- *Notification:* Used for business notifications such as a Notification of Failure. It represents a formal exchange between parties.
- *Information Distribution:* Represents an informal information exchange between parties.
- *Query/Response:* Used by a requester for an information query of which the responding party already has.
- *Request/Confirm:* Used where an initiating party requests confirmation about their status with respect to previous obligations or a responder's business rules.
- *Request/Response:* Used when an initiating party requests information that a responding party already has and when the request for business information requires a complex interdependent set of results.
- *Data Exchange:* Allows a partner, industry or community to define a specific Business Transaction pattern that is not in the concrete set. The semantics used for data exchange are partner-specific.
- *Legacy Business Transaction:* Retained in v2.0 and v2.0.1 technical specifications for conversion purposes only to enable the user community to migrate to the concrete patterns. This pattern is not recommended for use for the concrete Business Transaction patterns.

An ebBP definition references, but does not define, a set of logical Business Documents. It is important to note that the ebBP technical specification focuses on the logical Business Document not a wire format. ebBP definition provides a level of business abstraction independent of any platform, software or services, which also adds flexibility to its use with web services and other XML based technologies. Within an ebBP definition, XPath [49] and XSLT [52] can be used within condition expressions and variables. An ebBP definition,

describes interoperable business processes whether modular and targeted or complex, nested and involve multiple business partners. In the OASIS whitepaper important ebBP v2.0.x capabilities are stated as [33]:

- Standard and extensible business transaction patterns
- Support for multiple role bindings
- Flexibility for complex transaction activities
- Support for use of web service, hybrid and ebXML assets
- Late binding capabilities such as for timing
- Semantic tailoring for business processes and business documents

In our architecture, ebBP is used for defining (i) building blocks through a standard, machine processable business process specification language and (ii) representing the choreography of the jointly built inter-enterprise collaboration process which is generated by grouping these building blocks.

2.4 W3C Web Service and Underlying Standards

The World Wide Web Consortium (W3C) [43] develops interoperable technologies like specifications, guidelines, software and tools. W3C, led by Tim Berners-Lee, is a forum for information, commerce, communication, and collective understanding. W3C is a vendor-neutral organization and was founded in October 1994 to lead the World Wide Web to its full potential by developing common protocols that promote its evolution and ensure its interoperability. Today, W3C has over 450 Members and nearly 70 full-time staff around the world who contribute to the development of W3C specifications and software [44]. W3C promotes interoperability by designing and promoting open (non-proprietary) computer languages and protocols that avoid the market fragmentation of the past. This is achieved through industry consensus, encouraging an open forum for discussion and the definition of the Web as the universe of network-accessible information.

2.4.1 The XML Family

eXtensible Markup Language (XML) [47] was intentionally created for using richly structured documents over the web. XML allows anyone to design his or her own document format

and then create a new document in that format. Almost all documents have some kind of structure and the XML specification defines a standard way to add mark-up to such documents. Structured information contains both content (text, images, etc.) and some indication of what role that content plays (for example, content in a header section has a different meaning from content in a footnote).

The usage of XML Schema (XSD) [50] allows defining the syntactical structure of the data, but not the semantic relationships between the data. The purpose of a schema is to define a class of XML documents, and so the term "instance document" is often used to describe an XML document that conforms to a particular schema. An XSD express shared vocabularies and allow machines to carry out rules made by people. Although XSDs allow some kind of inheritance and generalization for the definition, this is on a pure syntactical level and states nothing about the semantics of the data types.

XML Namespaces are useful to distinguish information when mixing multiple vocabularies in a single XML document instance. The easiest way to uniquely identify the markup elements is to identify each XML element and attribute with an URI (Uniform Resource Identifier) [48]. Without namespaces, the applications would find ambiguous information when the developers of the vocabularies have chosen identical names.

The Extensible Stylesheet Language (XSL) [51] is a rendering vocabulary describing the semantics of formatting information for different media. With the help of XSL, the structure of a XML document can be separated from its visual representation. A XSL style sheet specifies the presentation of a class of XML documents by describing how an instance of the class is transformed into a XML document that uses the formatting vocabulary. Information using one vocabulary can be transformed into an alternate form by using the Extensible Stylesheet Language Transformations (XSLT) [52].

2.4.2 Web Services

A Web Service is a collection of functionality, which is bundled as a single component and published to the network for use by other programs. Web Services are building blocks for creating open distributed systems, and allow companies and individuals to make their digital assets available worldwide. Web Services support re-usability of software as well as dynamic access to remote software services or components. The following characteristics of Web Services will form the heart of the next generation, distributed Internet systems:

- Interoperability: any Web Service can interact with any other Web Service, because of

new standard protocols as Simple Object Access Protocol (SOAP) [36] supported by all of the major vendors.

- Ubiquity: Web Services communicate using Hypertext Transfer Protocol (HTTP) and XML. Therefore, any device that supports these technologies can both host and access Web Services.

- Low barrier to Entry: the concepts behind Web Services are easy to understand and free toolkits from vendors like IBM and Microsoft or from open source communities like Apache [44] allow developers to quickly create and deploy Web Services.

- Wide Industry Support: all of the major vendors are supporting the surrounding Web Services technology, e.g. Microsoft .NET platform, or IBM Visual Age.

- The core Web Services standards were jointly developed and submitted to the W3C and are briefly introduced next.

The Simple Object Access Protocol (SOAP) [36] is a lightweight and simple XML-based protocol, designed to exchange structured and typed information on the Web by invoking methods on servers, services, components and objects. The latest version, SOAP 1.2, has the status of a W3C recommendation. SOAP was originally developed for distributed applications to communicate over HTTP and through corporate firewalls. SOAP defines the use of XML and HTTP to access services, objects and servers in a platform-independent manner.

A significant aspect of Web Services is that any XML document can be encapsulated in a SOAP envelope and exchanged by applications. A SOAP client inserts an XML document into a SOAP envelope and posts it using HTTP (a process referred to as "marshalling") to a SOAP listener, which can be nothing more than a URL address containing Active Server Pages (ASP) or Java Server Pages (JSP) code that accepts the delivery. SOAP messaging is very attractive for the following reasons:

- The SOAP client and listener applications are very simple constructs. A SOAP client can be easily embedded as a routine in any program or web page and the listener is a URL endpoint.

- By using HTTP as its transport, a SOAP message can go anywhere (i.e. through firewalls), use existing Secure Sockets Layer (SSL) facilities for authentication and encryption, and leverage the infrastructure scale of the World Wide Web.

- Everything in the message, the SOAP envelope, header and body (the encapsulated documents) is expressed in XML, thus making the entire object entirely transparent. Because the semantics and structure of the documents are fully exposed, extensive validation, manipulation, routing and transformation functions can be applied to the documents without writing application code to do so.

- Unlike with the Common Object Requesting Broker Architecture (CORBA) [8], an earlier architecture for distributed computing, a Web Services network endpoint (an HTTP URL or other address) can support multiple XML formats, providing a real polymorph interface that will not break down with new versions.

The Web Service Description Language (WSDL) [46] is an XML-based language that provides a model for describing Web services and in June 2007 it was published as an official recommendation standard by W3C.

A WSDL document is written in XML. The WSDL document describes a Web Service, specifies the location of the service and the operations (or methods) the service exposes. A WSDL document resides at a URL location and is linked to a program module that can be located elsewhere. You can actually point a browser at a WSDL web page and it will expose the methods of the program to you. In addition, a WSDL document informs you of what information you have to provide the program to do its job, and what information the program will return. The way this information is conveyed to and from the program is through a SOAP message. The program in turn, sends the results of its process back to the request originator in another SOAP message. A SOAP message is simply an XML formatted text document encapsulated with an HTTP header for transport through the Internet. Because HTTP is one of the transport mechanisms used by SOAP, a Web Service method call can be made to and from any Web enabled computer anywhere in the world.

In summary, Web Services provide two distinct functions: WSDL exposes the methods of a software program and allows a remote user to invoke those methods by transmitting the variables and parameters of the operations in a SOAP envelope. SOAP Messaging, independent of WSDL, enables XML documents to be routed and manipulated using the semantic content of the envelopes and/or documents.

In our architecture, the functionalities of the legacy applications of supply chain partners who want to collaborate are exposed as web services. The partners design and customize the business process with the CPFR Designer Tool considering these web services as atomic elements of the process.

2.5 Web Services Business Process Execution Language - WS-BPEL

The OASIS standards organization has defined the Web Services Business Process Execution Language (WS-BPEL) as a standards-based way of orchestrating a business process composed of services [18]. As an execution language, WS-BPEL defines how to represent the activities in a business process, along with flow control logic, data, message correlation, exception handling, and more. WS-BPEL is a specification that models the behavior of web services in a business process interaction. The specification provides a grammar for describing the control logic required to coordinate web services participating in a process flow. This grammar can then be interpreted and executed by an orchestration engine, which is controlled by one of the participating parties. The engine coordinates the various activities in the process, and compensates the system when errors occur. WSDL has the most influence on the WS-BPEL language [18]. As it can be seen in Figure 2.5, WS-BPEL is essentially a layer on top of the service model defined by WSDL 1.1, with WSDL defining the specific operations allowed and WS-BPEL defining how the operations can be sequenced. A WS-BPEL document leverages WSDL in three ways:

1. Every WS-BPEL process is exposed as a web service using WSDL. The WSDL describes the public entry and exit points for the process.

2. WSDL data types are used within a WS-BPEL process to describe the information that passes between requests.

3. WSDL might be used to reference external services required by the process.

WS-BPEL provides support for both executable and abstract business processes. An executable process models the actual behavior of participants in a specific business interaction, essentially modeling a private workflow. Abstract business processes are partially specified processes that are not intended to be executed. An Abstract Process may hide some of the required concrete operational details. Abstract Processes serve a descriptive role, with more than one possible use case, including observable behavior and process template. Essentially, executable processes provide the orchestration support described earlier while the business protocols focus more on the choreography of the services. WS-BPEL is meant to be used to model the behavior of both Executable and Abstract Processes.

The specification includes support for both basic and structured activities. One can think of a basic activity as a component that interacts with something external to the process itself.

Figure 2.5: Relation of the Services Within WS Technology Stack

For example, basic activities would handle receiving or replying to message requests as well as invoking external services. The typical scenario is that there is a message received into the WS-BPEL process. The process may then invoke a series of external services to gather additional data, and then respond to the requestor in some fashion. In the Figure 2.6, the <receive>, <reply>, and <invoke> messages all represent basic activities for connecting the services together.

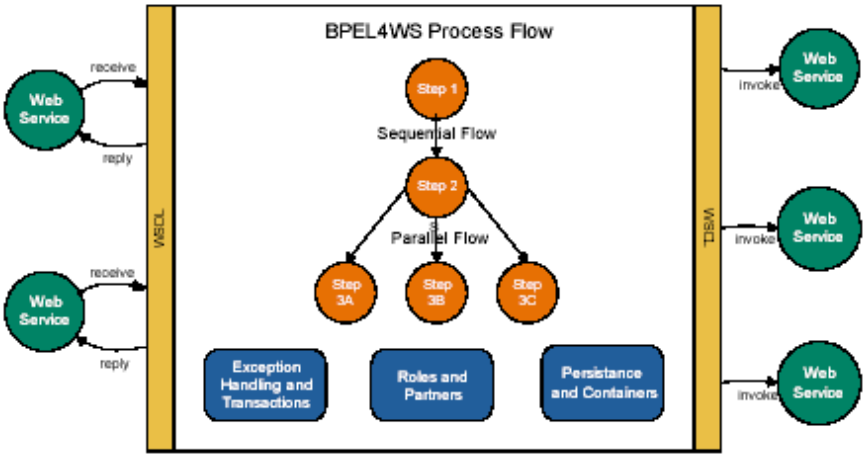

Figure 2.6: WS-BPEL Process Flow

In contrast, structured activities manage the overall process flow, specifying what activities should run and in what order. Structured activities might specify that certain activities should run sequentially or in parallel. These activities also provide support for conditional

looping and dynamic branching. One can think of structured activities as the underlying programming logic for WS-BPEL. In Figure 2.7 a simple example is shown illustrating how a sequential activity would be described, containing basic activities to receive a message, assign value to variables, invoke a transformation service, assign the transformed value to another variable and invoke a partner's service.

The complete list of WS-BPEL activities is: <receive>, <reply>, <invoke>, <assign>, <throw>, <exit>, <wait>, <empty>, <sequence>, <if>, <while>, <repeatUntil>, <forEach>, <pick>, <flow>, <scope>, <compensate>, <compensateScope>, <rethrow>, <validate>, <extensionActivity>.

```
<sequence>
        <receive partnerLink="receiveFromManufPL"
                operation="receiveFromManufacturer" variable="ECR_Seller"/>
        <assign>
                <copy>
                        <from part="message" variable="ECR_Seller"/>
                        <to part="message" variable="requestMess"/>
                </copy>
        </assign>
        <invoke partnerLink="sendToISUPL" operation="translateDocument"
                inputVariable="requestMess" outputVariable="responseMess"/>
        <assign>
                <copy>
                        <from part="message" variable="responseMess"/>
                        <to part="message" variable="ECR_Buyer"/>
                </copy>
        </assign>
        <invoke partnerLink="sendToRetailerPL"  operation="receiveECR"
                                                inputVariable="ECR_Buyer"/>
</sequence>
```

Figure 2.7: A WS-BPEL Sequence Activity Example

As stated above, WS-BPEL Business Processes offer the possibility to aggregate web services and define the business logic between each of these service interactions. Each service interaction can be regarded as a communication with a business partner. The interaction is described with the help of partner links. Partner links are instances of typed connectors which specify the WSDL port types the process offers to and requires from a partner at the other end of the partner link [45]. A partner link can be regarded as one particular communication channel. Such an interaction is potentially two sided: the process invokes the partner and the partner invokes the process. Therefore, each partnerLink is characterized by a partner link type and a role name. The Figure 2.8 below illustrates a simple example

of how partners can be defined in WS-BPEL.

Finally, WS-BPEL provides a robust mechanism for handling transactions and exceptions, building on top of the WS-Coordination and WS-Transaction specifications. These corollary specifications include the support necessary to manage and coordinate the operations of a business activity.

```
<partnerLinks>
        <partnerLink name="sendMessageToISUPL"
                     partnerLinkType="manubpel:sendMessageToISUPLType"
                     partnerRole="sendMessageToISURole"/>
        <partnerLink name="sendMessageToRetailerBpelPL"
                     partnerLinkType="manubpel:sendMessageToRetailerBpelPLType"
                     partnerRole="sendMessageToRetailerBpelRole"/>
        <partnerLink name="receiveDataFromRetailerBpelPL"
                     partnerLinkType="manubpel:receiveDataFromRetailerBpelPLType"
                     myRole="receiveDataFromRetailerBpelRole"/>
</partnerLinks>
```

Figure 2.8: A sample WS-BPEL Partner Links Definition

In WS-BPEL, a set of activities can be grouped in a single transaction through the <scope> tag. This tag signifies that the steps enclosed in the scope should either all complete or all fail. Within this scope, the developer can then specify compensation handlers that should be invoked if there is an error. For example, if part of a travel reservation process fails, the handler would identify how to rollback the other parts of the transaction. The transactional mechanisms within WS-BPEL work hand-in-hand with exception handling. BPEL provides a robust exception handling mechanism through the use of throw and catch clauses, similar to the Java programming language.

At a high level WS-BPEL is an XML language that provides a rich set of activities to describe an executable business process [28]. The processes and activities can be synchronous or asynchronous, short-lived or long-running; BPEL provides a sophisticated language for defining the process flow, system interactions, data manipulation, exception handling, compensation rules, and so on.

In our architecture, from the customized CPFR process in choreographic definition of ebBP, for each of the collaborating partners in the Supply Chain WS-BPEL files are generated which serves as the executable definitions of the CPFR process. WS-BPEL definitions for each partner will define the orchestration of their services from their point of view within complete CPFR business process.

CHAPTER 3

BUILDING BLOCKS OF COLLABORATIVE PLANNING PROCESS AS BINARY COLLABORATIONS

3.1 Introduction

Section 2.1 explained the CPFR®guidelines which were defined as the collaborative business practices enabling the trading partners to have visibility into one another's critical demand, the order forecasts and the promotional forecasts through a systematic process of shared brand and category plans, exception identification and resolution. The objective of CPFR is to improve efficiencies across the extended supply chain, reducing inventories, improving service levels and increasing sales.

In successful CPFR pilot applications, it has been reported that the definition and deployment of CPFR processes within a supply chain consortium is too costly and labor intensive. Although CPFR provides guidelines, there is no machine processable process templates defined. Also, CPFR does not mandate any technology to implement the CPFR technology: the consortiums agreed to collaborate based on CPFR guidelines, need to implement their own strategy. This also hampers the scale of CPFR processes: consider, for example, a company that needs to communicate with two different supply chain consortiums: most probably separate processes will be implemented for each of the CPFR processes needed.

For addressing these problems and accelerating the adoption and the usage of the CPFR, in this thesis work, machine processable building blocks in ebXML Business Process Specification (ebBP) have been defined. By using these building blocks, the generation of completely customized business process definitions is provided to the partners in the supply chain. In this chapter, the machine processable building blocks modeled for the CPFR Plan-

ning Process are explained. First, the graphical Business Process of the CPFR Planning Process is given and then the ebBP definitions and how the elements in the CPFR Planning Process are modeled with the ebBP definitions are presented.

3.2 The CPFR Process

As explained in Section 2.1, in the previous chapter, the CPFR process is defined with the nine primary activities in CPFR guidelines. These primary activities are defined in the abstract level and it is possible to deploy different scenarios to the same process model based on the relationships and competencies of the trading partners. The main idea of the CPFR process is sharing enough information in order to generate more accurate forecasts and iterating continuously over the planning process for improving the accuracy of the forecasts.

The contents of these nine steps can (and should) be interpreted differently from collaboration to collaboration. By the guidelines, only what should be achieved in the steps is defined, such as "2-Create Joint Planning Process", "3- Create Sales Forecast" or "4- Identify Exceptions for Sales Forecast" yet how it can be achieved is not described. The nine steps are not in the form of a solid planning process which can easily be implemented for automatization.

When CPFR Process is inspected, we realized that the planning process can be modeled by a number of message exchanges between the binary collaborators since at the core of the CPFR, there lies sharing information between the involved partners. Additionally, the message exchange process should be able to be performed iteratively over and over again to address the aim of the CPFR on the improvement of the forecast by the revisions.

3.2.1 GS1 Messages Used

As explained in the Section 2.2 of the Enabling Technologies Chapter, GS1 XML defines a number of message schemas covering the Order, Delivery, Plan and Payment operations of Supply Chain Management. Among these, most of the planning message schemas from the version 2.0.2 were included in the CPFR Designer Tool since the designer tool addresses the planning process of the Supply Chain Management. One of the excluded messages from the set was "Event" message. It has been excluded because another message defined exists, "Retail Event" message, which can represent all of the data that can be represented by the Event messages. The other message not used is "Item Information Request" representing a information request from the other side. In the CPFR Process, the business process defined

jointly before starting the execution of it. Therefore, all the message exchange order is defined from the beginning, i.e. it is already defined which message exchange will be performed after a message exchange. This removes the need of a specific request.

Additionally, one message from Delivery, "Inventory Activity Or Inventory Status", has been included into the messages used in the CPFR Designer Tool. Although it is defined under the Delivery category, it enables the information transfer of the Buyer to the Seller about its Inventory Status which is used by the Buyer before the generation of the Order Forecast.

Moreover, from the version 2.1.1, Purchase Conditions has been added to the set of the messages to be used during the planning in the CPFR Designer Tool. It will carry the results related to the planning after a collaboration agreement is generated. As a result, the list of the messages used during the exchange of information within CPFR Process is as follows:

- *Forecast*
- *Forecast revision*
- *Product activity*
- *Performance history*
- *Exception notification*
- *Exception criteria*
- *Item location profile*
- *Retail Event*
- *Inventory Activity Or Inventory Status*
- *Purchase Conditions*

3.2.2 CPFR Scenarios

The contents of the nine primary activities considerably change both based on the CPFR scenario chosen and based on the specific needs of the collaborating partners. For the first difference (i.e. the selection of the CPFR scenario), the CPFR Designer Tool presents a set of CPFR Templates to be chosen based on each CPFR scenario. The second need is achieved by providing the user with the ability to customize the default given CPFR Planning Process.

According to the CPFR Guidelines [2], there are four scenarios which are based on who the responsible partner is for performing the three main activities: Sales Forecast, Order Forecast and Order Generation. In these activities, both partners have input to the forecasting and the planning processes but one partner takes the ultimate ownership. Scenarios A, B, C, and D in Table 3.1 represent process variants which may be applied to a particular trading partner relationship based on competencies, resources, and systems. That CPFR does not dictate who ultimately manages the forecasting or the replenishment processes provides flexibility to the collaborating partners.

Table 3.1: CPFR Scenarios Described in Guideline

Scenario	Sales Forecast	Order Forecast	Order Generation
Scenario A	Buyer	Buyer	Buyer
Scenario B	Buyer	Seller	Seller
Scenario C	Buyer	Buyer	Seller
Scenario D	Seller	Seller	Seller

Scenario A is a buyer-led forecasting and replenishment process, in which the buyer controls the sales forecast, the order forecast, and the order release. Scenarios B, C, and D delegate the order release to the seller. In Scenario B, the buyer provides a forecast of demand, while the seller focuses on the replenishment. In Scenario C, the buyer leads the order forecasting process as well. In Scenario D, the seller facilitates the entire process, including both the demand forecasting and the replenishment planning.

In the thesis work, since we are addressing the planning process only, the Order Generation activity is excluded in the process. This is because the Order Generation is within the execution phase of the supply chain management and it is handled by the Enterprise Resource Planning (ERP) systems of the partners. Therefore, with the exclusion of the order generation, three CPFR scenarios remains since Scenario A and Scenario C would be the same without the Order Generation activity. Therefore, the scenarios that will be used in the implementation have been reshaped and renamed as it is shown in Table 3.2.

Table 3.2: CPFR Scenarios Modified for the Implementation

Scenario	Sales Forecast	Order Forecast
Scenario I	Buyer	Buyer
Scenario II	Buyer	Seller
Scenario III	Seller	Seller

3.2.3 Definition of an Example CPFR Process Template

As an example, among the process templates provided to the users in the CPFR Designer Tool, the process template of "Scenario I" is provided in this section. In this scenario, the responsibility of both the Sales Forecast and the Order Forecast belongs to the Buyer.

Representation of the Process

For each of the CPFR steps in the nine step CPFR process model given in Figure 2.2, a schema is presented as shown in Figure 3.1. In the illustration, "CPFR Step X" on the right upper corner of the figure means that the definition within the rectangle belongs to the CPFR Step X on the nine step CPFR process model. The steps drawn in dashed (like "CPFR Step Y" in Figure 3.1) represent a referral to a step where definition is provided somewhere else on the document. This usually appears at the end of a step definition stating from which step to continue after the defined step.

Messages are shown with a document icon and with the type of the document written below it. Some message types can be used for holding more messages for different purposes. For example, a message of the type Product Activity can carry both Point of Sale (POS) Data and Distribution Center (DC) Data. In these situations an explanation within parenthesis indicates the purpose of the message for the Supply Chain Partners. Messages are connected to each other with a thin arrow or with conditions. A direct connection to a Message B after Message A means that after the exchange of Message A is performed, the exchange of Message B should be performed. If the messages are connected via a condition, the outcomes of the condition are labeled on the arrows leaving the condition, like "yes" and "no" in the figure above.

The thick arrows to left or to right indicate the sender and the receiver of the message.

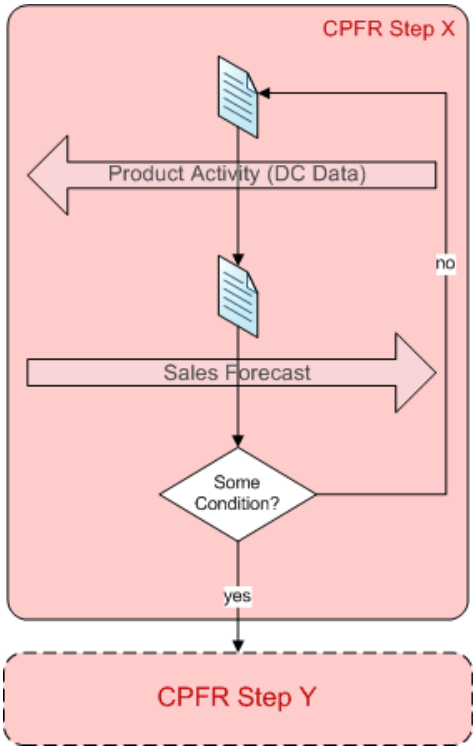

Figure 3.1: CPFR Steps Legend

Within this document, the following convention is used:

- Arrow *to the right*: Messages is sent *from Seller to Buyer*.
- Arrow *to the left*: Messages is sent *from Buyer to Seller*.
- Arrow *with two heads*: Message can be sent both *from Seller to Buyer* and *from Seller to Buyer*.

To shortly explain the message above, within the "CPFR Step X" there is two message exchanges: exchange of a Product Activity and exchange of a Sales Forecast. Product Activity Message is used for holding the POS Data. First, POS Data is sent from Buyer to Seller and then Sales Forecast is sent from Seller to Buyer. After these two message exchanges are finalized, "Some Condition?" is checked. If it is negative, the state is changed to the message exchange of Product Activity. If the result of the condition evaluates as true, then the process continues from the message exchanges within "CPFR Step Y".

CPFR Steps 1-2 (Collaboration Agreement and Joint Business Planning)

The Figure 3.2 presents the business process part for CPFR Steps 1 and 2. As described in the CPFR Guidelines, a CPFR Step 1 starts with the face-to-face meetings and with

the Collaboration Agreement. The Collaboration Arrangement is the preparatory step that defines the scope of the project, assigns roles and establishes procedures for data interchange, issues identification and resolution. With the meetings and the agreement, the following actions are performed:

- Receive and review background information from the sales organization or buyers
- Identify the product categories that should be included in the initial scope
- Define Collaboration Objectives
- Define specific metrics that reflect the above objectives.
- Determine the Event collaboration cycle
- Determine the times of the review meetings to discuss the results
- Document the data sources that are essential for a successful event collaboration process and
- Document additional information that can be used in the event analysis.

The formation of the agreement and the kickoff meeting are manual and they are out of the business process that will be executed with WS-BPEL.

The first step of CPFR Process continues with the exchange of the Messages Purchase Conditions. Afterwards, for determining the exception criteria that should be monitored and handled during the execution, Exception Criteria messages are exchanged. The exchange of the Exception Criteria Revision messages continue until the criteria is accepted by the both sides. Once the exception criteria are determined, the process follows the CPFR Step 2.

The focus of the Joint Business Planning is on upcoming promotions and other planned retail events (such as holidays, major sports events and other consumer-related activities). At this stage, the seller and the buyer collaborate on developing optimal plans for these events - prior to developing detailed sales forecasts during the next stage of the process. Basically, in CPFR Step 2 which is Joint Business Planning phase, there are two messages that should be exchanged and agreed on: Retail Event and Trade Item Location Profile. The revisions are exchanged until the agreement is achieved.

CPFR Steps 3-4-5 (Sales Forecast Generation, Exception Handling)

The CPFR Step 2 helps the buyer and seller agree to the event calendar and event details that meet their joint business and collaboration objectives. The objective of the event calendar

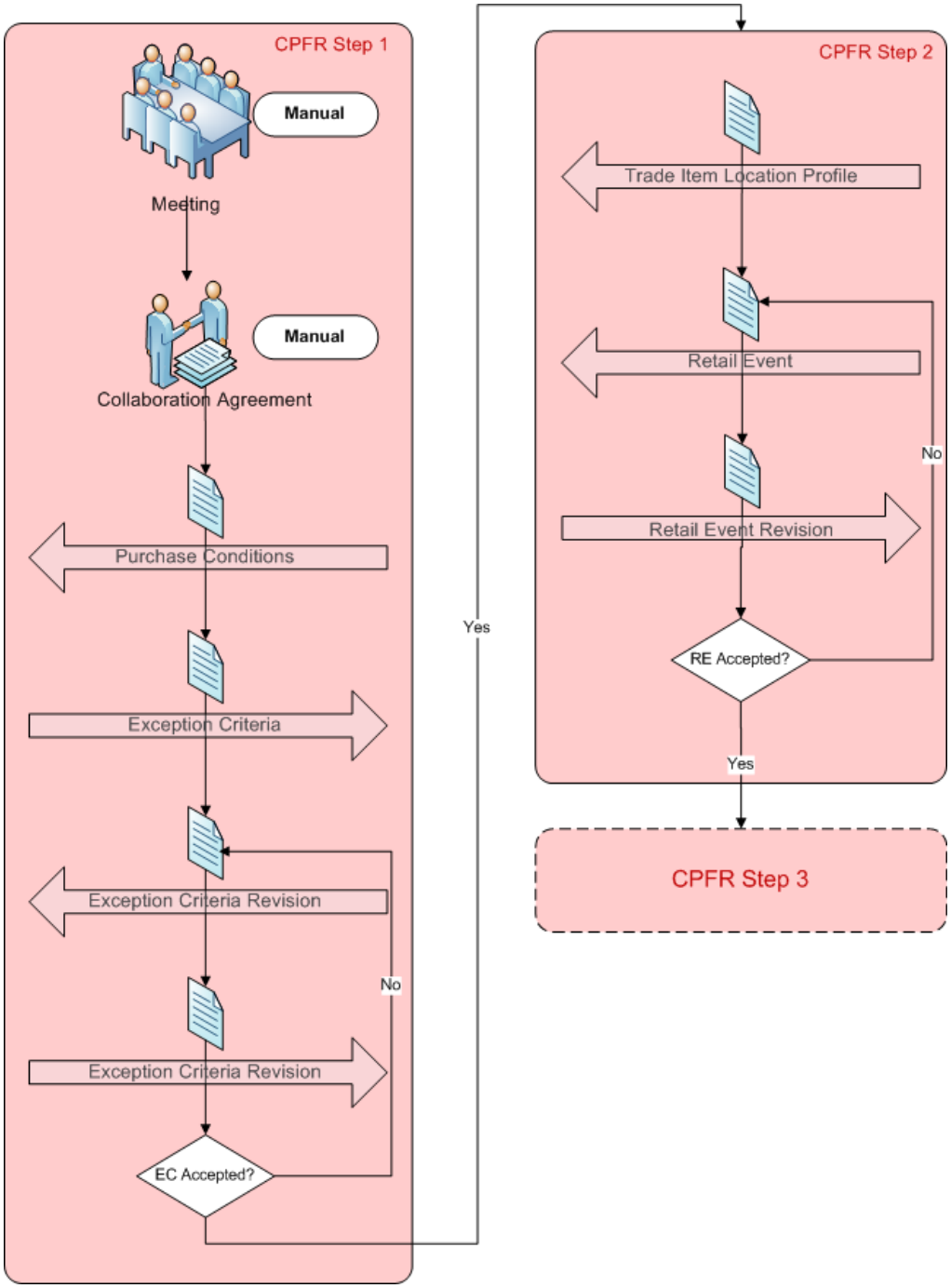

Figure 3.2: CPFR Steps 1-2

is to ensure events are planned to achieve the optimal results, and to enable both parties to plan the execution of the event more accurately, from the preparation of advertising and displays, to the production and delivery of the promotional stock.

In CPFR Step 3, the Sales Forecast is generated. Since we are following Scenario I from the implementation scenarios, the responsible partner for the generation of Sales Forecast

is Seller. Now already having Event Calendar information and the Delivery Plan in their system, there are two more kinds of information that Seller needs for an effective Sales Forecast: POS Data and DC Data. As it can be seen from the Figure 3.3, both of these data are sent within a Product Activity Message. This time there is no revision of the messages because these messages are statistical and historical information collected previously by the Buyer.

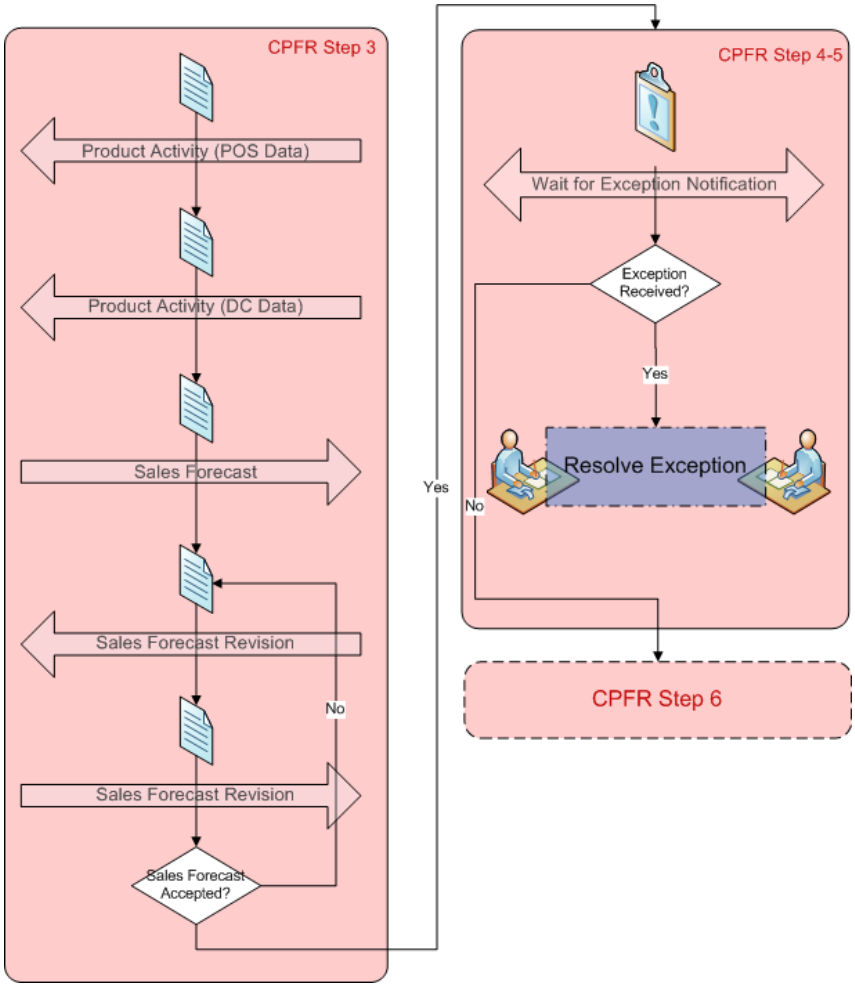

Figure 3.3: CPFR Steps 3-4-5

Based on the event details (dates, products, tactics, etc.) and using the available data source(s) a volume estimate/forecast is created for each product/store combination included in the scope of the event by Seller. During the calculation, sales forecasting algorithms make use of the coefficients for causal factors based on the event history. Once the Sales Forecast suggestion is generated and sent to the Buyer, the Buyer revises it and might recommend

some changes on the Forecast. The Forecast Revision message exchange continues until it is agreed by the both sides.

Between the Sales Forecast Generation and Order Generation there is on the average 6 weeks of time. During this period, both sides observe the changes on the conditions. If one of the partner detects an exception invalidating the exception criteria defined in the CPFR Step 1, it sends an Exception Notification message to the other party. Exceptional circumstances that may be communicated between trading partners are deviations between planned impacts (either between buyer and seller, or between subsequent generations of planned impacts from the same trading partner), as well as deviations between planned and actual impacts. It should be noted that, both sides might detect an exception and therefore, both sides should be capable of sending and receiving exceptions. Of course, for specific implementations if the collaborating parties want to change this behavior, they can customize the process so that one partner will be the responsible from the generation of the exception notifications. CPFR Step 4 is solely composed of the exception generation and receiving activity.

CPFR Step 5, on the other hand, is the resolution of the Exceptions. The activity for resolving the exceptions are modeled in the CPFR Designer Tool with one of the followings:

- Links to other messages (e.g. if there occurs a change in the promotion date, the sales forecast should be regenerated with the update). In this way, by further performing the message exchanges, the exception is tried to be resolved.

- Signals to other CPFR Processes (this is used if n-tier collaboration is performed). In this way, the other CPFR processes that a company is hosting are notified with the exception.

- If the exception is irresolvable, this causes a new partner search and partial or total cancellation of the collaboration.

If there is no Exception Notification Message within the defined period, the process continues with the Order Forecast Generation (CPFR Step 6).

CPFR Steps 6-7-8-9 & Exception Monitoring During Execution

In the supply chain process, it is important for sales forecasts that are created to be converted into the shipment (order) forecasts that can then be used in the production planning processes at the manufacturing locations and be incorporated into the ordering processes at

the retailer. Again as it can be seen from Figure 3.4 the responsibility for creating Order Forecast belongs to the Seller since we are following CPFR Scenario I. The sales forecasts can be transformed into the order forecasts by the incorporating inventory, in transit and on order information. Therefore, Buyer sends the updated versions of the Retail Event, Inventory Status and POS Data to the Seller.

After the Seller creates the Order Forecast using the obtained data, it sends the forecast to the Buyer. The Buyer checks the order forecast and sends back a revision document which if needed includes update requests. Until, there are no further update requests and the Order Forecast is agreed by both sides, the exchange of Order Forecast Revision continues.

After the Order Forecast is frozen, the process continues with the exception detection activity (CPFR Step 7). Similar to the exception detection process which follows Sales Forecast, the exception message exchange after the Order forecast is possible in two ways. The only difference from the Sales Forecast Exceptions is the content of the exceptions.

CPFR Step 8, Order Forecast Exception Resolution activity is handled similarly to the Sales Forecast Exception Resolution. There are links to the previous activities and to other CPFR Processes.

If there is no exception during a period of time, process continues with the Order Generation Step. This activity is not included in the planning process of the CPFR Designer Tool since it is already handled by the ERP systems of the partners. It is shown in the schema only to indicate that there is an additional exception monitoring step when the execution of the Order begins.

From the technical point of view, the exception monitoring and its resolution are exactly same with the Order Forecast Exception Handling and Sales Forecast Exception Handling. The difference is in the content of the exceptions. The actual events and orders are compared to the Forecasted Sales and Forecasted Orders. When there is a situation violating the normal conditions of exception criteria, one of the sides might generate an exception notification. Besides comparison of forecasts, the other information gathered during the execution is observed like event dates, POS data etc. The resolution of the exceptions is the same as the ones before.

During this exception monitoring time, Buyer also waits for a possible Termination message indicating the end of the collaboration. The termination message is not mentioned by the CPFR guidelines, but it is included in our work to indicate the successful termination of the CPFR collaboration. Since the CPFR Planning Process is supposed to be an iterative process, most of the time there will not be a termination message and the process

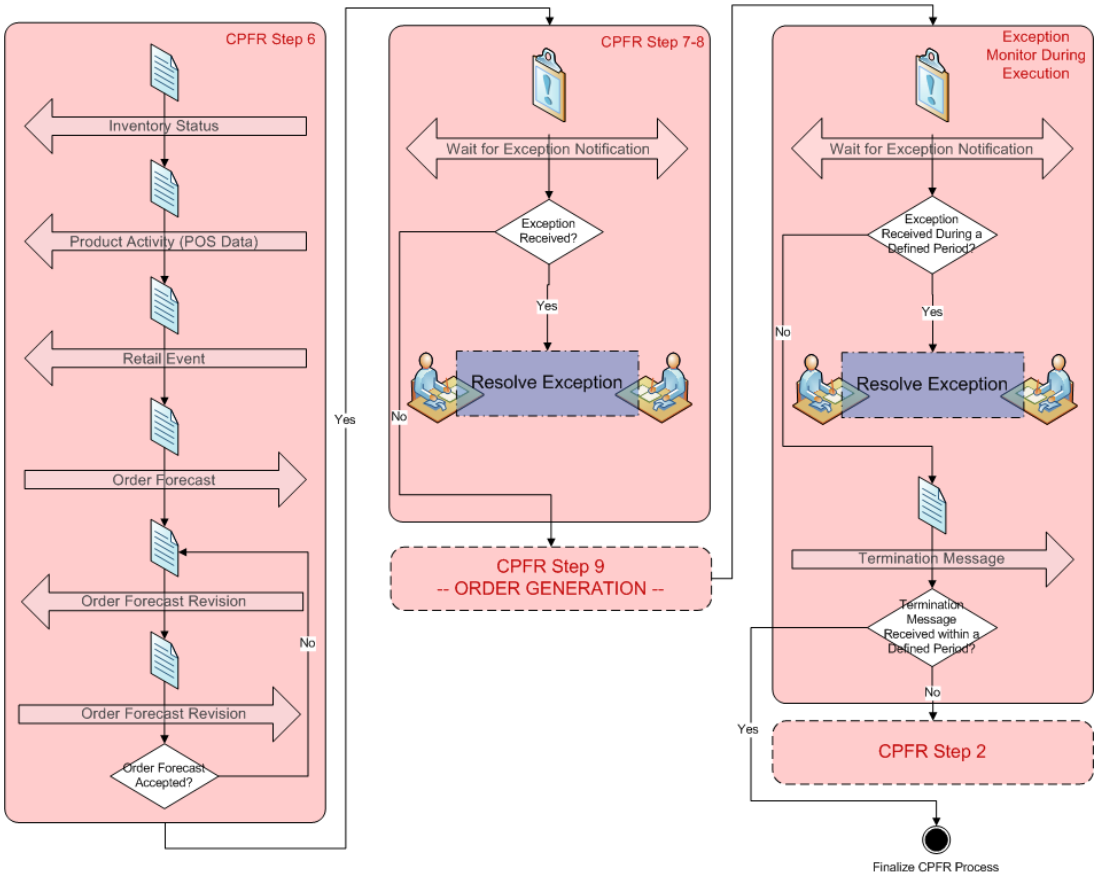

Figure 3.4: CPFR Steps 6-7-8-9

will continue from the CPFR Step 2 with a new cycle of generation of forecasts and order execution.

3.2.4 Customization of the Process

It should be noted that although the process defined above (CPFR Scenario I) or some other scenario will mostly fit to the requirements of most of the partners collaborating in the supply chain management, the customization is provided as it is quite possible that there will be a need for changes in the process. Therefore, we call the above process and the processes of other scenarios (although they are complete processes) as a template signifying that it will be customized by the partners most of the time.

3.3 Building Blocks in ebBP

For a proper definition of the collaborative planning process explained in the previous section, a choreographic definition language, ebBP, has been chosen. The main advantage of

the choreography approach is that it separates the process being followed by an individual business or system within a "domain of control" from the definition of the sequence in which each business or system exchanges information with others. This means that, as long as the "observable" sequence does not change, the rules and logic followed within the domain of control can change at will. Within the chorographic business process definition language standards, ebBP is one of the theoretically most expressive business process definitions. ebBP definition enables the comprehensive description of an interoperable business processes allowing business partners to collaborate and achieve a given business goal.

An ebBP process definition is composed of the following components:

- Business Documents
- Business Transactions
- Business Transaction Activities
- Business Collaborations
- Business Signals

Each of these components and how they are used for modeling the CPFR process templates (which are explained in Section 3.2) are described in the below sub-sections.

3.3.1 Business Document

In the technical specification of ebBP [17], it is stated that Business Documents are defined at the intersection between the ebBP technical specification and the ebXML Core Component specifications. It is added that an ebBP definition will reference, but not define, a set of logical Business Documents.

Therefore, the documents referenced within the ebBP definition do not reflect the actual document format of the partners. In the choreography definition, the business documents defined uses the GS1 Message schemas described in Section 3.2.1. These definitions state the type of the message to be used. The messaging formats of the partners participating to the collaboration can be different from this messaging format. The actual format is defined on the WS-BPEL definitions of the each partner.

In Figure 3.5, an example business document definition for the Retail Event message of GS1 XML specification is given. In the definition, primarily, the name of the document, an id for referencing it within the ebBP definition and the location of the schema are given.

```
<BusinessDocument nameID="Retail_Event_Document"
      name="Retail Event">
      <Specification type="schema"
            nameID="Retail_Event_Specification"
            location="Schemas/RetailEventProxy.xsd"
            name="Retail Event Specification"/>
</BusinessDocument>
```

Figure 3.5: A Business Document Definition Example

Since whatever the process is defined, the number of the document types will be fixed within the CPFR Designer Tool, the ebbP Business Document Definitions corresponding to the ten GS1 Messages mentioned earlier are all defined and are a part of the building blocks. As the user in CPFR Designer Tool uses these message types, they are added to the definition.

3.3.2 Business Transaction

In ebBP, a Business Transaction represents an atomic unit of work that may be associated with a trading arrangement between two business partners. A Business Transaction is conducted between two parties playing opposite abstract roles in that transaction. Each party, as an abstract partner, assumes an abstract role in a Business Transaction. Those roles are always generic and labeled as Requesting and Responding roles. The specific roles (e.g. buyer, seller) are specified at the Business Transaction Activity level, when the Business Transaction definition is used for a distinct purpose. At that point, the abstract partner assumes and occupies a specific role, as a role occupant.

In Figure 3.6, an example ebBP Business Transaction Definition is given which uses the Business Document sample defined in the previous section. Among the possible Business Transaction types in ebBP, Notification which represents a formal exchange between parties has been chosen. Since the business process is defined directly via the tool and the sequence of the message exchanges are known from the CPFR guidelines, there is no need to request-response type of communication.

In the example definition, the notification has three attributes: nameID, name and isGuaranteedDeliveryRequired. nameID attributes provides the ability for this definition to be referenced by another ebBP component. Since the Business Transactions define an abstract transaction and they are also reusable like Business Documents, there is same number of Business Transactions as the Business Documents. The name of the Business Transaction is

```
<Notification
      nameID="iSURF_Retail_Event_Transaction"
      name="Retail Event Transaction"
      isGuaranteedDeliveryRequired="true">
            <RequestingRole
                  nameID="iSURF_RET_Initiator"
                  name="Initiator"/>
            <RespondingRole
                  nameID="iSURF_RET_Responder"
                  name="Responder"/>
            <RequestingBusinessActivity
                  nameID="iSURF_RET_ResBA"
                  name="Retail Event Responding Business
            Activity"
                  isAuthorizationRequired="true"
                  isNonRepudiationRequired="true">
                  <Documentation>
                        This is the messaging for Retail Event
                        exchange.
                  </Documentation>
                  <DocumentEnvelope
                        nameID="iSURF_RET_DocEnv"
                        businessDocumentRef="Retail_Event_Docum
                  ent"
                        name="Retail Event Document"
                        isAuthenticated="persistent"
                        isConfidential="persistent"/>
                  <ReceiptAcknowledgement
                        name="ra2"
                        nameID="iSURF_RET_RBA_RA"
                        signalDefinitionRef="ra2"/>
                  <ReceiptAcknowledgementException
                        name="rae2"
                        nameID="iSURF_RET_RBA_RAE"
                        signalDefinitionRef="rae2"/>
                  <AcceptanceAcknowledgement
                        name="aa2"
                        nameID="iSURF_RET_RBA_AA"
                        signalDefinitionRef="aa2"/>
                  <AcceptanceAcknowledgementException
                        name="aae2"
                        nameID="iSURF_RET_RBA_AAE"
                        signalDefinitionRef="aae2"/>
            </RequestingBusinessActivity>
            <RespondingBusinessActivity
                  nameID="iSURF_ECT_ResBA"
                  name="Exception Criteria Responding Business
            Activity"/>
</Notification>
```

Figure 3.6: A Business Transaction Definition Example

built upon the name of the corresponding Business Document. That the isGuaranteedDeliveryRequired is true indicates that the signals should be used during the execution of this transaction.

One being the Requesting Role (the actor initiating the transaction) and the other being the Responding Role (the actor responding in the transaction) are defined at the beginning of the notification. These roles are abstract and later will be bind to the real roles in collaboration.

Then requesting business activity is defined. In the documentation part, a description of the activity is given. DocumentEnvelope definition makes use of the previously defined Business Documents by referencing one of them. Finally, the four signals are defined by referencing to the generic signals defined.

3.3.3 Business Transaction Activity

Within an ebBP collaboration definition, a Business Transaction Activity (BTA) is the performance of a Business Transaction. Business Transaction definitions can be associated to any number of BTA elements. This means that the same Business Transaction can be performed by multiple Business Transaction Activities in different collaborations, or by multiple Business Transaction Activities in the same collaboration, sometimes with opposite roles.

In other words, a Business Transaction could be used by the two Business Transaction Activities, which can be performed by opposite roles. For example, after a purchase order has been accepted, either party could cancel it (for a certain period of time) using the exact same Business Document interchange.

```
<BusinessTransactionActivity
      name="Send Retail Event BTA-S2B"
      nameID="iSURF_Send_RE_BTA_S2B"
      businessTransactionRef="iSURF_Retail_Event_Transaction">

      <Documentation>CPFR Step 2</Documentation>
      <Documentation>Retail Event</Documentation>
      <TimeToPerform/>

      <Performs currentRoleRef="IDSeller"
                performsRoleRef="iSURF_RET_Initiator"/>
      <Performs currentRoleRef="IDBuyer"
                performsRoleRef="iSURF_RET_Responder"/>
</BusinessTransactionActivity>
```

Figure 3.7: A Business Transaction Activity Definition Example

In the figure above (Figure 3.7), a BTA definition referencing the Business Transaction defined in the previous section is given. First of all, a BTA has a name, a nameID and a businessTransactionRef. Name of the BTA is generated by appending the direction of the message to the end of the name of the message exchanged. Every BTA is given a nameID, since they will be referenced during the collaboration definition while the choreography of the process is defined.

Each BTA references to a Business Transaction and binds the actual roles to the abstract (requestor and responder) roles in Business Transaction. This is done with assigning the roles defined within the Business Collaboration to the role on the Business Transaction via Performs element.

With the Documentation elements, a descriptive name of the message and the step that it belongs are given. Apart from the usual description, the descriptive name is also used when a message type can contain messages for different purposes like the Product Activity Message which can contain both POS Data and DC Data. The CPFR Step that the message exchange belongs is for visualizing the CPFR business process also indicating the steps.

TimeToPerform element which define how much time should the message is expected to come at most is used for optional messages in CPFR Process. The mostly used optional message is the Exception Notifications. They may or may not occur during the process execution. If for a specific period of time, the message is not received, the process continues from the next message exchange stated in the process definition. There is also another next step which is chosen in case the message arrives. In fact, optional message are modeled as the decisions whose condition is the arrival of the message.

Since for each message that is defined in the process, a separate BTA should be defined, the definition of BTA is generated on the fly from the CPFR Designer Tool. The id of the BTA is given as a UUID if it cannot be created from the name of the message. BTAs are the one to one correspondent to the message exchanges defined within the CPFR Process.

3.3.4 Business Collaboration

In ebBP, a Business Collaboration is a set of Business Activities executing Business Transactions between the business partners or the collaborating parties. Each business partner plays one or more abstract partner roles in the Business Collaboration. The state of the Business Collaboration is logical between the parties interacting in a peer-to-peer rather than a controlled environment. The virtual state of the Business Collaboration lies with the involved partners. Peer-to-peer collaboration may involve business partners as well as the

distributed collaborating parties.

Both the Business Transaction Activities and the Choreography of the Business Collaboration describing the ordering and the transitions between Business Transactions or sub collaborations within a Business Collaboration are defined within this component. Since Business Transaction Activities are already mentioned in the previous section, the remaining definitions will be mentioned in this section.

Roles

When a Business Collaboration is specialized, a Binary Business Collaboration involves two top-level or abstract partner roles only. Multiparty Business collaborations involve more than two partner roles. Business Collaborations are expressed as a set of Business Activities between these roles. Each abstract partner role occupies a specific role when associated with a Business Activity.

```
<Role name="Retailer Ltd." nameID="IDBuyer"/>
<Role name="Manufacturer Ltd." nameID="IDSeller"/>
```

Figure 3.8: Roles Definition within a Business Collaboration

In Figure 3.8, an example definition of the roles which is used by the Business Transaction Activities is displayed. Here, the companies are given an id for being able to be referenced.

Start, Success and Failure States

Start activity indicates the starting state of the collaboration. It might have links to one or more Business Transaction Activities. In the Figure 3.9, a start state with link to one Business Transaction Activity is shown.

```
<Success name="Success" nameID="iSURF_COF_Success"/>
<Failure name="Failure" nameID="iSURF_COF_Failure"/>
<Start>
      <ToLink toBusinessStateRef="iSURF_Send_EC_BTA"/>
</Start>
```

Figure 3.9: Start and Completion States

Two completion states, Success and Failure are defined in the example above. These states give the possibility to define whether a Business Transaction Activity has been performed or not within the specified period. Business Success is true (i.e. the status reported to the choreography is true) when the activity is completed. As stated above, it is mostly used by defining the optional elements in the CPFR Business Process.

Direct Transitions

A direct transition signifies the sequence of the message exchanges, i.e. what message exchange should be expected after a message exchange occurs. The Business Collaboration is either in the state of performing a given Business Activity (or multiple concurrent Business Activities) or waiting to start a Business Activity, unless it has reached a completion state. Once a Business Activity completes a transition from this Business Activity, it navigates to another Business Activity. In Figure 3.10, a transition is specified indicating when "iSURF_Send_EC_BTA" is performed as the next transaction "iSURF_Send_ECR_BTA_B2S" is expected.

```
<Transition nameID="TR0">
      <FromLink fromBusinessStateRef="iSURF_Send_EC_BTA"/>
      <ToLink toBusinessStateRef="iSURF_Send_ECR_BTA_B2S"/>
</Transition>
```

Figure 3.10: A Direct Transition example

Optionality through Condition Guards on Success and Failure

In the CPFR Process there are message exchanges which might or might not happen. One example is the exception notification messages. If for a period of time, there is no exception message, the CPFR Process should continue as usual while if there is one the resolution should be performed.

In Figure 3.11, the decision for an optional message, Termination Message, is shown. Business Transaction Activity of Termination Message already includes a duration specified. If this message arrives within that period, in the above example, the state of the process is selected as "iSURF_COF_Success" which is the successful completion state.

```
<Decision name="Finalize Planning Decision" nameID="D3">
      <Documentation/>
      <FromLink fromBusinessStateRef="iSURF_Send_TM_BTA"/>
      <ToLink toBusinessStateRef="iSURF_COF_Success">
            <ConditionExpression
                  expressionLanguage="ConditionGuardValue"
                  expression="iSURF_COF_Success"/>
      </ToLink>
      <ToLink toBusinessStateRef="iSURF_Send_EC_BTA">
            <ConditionExpression
                  expressionLanguage="ConditionGuardValue"
                  expression="iSURF_COF_Failure"/>
      </ToLink>
</Decision>
```

Figure 3.11: An Optional Transition example

If no message arrives within the period, this means that another loop should begin and "iSURF_Send_EC_BTA" is chosen as the next message exchange.

Conditions

Within a business process, the conditions are the elements directing the flow to different paths based on the decisions. In ebBP, Transitions may also have a Condition Expression element. Condition expression can be bound to variables. Variables are the named information elements that are available to bind concepts across Business Transaction. They also serve to make the semantics clear in a condition expression.

```
<Decision nameID="D1">
      <FromLink fromBusinessStateRef="iSURF_Send_RER_BTA_S2B"/>
      <ToLink toBusinessStateRef="iSURF_Send_RER_BTA_B2S">
            <ConditionExpression
                  expressionLanguage="XPath1"
                  expression="RE_Not_Accepted"/>
      </ToLink>
      <ToLink toBusinessStateRef="iSURF_Send_TILP_BTA">
            <ConditionExpression
                  expressionLanguage="XPath1"
                  expression="RE_Accepted"/>
      </ToLink>
</Decision>
```

Figure 3.12: A Conditional Transition example

In Figure 3.12, a condition example checking whether the retail event has been approved

or not is illustrated. Since based on the approval or the revision decision, there can be two paths; there are two ToLink elements in the decision. The decision makes use of two different variables for these two paths. ebBP definition does not have an if-else structure, and the mutual exclusion of the conditions are left to the modeler. As it can be seen from the variable definitions in Figure 3.13, the only difference between the two variables is the names of the variables and the negation on the condition of the expressions.

In the variable definition, the business transaction of the variable and a reference to the business document are also given. As the expressions, XPath is used for all of the variables.

```
<Variable name="RE Not Accepted"
      nameID="RE_Not_Accepted"
      businessTransactionActivityRef="iSURF_Send_RER_BTA_S2B"
      businessDocumentRef="Retail_Event_Revision_Document">
      <ConditionExpression
            expressionLanguage="XPath1"
            expression="//RetailEvent[@status!='APPROVED']"/>
</Variable>
<Variable name="RE Accepted"
      nameID="RE_Accepted"
      businessTransactionActivityRef="iSURF_Send_RER_BTA_S2B"
      businessDocumentRef="Retail_Event_Revision_Document">
      <ConditionExpression
            expressionLanguage="XPath1"
            expression="//RetailEvent[@status='APPROVED']"/>
</Variable>
```

Figure 3.13: Example Variable Definitions

3.3.5 Business Signals

The type of Business Transaction in an ebBP definition specifies whether a Receipt Acknowledgement and/or an Acceptance Acknowledgement signal are required. These two signals are used in ebBP definition for the following purposes:

- *The Receipt Acknowledgement Business Signal:* signals that a message (Request or Response) has been properly received by the BSI software component. If this property is chosen as true, isIntelligibleCheckRequired reveals that a Receipt Acknowledgement confirm a message only if it has passed structure/schema validity check.

- *The Acceptance Acknowledgement Business Signal:* signals that the received request or response message has been accepted for business processing and that processing is

complete and successful by the receiving application. This is the case if the contents of the business message's Business Documents and Document Envelope have passed a business rule validity check.

In the building blocks defined in ebBP, the definitions for both of these signals (Receipt Acknowledgement Signal and Acceptance Acknowledgement Signal) and for their exceptions are used. All Business Transactions makes use of these four signal definitions which are shown in Figure 3.14.

```
<Signal name="ReceiptAcknowledgement" nameID="ra2">
      <Specification location="http://docs.oasis-
open.org/ebxmlbp/ebbp-signals-2.0"
            name="ReceiptAcknowledgement"
            nameID="rabpss2"/>
</Signal>
<Signal name="ReceiptAcknowledgementException" nameID="rae2">
      <Specification location="http://docs.oasis-
open.org/ebxmlbp/ebbp-signals-2.0"
            name="ReceiptAcknowledgementException"
            nameID="raebpss2"/>
</Signal>
<Signal name="AcceptanceAcknowledgement" nameID="aa2">
      <Specification location="http://docs.oasis-
open.org/ebxmlbp/ebbp-signals-2.0"
            name="AcceptanceAcknowledgement"
            nameID="aabpss2"/>
</Signal>
<Signal name="AcceptanceAcknowledgementException"
        nameID="aae2">
      <Specification location="http://docs.oasis-
open.org/ebxmlbp/ebbp-signals-2.0"
            name="AcceptanceAcknowledgementException"
            nameID="aaebpss2"/>
</Signal>
<Signal name="GeneralException" nameID="ge2">
      <Specification location="http://docs.oasis-
open.org/ebxmlbp/ebbp-signals-2.0"
            name="GeneralException" nameID="gebpss2"/>
</Signal>
```

Figure 3.14: The Business Signal Definitions

3.3.6 The Complete Definition

CPFR Designer Tool builds the definition of the complete CPFR process by combining the components above. The Signals, Business Documents and Business Transactions are the static components since the signal definitions are generic and the types of the documents that

will be used in the CPFR Process are fixed. The definitions within the Business Collaboration (i.e. the Business Transaction Activity definitions and the Choreography definition) are generated automatically by the CPFR Designer Tool. In Appendix A, the partial definition of the CPFR Scenario I is given.

CHAPTER 4

COLLABORATIVE PLANNING PROCESS DEFINITION TOOL

4.1 Introduction

Within the scope of the thesis work, a CPFR Designer Tool has been implemented which in the background uses the ebBP building blocks defined for generating the complete ebBP definition for the designed CPFR Process. The tool is supposed to be used after the partners come together and make a collaboration agreement.

The tool basically gathers the information that is agreed on the collaboration like the product to collaborate, the goals and the metrics of the collaboration. Then, the users are requested for selecting one of the three templates (CPFR Scenario I-II-III) explained in Section 3.2.1. Once a template is selected, the existing ebBP definition of the corresponding business process is loaded and a user friendly graphical designer is displayed allowing the user to customize the process to generate a fully complying process for the collaboration. At the end, when the user saves the newly created business process, the ebBP definition is saved and can be retrieved also later.

For creating an executable business process, on a very similar interface like the designer, the users are requested to enter the WSDL endpoints of their system for each of the message exchanges defined in the CPFR Process. After all completed, by pressing on a button, for each of the partners in the collaboration a WS-BPEL deployment package is generated. When it is deployed, the designed CPFR Process can be started. In this chapter, the details and capabilities of the CPFR Designer Tool are presented.

4.2 Overview of the CPFR Designer Tool

The CPFR Designer Tool has been implemented with Flex [23]. The implementation with Flex has the advantages that the application is:

- web based (accessible from anywhere with internet connection) and works with all major browsers
- cross-platform (can be launched independent of the underlying operating system)
- also executable as a desktop application when needed.

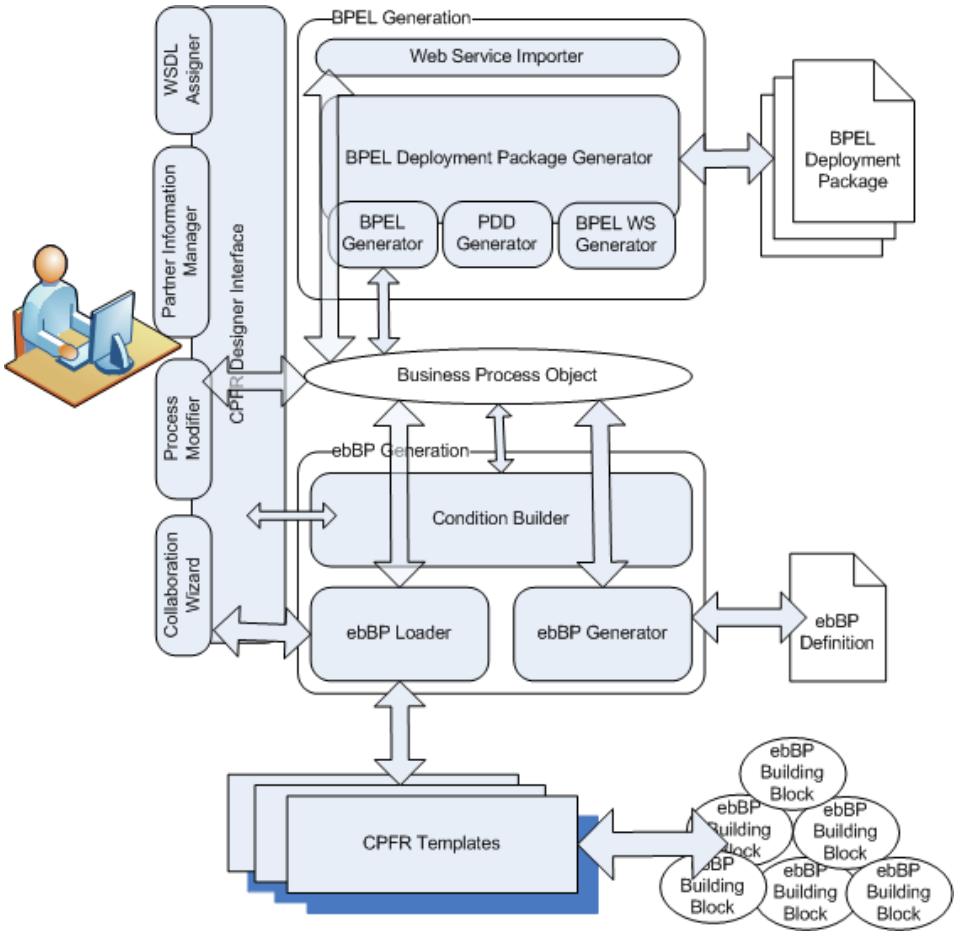

Figure 4.1: The CPFR Designer Tool Overview

The general architecture of the CPFR Designer Tool is given in the Figure 4.1. There are two main functionalities of the system: the Design of CPFR Planning Process together with the ebBP definition and the generation of WS-BPEL files and the deployment package.

The usage of the system is as follows: The users first manage the information belonging to their companies. This includes the company information, the contact people and the products. Then they select designing a new CPFR Planning Process. The partners collaborating, the scenario that will be used and the product to collaborate are selected via the Collaboration Wizard. The corresponding CPFR Planning Process template is loaded from the CPFR Templates (composing of ebBP building blocks) by ebBP Loader and converted to the Business Process Objects. Then the CPFR Designer Interface displays the graphical view of the CPFR Process. The users then add/remove/modify the Message Exchange States (MES) and the message flow. The new conditions might be built by Condition Builder. Once the users finalize the customizing process, ebBP definition of the process is generated and saved.

Now a choreographic definition of the customized business has been generated. To generate the executable WS-BPEL business, the users advance to the WS-BPEL Design phase after finalizing the visual process definition. In this step, the users load the web services for their legacy systems and associate each of the actions on the message exchange points with an operation from the related web service. Once this process is also complete, the WS-BPEL deployment package can be generated by the system. The package includes the WS-BPEL (.bpel) files, the Process Deployment Descriptor (.pdd) files of ActiveBPEL and the WSDL files (describing the operations of the bpel processes) for each of the collaborating partners. This deployment package can be deployed to an ActiveBPEL engine [3], an open source BPEL engine, and from their existing systems the planning process can be started.

In the below sections, the implementations of the ebBP Generation Module, CPFR Designer Interface and WS-BPEL Generation module are described.

4.3 ebBP Generator Module

4.3.1 ebBP Loader

ebBP definitions are the key definitions in CPFR Designer Tool since both the CPFR Templates and the customized CPFR Processes are kept as ebBP definitions. There are three CPFR Process Templates corresponding to the three scenarios described in Chapter 3:

- Buyer-led Forecasting (Scenario I): Buyer owns the responsibility of both Sales Forecast and Order Forecast.
- Shared Forecasting (Scenario II): Buyer owns the responsibility of Sales Forecast and

the Seller owns the responsibility of Order Forecast.

- Seller-led Forecasting (Scenario III): Seller owns the responsibility of both Sales Forecast and Order Forecast.

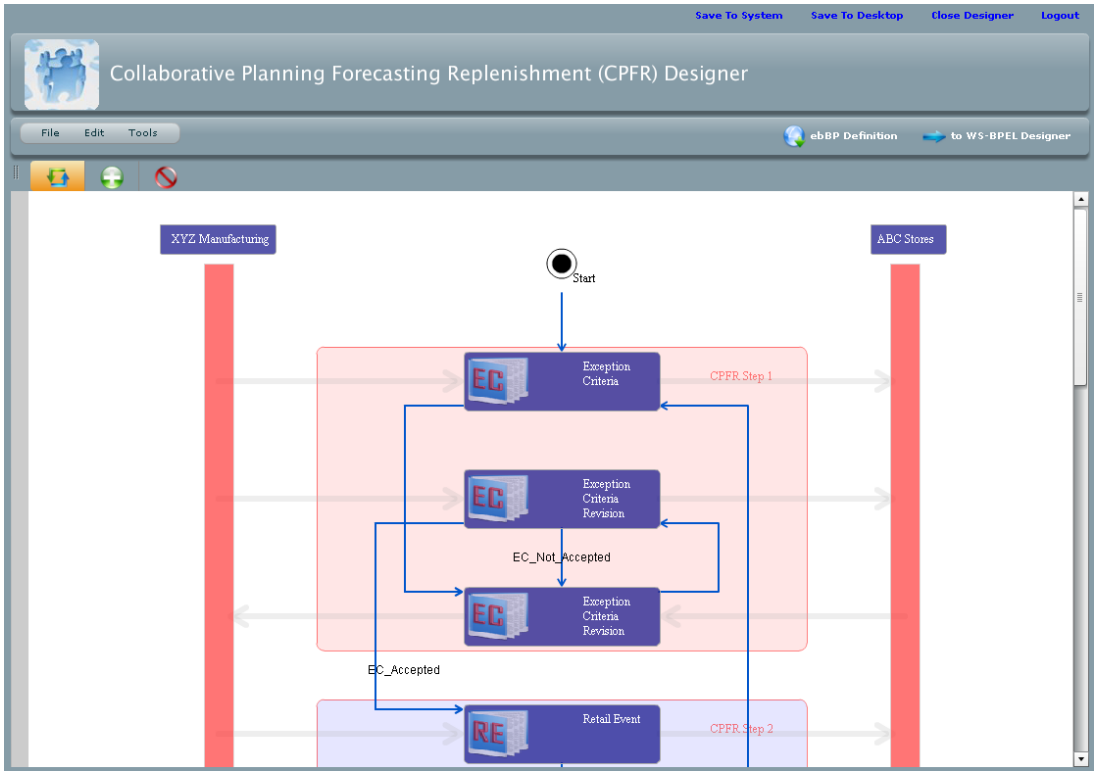

Figure 4.2: CPFR Designer Tool Main View

According to the selection of the scenario, the appropriate ebBP definition is loaded from the server by ebBP Loader Module. The ebBP Loader Module first reads the Business Transaction Activities together with its Business Transaction and Business Document references. Then the choreography of the collaboration is read and Business Process Objects are built which is used by all other components in CPFR Designer Tool. The main view of the application after CPFR Scenario I is chosen is shown in Figure 4.2. When the graph is displayed, the CPFR Process is ready to be customized.

4.3.2 Customizing CPFR Process Template

The CPFR Process Template can be customized by addition, deletion or editing of the message exchanges or their choreography. For defining the choreography, a more intuitive

and easier interface, there are no elements indicating the loops or conditions. Instead, the user can put any number of links between message exchange states. If there is more than one outgoing edge from a message exchange state (MES), the user is asked to enter a condition for those edges. At the end the loops and the conditions are detected automatically by the system.

There are three modes for customizing the CPFR Process Template: Edit/Link Mode, Add MES Mode, and Delete Mode. These modes are activated from the left upper side of the application which is shown in Figure 4.3.

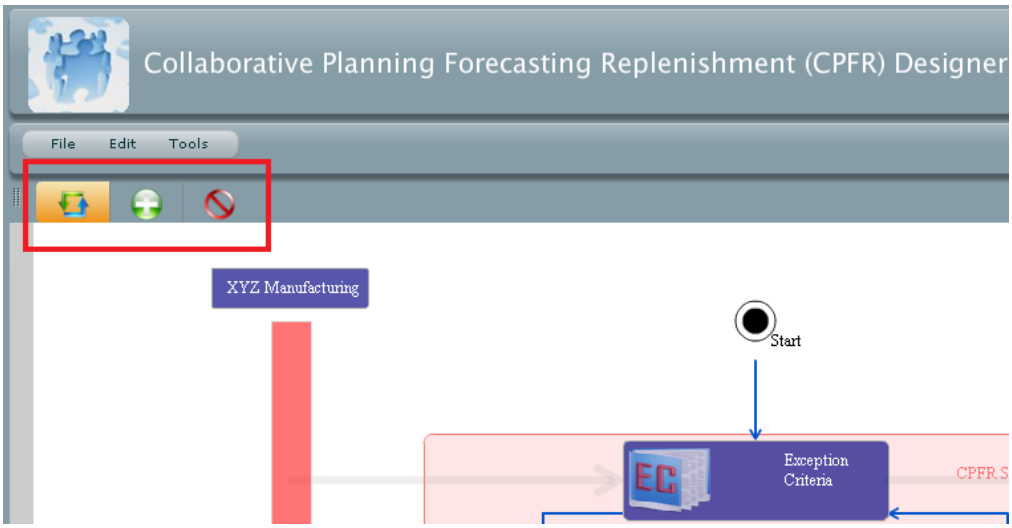

Figure 4.3: CPFR Designer Tool Modes

Edit/Link Mode:

When this mode is active, the user can edit the information of a message exchange state, add new links between the MESs, and edit the conditions of the links.

To view and edit the information of a message exchange state, the user should double click on a message exchange. Then the dialog shown in the Figure 4.4 is displayed.

The name of the Message Exchange State is the descriptive name for the message. This is used for the messages like Product Activity which can hold both Point of Sales (POS) Data and Distribution Center (DC) data. This name is used for the users to differentiate the content of the messages in those situations.

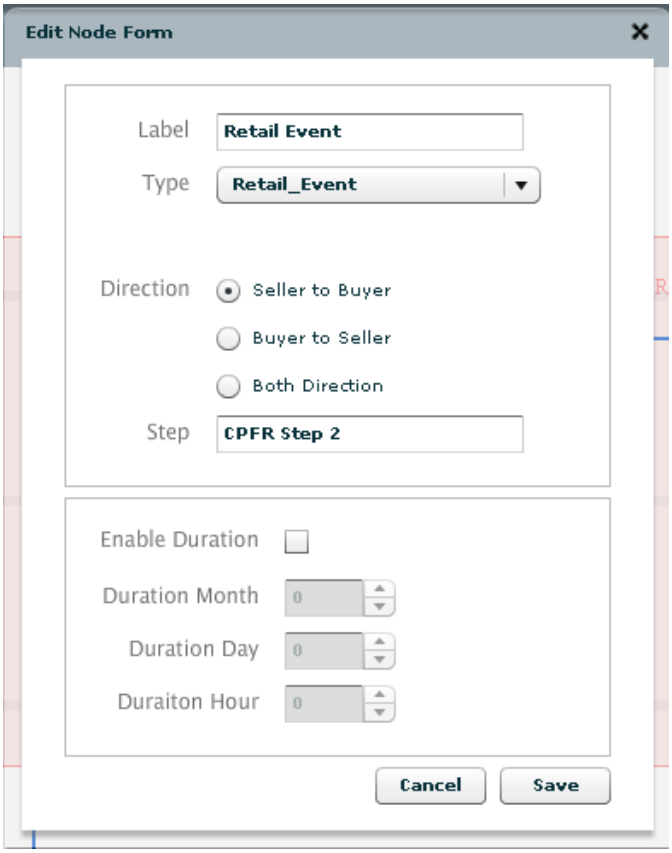

Figure 4.4: Message Exchange State Editing Dialog

The type of a Message Exchange State as parallel to the message types described in Section 3.2.1 can be any of the following:

- Sales Forecast
- Sales Forecast Revision
- Order Forecast
- Order Forecast Revision
- Product Activity
- Performance History
- Exception Notification
- Exception Criteria
- Trade Item Location Profile

- Retail Event
- Inventory Activity Or Inventory Status
- Purchase Conditions

As it can be seen from the list, separate types for Sales Forecast and Order Forecast are given as the message type while there is only one corresponding message type Forecast. The aim is to signify that these two forecasts are separate entities.

The next property of the message exchange state is the direction of the exchange. It is either "From Buyer to Seller", "From Seller to Buyer" or "On Both Directions". The last option, "On both directions" is valid only if the message is optional. This is usually modeled on the Exception Notification messages as who will send the notification or whether one side will send is determined at runtime.

Finally, a duration period can be specified for a message indicating how much time should be waited for the message exchange state. When specified, this duration can make the message optional if the conditions are adjusted accordingly.

If the user makes a single click on a message exchange state, an arrow is started to be drawn. With the next click on another node, if there is no edge in that direction between those message exchange states, a new edge is drawn. In this way, new condition paths can be added to the existing message states or direct paths can be converted to a conditional path.

If the user double clicks on an edge, a condition editor is displayed. This is a user friendly editor where the user can select the message and the content to be used while evaluating the condition for a specific edge. The condition function is also set with this dialog.

Add New MES Mode:

When the MES addition mode (the second button of the mode selection button group) is chosen, the available message types for addition are displayed to the user as shown in Figure 4.5. The list of the messages is the same as the list given on above section.

When any of the messages dragged over the graph, the available places where the message can be put are highlighted in yellow. When the user drops the dragged message, it is placed on the heighted area and for setting its properties a dialog like on the Edit MES Mode is displayed. After the addition of a new message state, initially there is no connection with the other message states. They can be linked with in the edit mode as described above. A

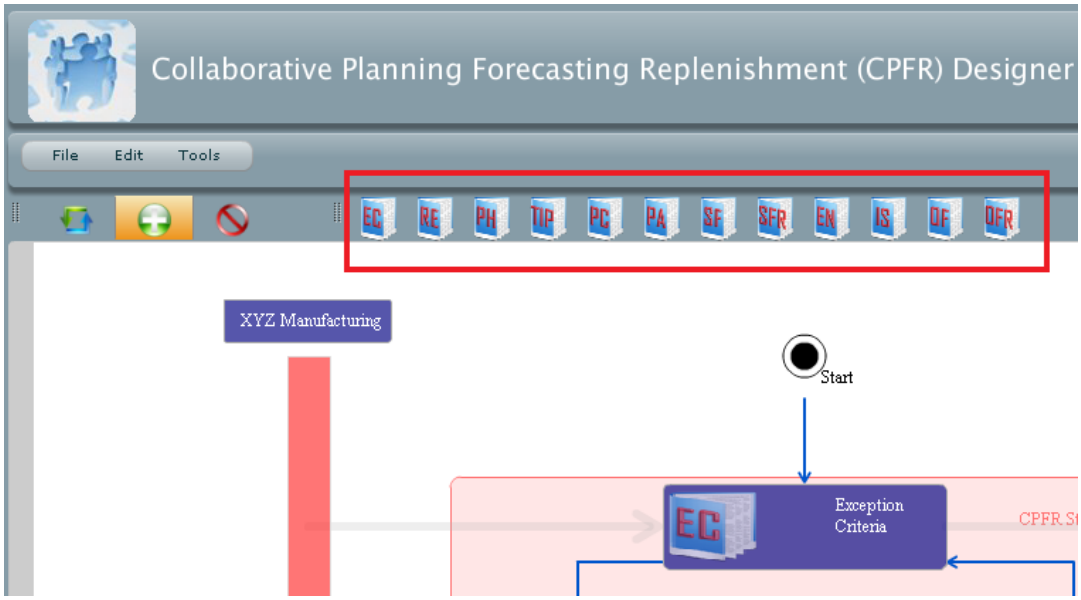

Figure 4.5: Add New Message Exchange State Mode

snapshot of the CPFR process graph during the addition of a message state is shown below in Figure 4.6.

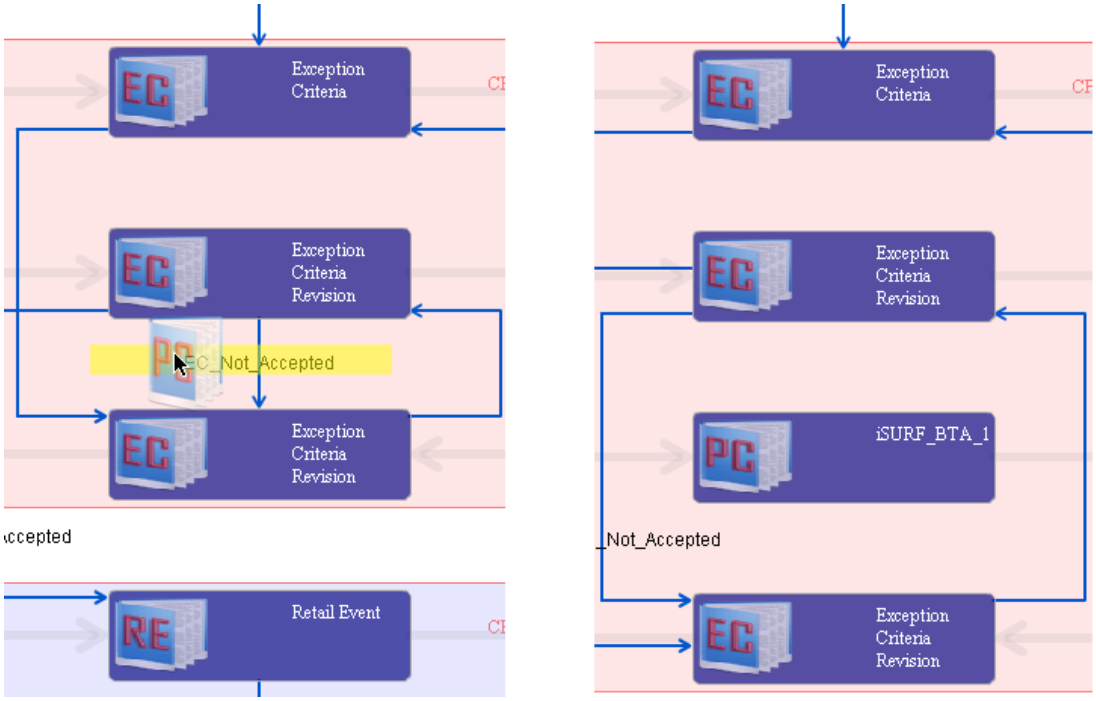

Figure 4.6: Adding a Message Exchange State

Delete Mode:

The last mode for the customization of CPFR Graph is deletion mode. With this mode is selected, the user can delete the message exchange states or their connections. On the single click in Deletion Mode, a confirmation for deleting the selected object is displayed.

Deletion of links are straightforward, the selected link is deleted. If only one conditional link remains, in the background, the connection type is converted to direct connection. Just before saving the CPFR Process, if there are direct types with a condition, the user is warned that the conditions will take no effect since it is a direct connection.

If a MES is selected to be deleted, the message itself and all connections outgoing and incoming to this MES are deleted. The same checks on the deletion of the links are done for all of the deleted links.

With all of these three modes (Edit/Link, Addition, Deletion), the CPFR process can be customized so that the process is fully adjusted to the needs of the collaborating parties.

4.3.3 ebBP Generator

After the customization of the CPFR Process is completed, the process can be saved by generating its ebBP definition. This is achieved by the ebBP Generator Module of the CPFR Designer Tool. The dialog shown in Figure 4.7 is displayed when the user chooses to generate ebBP definition by clicking the "ebBP Definition" button on the right upper corner of the screen. Please see Appendix A for a sample ebBP definition generated by the CPFR Designer.

With the same dialog it is also possible to load a previously generated ebBP definition. There is no need to additional file since the CPFR Process Graph is directly drawn with the information within the generated ebBP definition.

4.4 Business Process Object

As it can be seen from the general overview of the application in Figure 4.1, all components uses the Business Process Object for accessing the CPFR Process information, hence, in the CPFR Designer Tool, the Business Process Objects are the core information objects.

The class diagram of Business Process Object and its related classes are given in Figure 4.8. The classes and their descriptions are as follows:

- Binary CPFR Process: This is the main class representing a binary CPFR Business

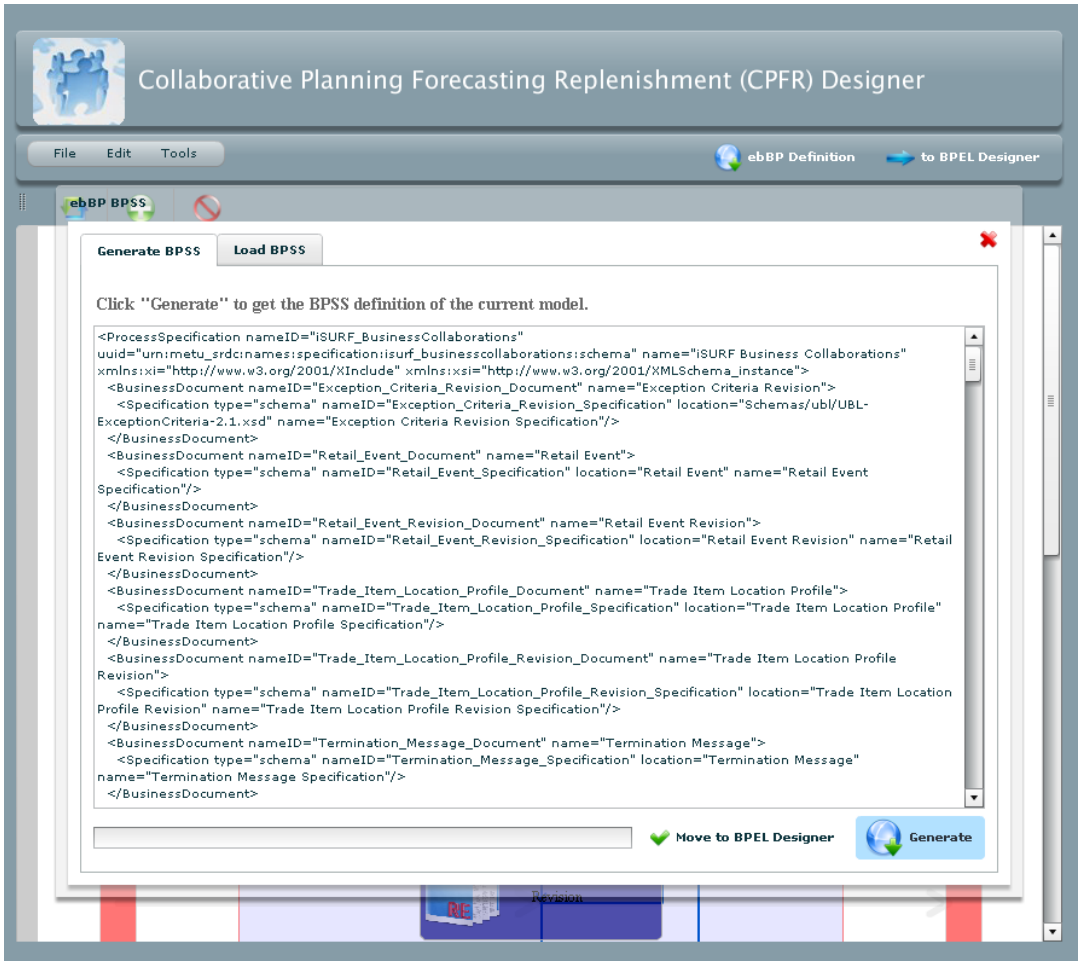

Figure 4.7: ebBP Definiton Generation

Process Object. It includes two Business Partners, a list of CPFR Messages, a starting CPFR Message, a reference to ISU Configuration, name, success and failure states.

- Business Partner: This class represents the each participant in the CPFR Process. It has a role and a reference to the Company class for the detailed information of the company it is representing.

- CPFR Message: This class represents the each CPFR MES in the CPFR Designer Tool. It has the properties of a MES and a reference to the Message Connection class for the links it has to other MESes. It also has operation class for holding the operation assignments to be used during the WS-BPEL generation.

- Message Connection: This class represents the links between the CPFR Messages. It can have zero or more links. It has a type which can be any of the following: DIRECT, OPTIONAL, CONDITION. Each of the links contained is represented with Expression

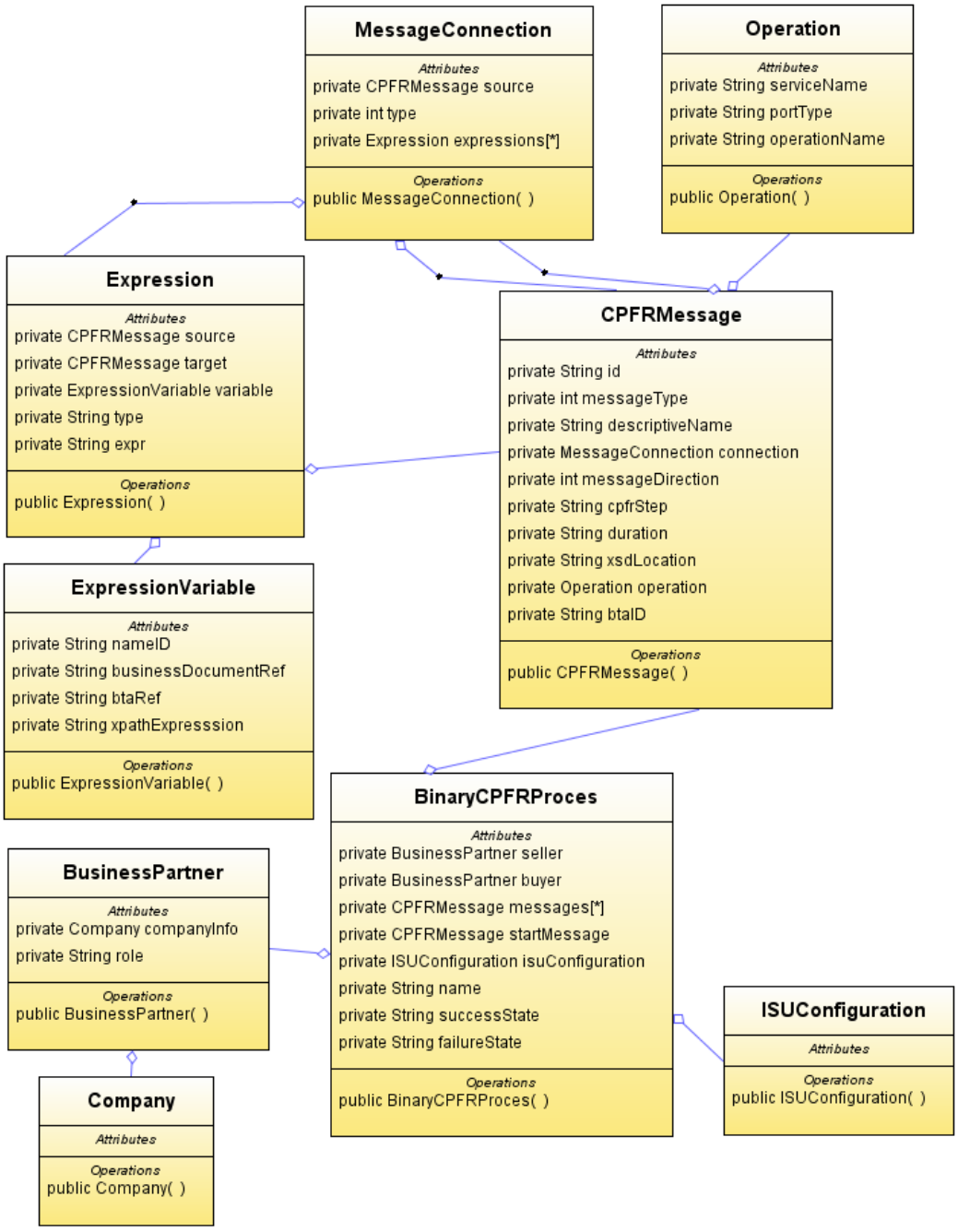

Figure 4.8: Class Diagram of Business Process Object

class.

- Expression: This class represents a single link between two Message Connections. It might or might not have a condition. If it has a condition, it might or might not have an expression variable. If there is an expression variable, this means that this is a

branch of a condition. If it has only condition but not an expression variable, then this means it is a branch of optional decision.

- Expression Variable: This class represents the each variable in an ebBP definition. On the CPFR Graph, the name of the variable is displayed on the middle of the edge containing it. The XPath expression of the variable is built visually by the Condition Builder component.

- Operation: Before a WS-BPEL definition is generated, each message should have an associated operation. During the generation of the WS-BPEL definition, the name of the operations, its port type and service name are used.

- Company: This is the class having the full company information. It has all basic information of the company like name, address, phone, contact people, and the expertise. The information is gathered on the registration of the company for the first time.

- ISUConfiguration: As mentioned in Section 1, the CPFR Planning Process Designer Tool has interfaces with the iSURF Interoperability Service Utility (ISU). During the execution of the planning process, this component provides the semantic and syntactic transformation of the messages if the message formats of the sending and receiving collaborators are different. This class provides the configuration information of ISU which can be used during the WS-BPEL definition generation.

4.5 WS-BPEL Deployment Package Generator

The ebBP definition generated is not an executable process model. It is the choreography definition of the customized CPFR Planning Process. CPFR Designer Tool also provides a means of generating executable process definition from the customized CPFR process definition conformant to the WS-BPEL standard. In this way, without a need to knowing the details of WS-BPEL specification, the users of the CPFR Designer Tool can generate the executable processes.

However, additional information is needed for being able to generate the WS-BPEL definition. The message exchange states should be bound to web service operations so that the existing systems of the companies can communicate with the generated business process through web services.

When the design of the CPFR Process is completed, to generate WS-BPEL definitions, the customization of the CPFR Process is frozen by going to the WS-BPEL generation view.

This is done by clicking on the "to WS-BPEL Designer" button on the upper right corner of the screen as seen in the Figure 4.9.

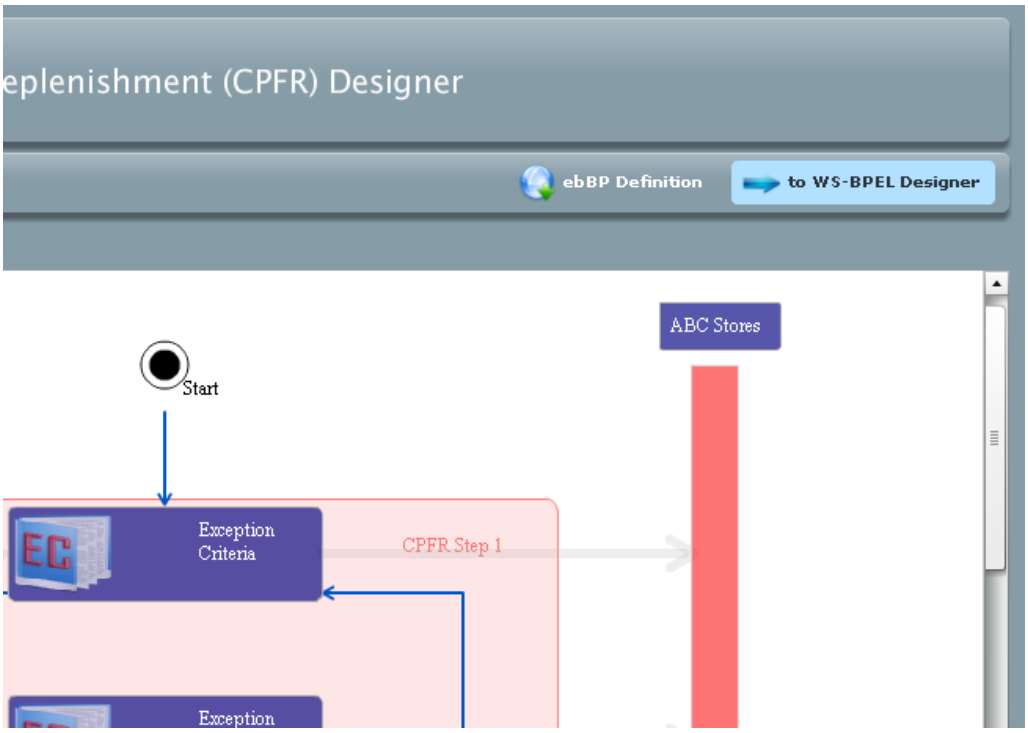

Figure 4.9: Moving to the WS-BPEL View

After the view is changed to WS-BPEL Designer view, on the left side of the screen, there will appear web services lists for Buyer, Seller and ISU. While Buyer and Seller may have more than one service; the ISU will have only one service. This restriction saves the user from selecting a service and an operation of ISU for each of the messages.

Both retailer and manufacturer add the services wrapping the functionalities of their existing enterprise system to their lists on the CPFR Designer Tool. These web services will be created by a Legacy Wrapping Tool in iSURF. Since it is not implemented yet, in this work, manually developed services are used. However, from the point of CPFR Designer Tool, there will be no differences since they are imported to the system in the same way. The dialog for importing the web services to the CPFR Designer is shown in Figure 4.10. The name of the service, the port types and the operations under each port type is displayed as in the figure when the Load button is pressed. If it is planned to be used, the ISU Service is also set in the same way.

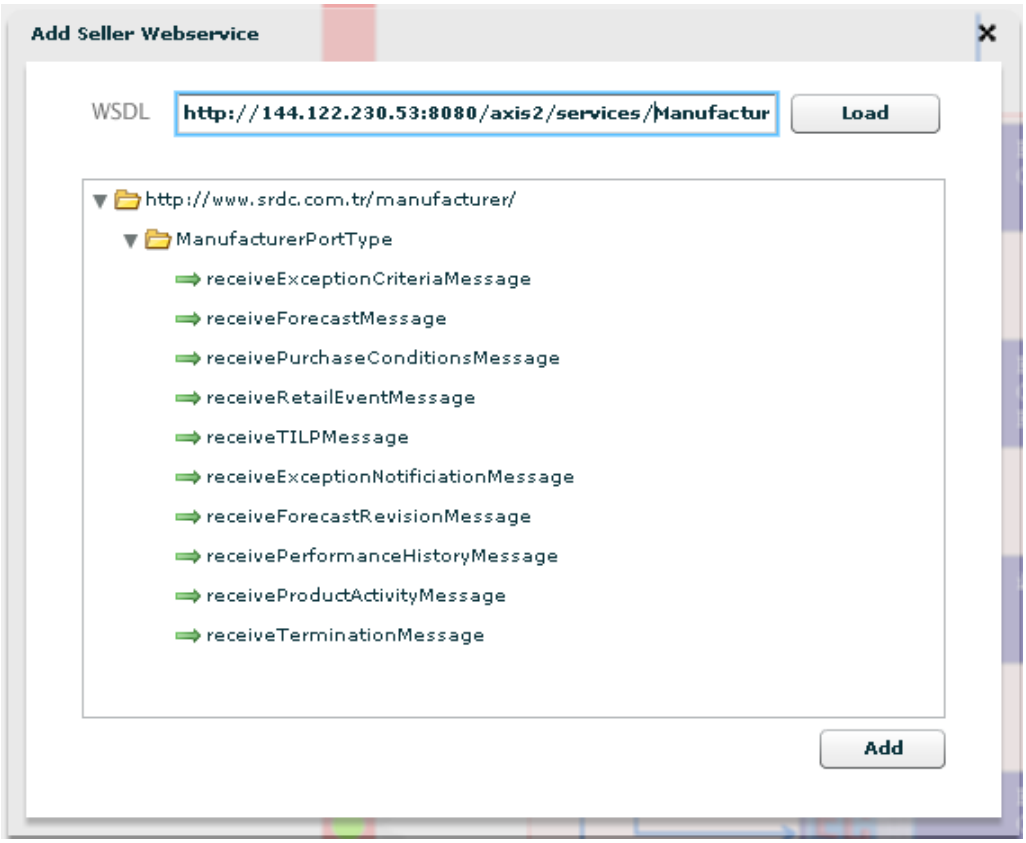

Figure 4.10: Importing Web Services to the CPFR Editor

After the web services are added, there only remains assigning an operation to the message exchange states. The WS-BPEL view of the CPFR Process Designer Tool can be seen in the following figure (Figure 4.11). The points that need to be set are the receiving sides of each message exchange state (shown in green in the designer). When clicked on these points, the users can associate operations to the messages exchange states on a very similar dialog used for loading the services. The interface for setting the web service operations to the endpoints on message exchanges are illustrated in Figure 4.12.

Please note that as the CPFR Process is frozen in this mode, no change (edition of the messages or links) can be done on the CPFR Process itself. The reason is to make sure that the two definitions (i.e. the ebBP definition and BPEL Definition) are kept consistent and parallel. If the users want to change a part of the process, they go back to the ebBP Designer view by clicking the button which was used to move to the WS-BPEL Designer view. Then, they make the changes, generate the ebBP definition again and continue with the WS-BPEL view.

After the endpoints of each message exchange state is associated with a web service,

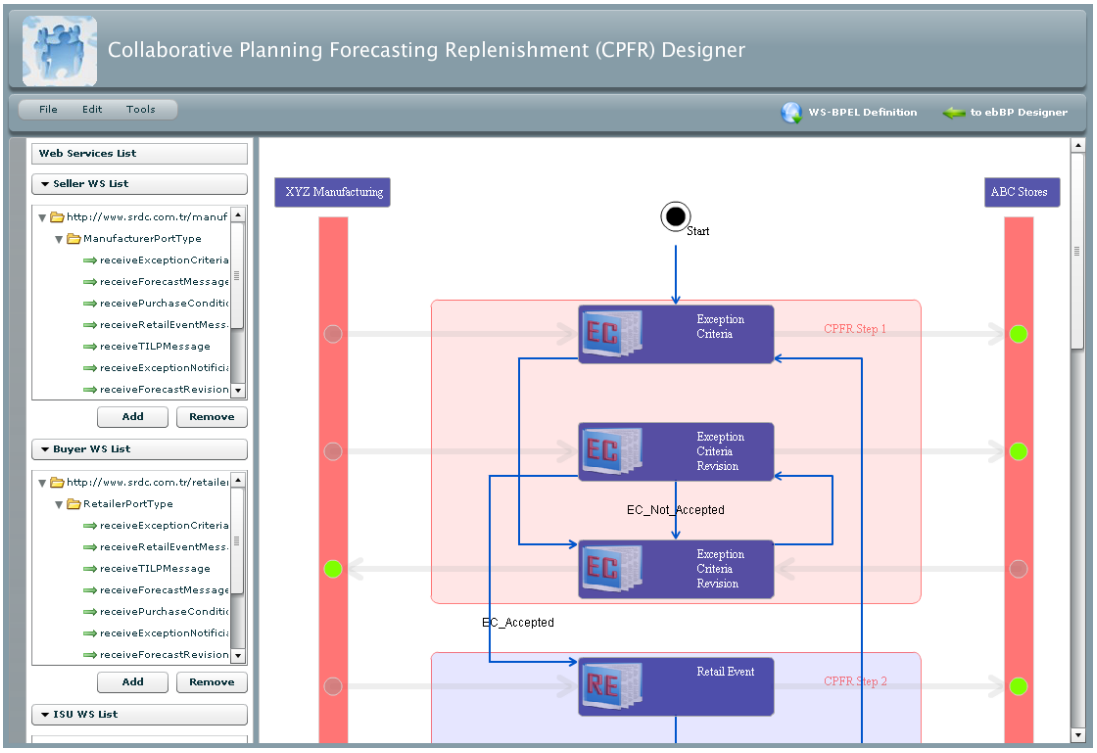

Figure 4.11: WS-BPEL View

WS-BPEL and other related files for the deployment can be created with the "WS-BPEL definition" button on the upper right corner of the screen. The dialog on the following figure (Figure 4.13) will appear. For each of the partners in the collaboration a separate WS-BPEL definition will be generated since WS-BPEL standard defines an orchestration definition. Here it is good to remember that orchestration always represents control from one party's perspective by which differs from the choreography, which is more collaborative and allows each involved party to describe its part in the interaction. This is why a single ebBP definition is generated for the whole business process.

As the executable business process definition, only WS-BPEL definitions are enough. However, in order to be able to execute that definition, each engine requires some additional information related to the new services which will be exposed by the WS-BPEL process. In this work, ActiveBPEL Community Edition Engine [3] is selected as the WS-BPEL Engine. ActiveBPEL Engine requires .pdd (process deployment descriptor) files along with the standard .bpel files. Also partner links for the used services and the operations of the service exposed by the WS-BPEL processes are described in a .wsdl file (Please see Chapter 5 for more details.). All these files are generated at once on the dialog above. WS-BPEL Deployment Package Generator has sub-components for building these additional PDD and

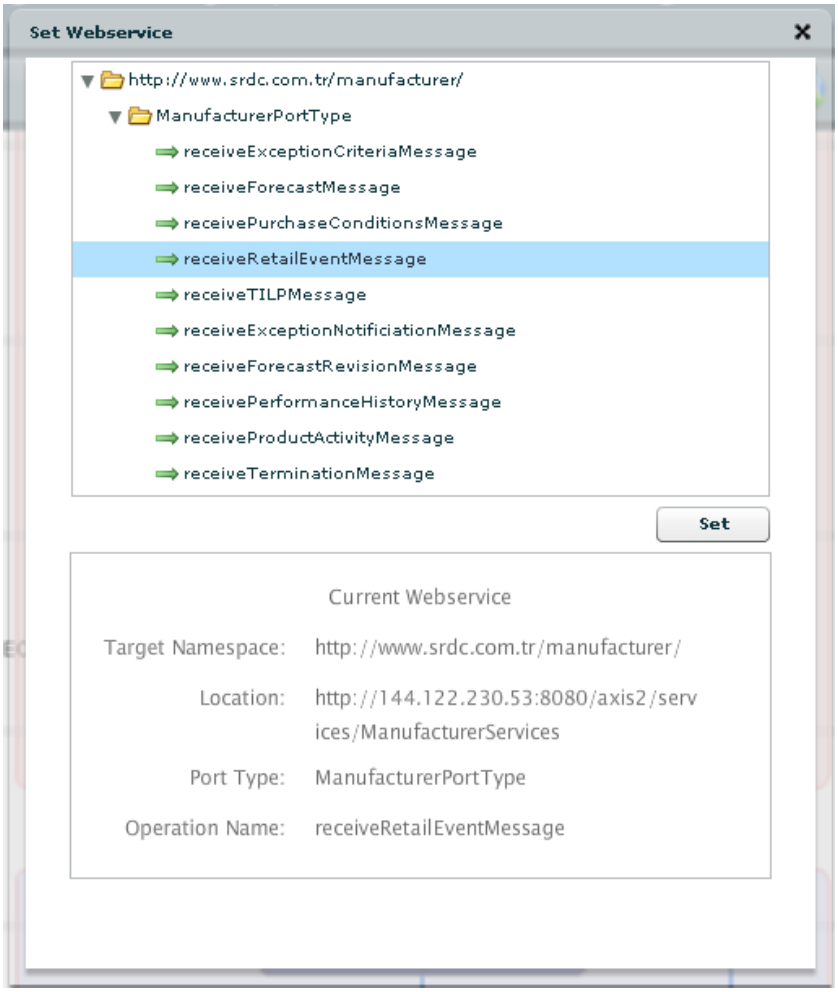

Figure 4.12: Dialog for setting operations for WS-BPEL Generation

WSDL files.

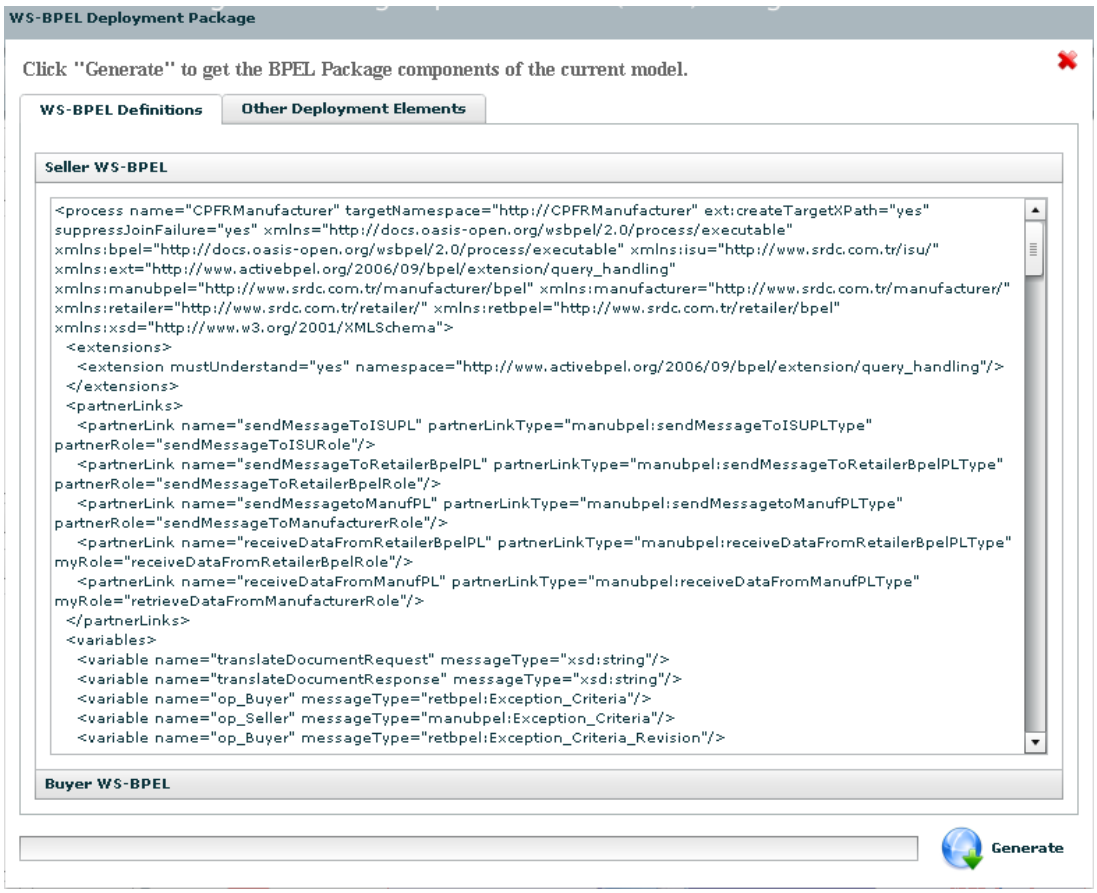

Figure 4.13: WS-BPEL Generation Dialog

CHAPTER 5

COLLABORATIVE PLANNING PROCESS EXECUTION ENVIRONMENT

In the CPFR Designer Tool, using the Business Process Object (which are built from ebBP definitions), an executable business process definition in WS-BPEL is generated. This definition is in fact in the form of a deployment package which can be directly deployed to the WS-BPEL engine and be started. The existing applications (like ERP and VMI) of the collaborating companies participate to the CPFR Planning process through their Web Services which were assigned during the generation of WS-BPEL definition in the CPFR Designer Tool. In this chapter, the generation and the execution of WS-BPEL process are described.

5.1 Generation of WS-BPEL Deployment Package

Based on the WS-BPEL specification, the business process definitions are generated by the CPFR Designer Tool. However, for it to be deployed to the ActiveBPEL, additional deployment files are also needed. First of all, for each WS-BPEL process that will be deployed to ActiveBPEL, a Process Deployment Descriptor (.pdd) file is needed to be created. This XML file tells the ActiveBPEL engine about the BPEL processes. Each process (each .bpel file) has its own .pdd file. Therefore, a .pdd file is generated for each of the partners participating into the collaboration. As mentioned in the enabling technologies section, each WS-BPEL process is exposed as a web service itself. For this reason, a WSDL file is required which describes the public entry and exit points of the process. In this section, the details of the generating these files are explained. A sample WS-BPEL definition generated by CPFR Designer Tool for Manufacturer is given in Appendix B.

5.1.1 Generation of WS-BPEL Definitions

Before starting to give the details of generating the WS-BPEL definitions, it is good to have a look at the infrastructure of the message exchanges via WS-BPEL processes. First of all, there is a separate WS-BPEL process for each of the partners in the collaboration. Basically, each WS-BPEL process receives messages from either the web services of its company or from another WS-BPEL process. In the Figure 5.1 below, the flow of the two messages is depicted, a message sent from the Seller to the Buyer and then a second message sent from the Buyer to the Seller.

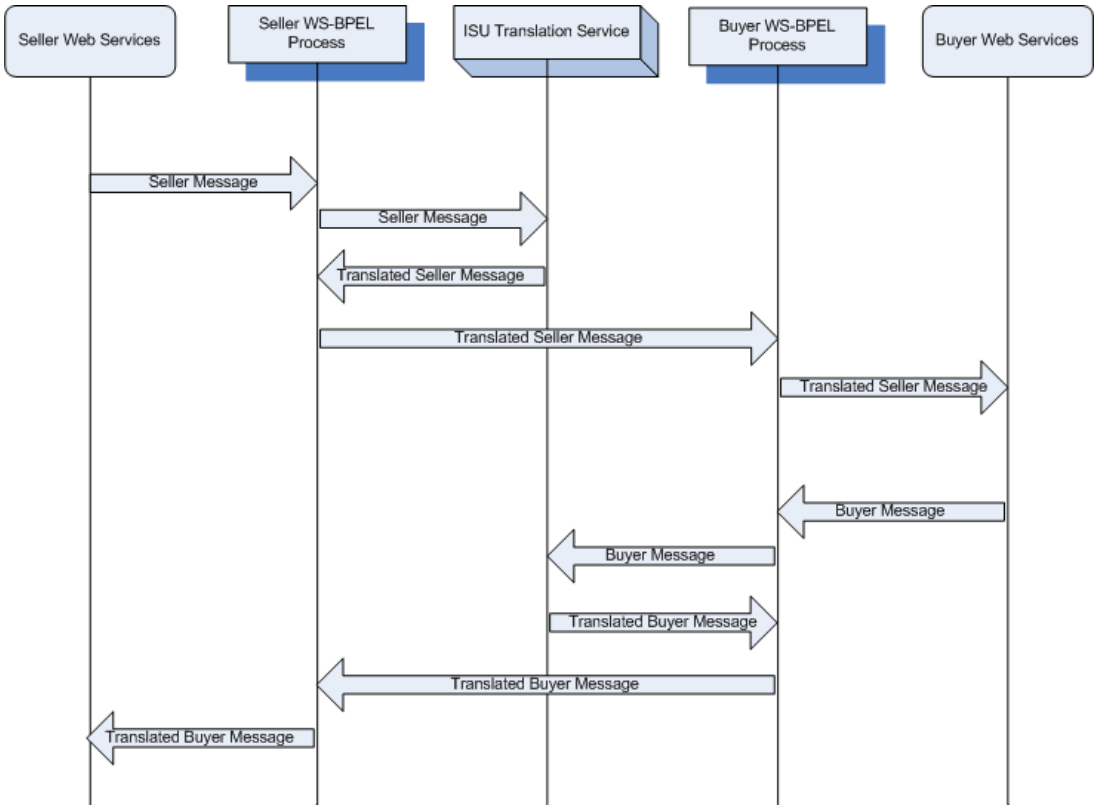

Figure 5.1: Messaging Infrastructure through WS-BPEL Processes

As seen in the figure, if a message is going to be sent from the Seller to the Buyer, the Seller WS-BPEL Process waits a message from the Seller Web Service. In fact this is the reason why we call each of the messaging elements as Message Exchange State; that message does not have to be sent immediately, it can be sent anytime. Therefore, it is better to say that the processes are waiting on a message state.

Once the message is received from the Seller Web Service, in case a message transformation is needed, it is sent to the ISU Translation service for the message to be converted to receiver's format. Afterwards, the message is forwarded to the Buyer WS-BPEL process again by the Seller WS-BPEL process. Finally, the Buyer WS-BPEL process passes the message to the Buyer Web Services. On the way back, while the message is sent from the Buyer to the Seller, the same procedure is followed as the previous message from the Seller.

Please note that the web services of the Buyer and the Seller provide the functionalities of their existing enterprise systems to the BPEL Processes. In this way, the planning messages are generated and received by the Enterprise Applications of the companies. The users will continue to use their systems without any noticed change for them.

From the figure, it can easily be noticed that the WS-BPEL services will receive messages from either the web services of its side, or from other WS-BPEL processes. Since the message format differences are handled by the sender before sending a message, all of the messages received will be in the format of the receiver partner.

Process Definition

As described in WS-BPEL specification, a WS-BPEL process starts with a <process> tag and it can include <extensions>, <partnerLinks>, <variables> and <sequence> within this <process> tag. The template that is used while producing the WS-BPEL processes with the CPFR Designer Tool is shown in Figure 5.2. The extensions element by default contains an extension stating that this business process should be run by an engine understanding the activeBPEL namespace.

Other than the usual attributes and namespace definitions, additional namespaces are defined which are used later within the process. These are:

- *xmlns:isu* is the namespace referencing the translation service of the iSURF ISU.
- *xmlns:manufacturer* is the namespace referencing the web services of the manufacturer.
- *xmlns:retailer* is the namespace referencing the web services of the retailer.
- *xmlns:manubpel* is the namespace referencing the web service that is exposed by the manufacturer WS-BPEL definition.
- *xmlns:retbpel* is the namespace referencing the web service that is exposed by the retailer WS-BPEL definition.

The content of the partnerLinks, variables and sequence is explained below.

```
<process name="CPFRManufacturer"
      targetNamespace="http://CPFRManufacturer"
      ext:createTargetXPath="yes"
      suppressJoinFailure="yes"
      xmlns="http://docs.oasis-
      open.org/wsbpel/2.0/process/executable"
      xmlns:bpel="http://docs.oasis-
      open.org/wsbpel/2.0/process/executable"
      xmlns:isu="http://www.srdc.com.tr/isu/"
      xmlns:ext="http://www.activebpel.org/2006/09/bpel/extensi
      on/query_handling"
      xmlns:manubpel="http://www.srdc.com.tr/manufacturer/bpel"
      xmlns:manufacturer="http://www.srdc.com.tr/manufacturer/"
      xmlns:retailer="http://www.srdc.com.tr/retailer/"
      xmlns:retbpel="http://www.srdc.com.tr/retailer/bpel"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema">

      <extensions>
            <extension mustUnderstand="yes"
                  namespace="http://www.activebpel.org/2006/09/
            bpel/extension/query_handling"/>
      </extensions>
      <partnerLinks>
            .
            .
      </partnerLinks>
      <variables>
            .
            .
      </variables>
      <sequence>
            .
            .
      </sequence>
</process>
```

Figure 5.2: WS-BPEL Definition Template

PartnerLinks

In a WS-BPEL definition, the <partnerLinks> section defines the different parties that interact with the business process in the course of processing the order. In the below, Figure 5.3 shows an example partnerLinks definition generated by CPFR Designer Tool for the seller (manufacturer) process definition. The five <partnerLink> definitions shown here correspond to the senders of the messages (Manufacturer Web Service and Retailer WS-BPEL Process), as well as the three service providers (iSURF ISU Service, Manufacturer Web Service and Retailer WS-BPEL Process). In fact, these services provide the message receiving points.

Each <partnerLink> is characterized by a partnerLinkType and either one or two role names. This information identifies the functionality that must be provided by the business

```
<partnerLinks>
  <partnerLink name="sendMessageToISUPL"
    partnerLinkType="manubpel:sendMessageToISUPLType"
    partnerRole="sendMessageToISURole"/>
  <partnerLink name="sendMessageToRetailerBpelPL"
    partnerLinkType="manubpel:sendMessageToRetailerBpelPLType"
    partnerRole="sendMessageToRetailerBpelRole"/>
  <partnerLink name="sendMessagetoManufPL"
    partnerLinkType="manubpel:sendMessagetoManufPLType"
    partnerRole="sendMessageToManufacturerRole"/>
  <partnerLink name="receiveDataFromRetailerBpelPL"
   partnerLinkType="manubpel:receiveDataFromRetailerBpelPLType"
   myRole="receiveDataFromRetailerBpelRole"/>
  <partnerLink name="receiveDataFromManufPL"
    partnerLinkType="manubpel:receiveDataFromManufPLType"
    myRole="retrieveDataFromManufacturerRole"/>
</partnerLinks>
```

Figure 5.3: WS-BPEL PartnerLinks Definition Example

process and by the partner service for the relationship to succeed, that is, the port types that this process and the partner need to implement. The partnerLinkTypes are defined in the additional WSDL file by retrieving the portTypes from the services assigned during the WS-BPEL generation in CPFR Designer Tool.

Variables

The <variables> section defines the data variables used by the process, providing their definitions in terms of WSDL message types, XML Schema types (simple or complex), or XML Schema elements. In the WS-BPEL process variables allow processes to maintain state between message exchanges. Below in Figure 5.4, a part of the variables definition which was generated by the CPFR Designer Tool for the manufacturer is shown. At the beginning, there are two messages (translateDocumentRequest and translateDocumentResponse) of string type which will be the input and output to the ISU translation service. A variable definition is generated for each of the Messages that is planned to be exchanged. Moreover, if the message is sent from this WS-BPEL process to other WS-BPEL process, also another variable definition is created since the message structure of the receiver might be different. These variables can be also used for the conditions within the sequence of the process.

Finally, if there are optional messages (the message that are allowed to both be sent and not to be sent), a variable of Boolean type is also added to the variables list. This variable is used to detect if the message arrived or not on the expiration of the defined period. "iSURF_Send_TM_BTA_arrived_Seller" is this kind of variable in the above definition.

```
<variables>
      <variable name="translateDocumentRequest"
            messageType="xsd:string"/>
      <variable name="translateDocumentResponse"
            messageType="xsd:string"/>
      <variable name="iSURF_Send_EC_BTA_Buyer"
            messageType="retbpel:Exception_Criteria"/>
      <variable name="iSURF_Send_EC_BTA_Seller"
            messageType="manubpel:Exception_Criteria"/>
      <variable name="iSURF_Send_RE_BTA_Buyer"
            messageType="retbpel:Retail_Event"/>
      <variable name="iSURF_Send_RE_BTA_Seller"
            messageType="manubpel:Retail_Event"/>
      <variable name="iSURF_Send_TM_BTA_Buyer"
            messageType="retbpel:Termination_Message"/>
      <variable name="iSURF_Send_TM_BTA_Seller"
            messageType="manubpel:Termination_Message"/>
            .
            .
            .
      <variable name="iSURF_Send_TM_BTA_arrived_Seller"
            type="xsd:boolean"/>
</variables>
```

Figure 5.4: WS-BPEL Variables Definition Example

Sequence

In the generated WS-BPEL process, the structure of the main activity of the process is defined by the outer <sequence> element, which states that all of the activities contained inside are performed in order. Before the sequence of the process is built, a pass over the Business Process Definition is performed for detecting the loops. During the generation of this processed definition there are mainly three different cases: handling of normal messaging, handling of loops and handling of the optional messages.

Handling of Normal Messaging in WS-BPEL

As mentioned above in this section, the receiving and sending pattern of the WS-BPEL processes are fixed. If the message direction is from the process to the other process, the current process receives the message from the service of its company, it sends the received message to the ISU translation service for possible transformation of the message and finally send it to the other process. If the direction is from other process to this process, it simply receives the message from the other process and passes it to the web service of its company.

An example definition for manufacturer (seller) is given for the second case in Figure 5.5. Here the manufacturer was supposed to receive a message from the retailer. As it can be

```
<receive name="receive_iSURF_Send_ECR_BTA_B2S"
      partnerLink="receiveDataFromRetailerBpelPL"
      operation="iSURF_Send_ECR_BTA_B2SfromRetBPEL"
      variable="iSURF_Send_ECR_BTA_B2S_Seller"/>
<invoke name="InvokeManufBPEL"
      partnerLink="sendMessagetoManufPL"
      operation="iSURF_Send_ECR_BTA_B2S"
      inputVariable="iSURF_Send_ECR_BTA_B2S_Seller"/>
```

Figure 5.5: Direct Messaging Example for the Incoming Messages

seen from the definition, the process waits a message from other process with the <receive> activity. The message is assigned to the iSURF_Send_ECR_BTA_B2S_Seller variable. The message will be in the format of the Seller as indicated in the name of the variable. Therefore, the process directly invokes the manufacturer web service giving iSURF_Send_ECR_BTA_B2S_Seller variable as an input.

In Figure 5.6, the definition for manufacturer (seller) for the messages that should be sent to the other party is shown. The definition again starts with a receive activity. The message is received from the Manufacturer Web Service in the format of the manufacturer. Differently from the previous case, an assign activity is realized so that the input to the translation service of ISU is prepared. For this translateDocumentRequest variable is used. Then, a request-response invocation is performed by sending the transformation request to the ISU web service. The returned value (translated message) is assigned to a variable in the format of the retailer. Finally, the retailer WS-BPEL process is invoked and this case is completed.

Handling of Loops in WS-BPEL

While the direct messages are handled on the detected loop starting points before the normal definition explained above is included in the process, a repeatUntil activity and a following sequence activity is placed as shown in Figure 5.7. Until a condition message is met, all the definition is placed within the sequence activity. In this way, the first part of the loops is handled.

When a condition message is encountered, its messaging is also added as if it was a direct message. Exactly the same procedure is followed as explained in "Handling the Direct Messages in WS-BPEL" section. After that the sequence is closed and the condition is placed within repeatUntil activity. An example condition is given in Figure 5.8.

```
<receive name="receive_iSURF_Send_EC_BTA"
            partnerLink="receiveDataFromManufPL"
            operation="iSURF_Send_EC_BTAfromManufacturer"
            variable="iSURF_Send_EC_BTA_Seller"/>
<assign>
      <copy>
            <from part="message"
                  variable="iSURF_Send_EC_BTA_Seller"/>
            <to part="message"
                  variable="translateDocumentRequest"/>
      </copy>
      <copy>
            <from>'Seller'</from>
            <to part="from"
                  variable="translateDocumentRequest"/>
      </copy>
</assign>
<invoke name="InvokeISUforMapping"
            partnerLink="sendMessageToISUPL"
            operation="translateDocument"
            inputVariable="translateDocumentRequest"
            outputVariable="translateDocumentResponse"/>
<assign>
      <copy>
            <from part="message"
                  variable="translateDocumentResponse"/>
            <to part="message"
                  variable="iSURF_Send_EC_BTA_Buyer"/>
      </copy>
</assign>
<invoke name="InvokeRetailerBPEL"
            partnerLink="sendMessageToRetailerBpelPL"
            operation="iSURF_Send_EC_BTAfromManufBPEL"
            inputVariable="iSURF_Send_EC_BTA_Buyer"/>
```

Figure 5.6: Direct Messaging Example for the Outgoing Messages

```
<repeatUntil>
      <sequence>
      </sequence>
</repeatUntil>
```

Figure 5.7: Repeat Until Activity for Loops

```
<condition>
//TradeItemLocationProfile[@status!='APPROVED']
</condition>
```

Figure 5.8: A sample Repeat Until Condition

Handling of Optional Messaging in WS-BPEL

In fact the Handling of Optional Messaging is very similar to the conditional messages. The only difference appears on the condition. However, this time the message receiving part is

defined in a different way since we may or may not receive the message. A sample definition is given in Figure 5.9 below.

```
<pick>
  <onMessage
      partnerLink="receiveDataFromManufPL"
      operation="iSURF_Send_TM_BTAfromManufacturer"
      variable="iSURF_Send_TM_BTA_Seller">
        <sequence>
              <assign>
                    <copy>
                          <from part="message"
      variable="iSURF_Send_TM_BTA_Seller"/>
                          <to part="message"
      variable="translateDocumentRequest"/>
                    </copy>
                    <copy>
                          <from>'Seller'</from>
                          <to part="from"
      variable="translateDocumentRequest"/>
                    </copy>
              </assign>
              <invoke name="InvokeISUforMapping"
      partnerLink="sendMessageToISUPL"
      operation="translateDocument"
      inputVariable="translateDocumentRequest"
      outputVariable="translateDocumentResponse"/>
              <assign>
                    <copy>
                          <from part="message"
      variable="translateDocumentResponse"/>
                          <to part="message"
      variable="iSURF_Send_TM_BTA_Buyer"/>
                    </copy>
              </assign>
              <invoke name="InvokeRetailerBPEL"
      operation="iSURF_Send_TM_BTAfromManufBPEL"
      inputVariable="iSURF_Send_TM_BTA_Buyer"/>
        </sequence>
  </onMessage>
  <onAlarm>
        <for>'PT4H'</for>
        <assign>
              <copy>
                    <from expression="false()"/>
                    <to
      variable="iSURF_Send_TM_BTA_arrived_Seller"/>
              </copy>
        </assign>
  </onAlarm>
</pick>
```

Figure 5.9: A sample Optional Message expected from Manufacturer WS

The pick activity is composed of one or more onMessage part and one onAlarm part. The example above belongs to the Manufacturer process and a message is expected to be sent from Manufacturer to the Retailer. With the onMessage part, it is defined that a message can be received from the web services of Manufacturer. If received the activities defined within the sequence in the onMessage part are performed. These activities are similar to those performed during a direct message handling. The input is copied to the translationRequest variable, the ISU translation service is called, the return value of the ISU is copied to another variable, and finally the retailer WS-BPEL process is called.

If the message direction was on the other way, the content of the onMessage would be receiving message from the retailer process and sending it to the manufacturer web service.

These are the one way optional messages. However, Exception Notification messages should be able to be sent from both sides to the other side; from Retailer to the Manufacturer and from Manufacturer to the Retailer. In this case, both of the onMessage parts described above would be included in the pick activity.

onAlarm part of the pick activity is executed if there is no message received within the specified duration. The body of the onAlarm part simply assigns a false value to the variable indicating that the message did not arrive. This variable is later used in the conditions. This variable is always assigned true with an assign activity defined just before the pick activity as shown in Figure 5.10. In this way, the value of the variable indicates whether the message has arrived or not during the specified period.

```
<assign>
      <copy>
            <from expression="true()"/>
            <to variable="iSURF_Send_TM_BTA_arrived_Seller"/>
      </copy>
</assign>
```

Figure 5.10: Assigning initial true to the Variable Before Pick Activity

If one of the branches of the optional message constitutes a loop, then a condition is added to the end of the pick activity definition as shown in Figure 5.11.

With all these three cases (direct, loop and optional messaging), the business logic within the CPFR process defined with CPFR Designer Tool is defined in the WS-BPEL Process. At the end, there will be a separate WS-BPEL definition for each of the partners defining

```
<condition>
iSURF_Send_TM_BTA_arrived_Seller
</condition>
```

Figure 5.11: An Optionality Condition

the executable definition of the choreographic CPFR process in ebBP from their point of view.

5.1.2 Generation of PDD Definitions

After the WS-BPEL definition is generated, for being able to deploy it to the ActiveBPEL engine, PDD definitions are generated. It is defined with the <process> element containing partner links and WSDL references (composes of <partnerLinks> and <references> tags). The template of a PDD file is shown in Figure 5.12.

```
<process
xmlns="http://schemas.active-endpoints.com/pdd/2006/08/pdd.xsd"
xmlns:bpelns="http://CPFRManufacturer"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing"
location="bpel/CPFRDemonstration/CPFRManufacturer.bpel"
name="bpelns:CPFRManufacturer">
      <partnerLinks>
            .
            .
      </partnerLinks>
      <references>
            .
            .
      </references>

</process>
```

Figure 5.12: PDD Definition Template

Inside the partnerLinks, a number of partnerLink definitions are given. If this partnerLink is for an operation of WS-BPEL definition accessed from outside, simply the name of the service is provided within a myRole element as shown in Figure 5.13.

If the partnerLink defined is a link for accessing to an external service, then a more detailed information is provided including its namespace, service address, service name and the port type. An example definition is provided in Figure 5.14 below.

```
<partnerLink name="receiveFromManufPL">
      <myRole allowedRoles="" binding="MSG"
            service="receiveFromManufPLService"/>
</partnerLink>
```

Figure 5.13: A partnerLink Definition for the WS-BPEL to be Called from Outside

```
<partnerLink name="sendMessageToISUPL">
      <partnerRole endpointReference="static">
            <wsa:EndpointReference
                  xmlns:s="http://www.srdc.com.tr/isu/"
                  xmlns:wsa="http://schemas.xmlsoap.org/ws/2003
            /03/addressing">

                  <wsa:Address>
http://144.122.230.57:8080/axis/services/ISUTranslationService
                  </wsa:Address>
                  <wsa:ServiceName
                  PortName="ISUTranslationService">
                              s:ISUTranslationService
                  </wsa:ServiceName>
            </wsa:EndpointReference>
      </partnerRole>
</partnerLink>
```

Figure 5.14: A partnerLink Definition for External Service Provider

After all partnerLinks defined, the referenced WSDL are given under the references element. For this, simply the location of the web service along with its namespace is provided with a wsdl element as shown in Figure 5.15.

```
<wsdl
location="http://144.122.230.57:8080/axis/services/ISUTranslati
onService?wsdl" namespace="http://www.srdc.com.tr/isu/"/>
```

Figure 5.15: A WSDL Reference Example

5.1.3 Generation of WSDL Definitions for WS-BPEL Processes

Since the deployed WS-BPEL process will be also exposed as a web service for being accessible by web services of the company or by the other WS-BPEL processes, a WSDL file is generated describing the operations WS-BPEL process owns. Additionally, web services

do not have to include partnerLinkTypes and usually they do not include. In order to be able to use the web services wrapping the enterprise applications of the companies directly, partnerLinkTypes of those services are defined again in the WSDL file. A sample definition is given in Figure 5.16.

```
<wsdl:definitions €
      targetNamespace="http://www.srdc.com.tr/manufacturer/bpel
"
      xmlns:manubpel="http://www.srdc.com.tr/manufacturer/bpel"
      xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
      xmlns:apachesoap="http://xml.apache.org/xml-soap"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:plnk2="http://docs.oasis-
open.org/wsbpel/2.0/plnktype"
      xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
      xmlns:manufacturer="http://www.srdc.com.tr/manufacturer/"
      xmlns:isu="http://www.srdc.com.tr/isu/"
      xmlns:retbpel="http://www.srdc.com.tr/retailer/bpel">

      <wsdl:types>
            .
            .
            .
      </wsdl:types>

      <wsdl:message name="HistShipmentMessage">
            <wsdl:part name="inputMessage"
            element="manubpel:historicalShipmentMessage"/>
                  .
                  .
                  .
      </wsdl:message>

      <wsdl:portType name="retrieveDataFromManufPortType">
            <wsdl:operation
                  name="retrievePAFromManufacturerOperation">
                  <wsdl:input name="inputMessage"
                  message="manubpel:HistShipmentMessage"/>
            </wsdl:operation>
                  .
                  .
                  .
      </wsdl:portType>
</wsdl:definitions>
```

Figure 5.16: An Example of WSDL for WS-BPEL Process

Basically, it is very similar to the usual WSDL definitions. Message types can be defined at the top. Then the messages are defined. Differently from a usual WSDL definition there is no service and binding definitions. The list of operations is given in the portType element.

```
<!-- send to isu services pl type -->
<plnk2:partnerLinkType
xmlns:plnk2="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
      name="sendISUMessagePLType">

      <plnk2:role name="sendMessageToISURole"
                  portType="isu:ISUServicePortType"/>

</plnk2:partnerLinkType>
```

Figure 5.17: PartnerLinkType Definition Example

Finally, partnerLinkTypes are defined just inside of the <wsdl:definitions> tag. These partnerLinkTypes uses the portTypes defined both inside this WSDL document, and also inside the WSDL document of the Web Services wrapping the systems of companies. How a partnerLinkType can be defined is shown in Figure 5.17.

With all these definitions (.bpel, .pdd and .wsdl), the deployment package is ready to be deployed on the WS-BPEL engine.

5.2 Execution of the WS-BPEL Processes

5.2.1 ActiveBPEL Community Edition Engine

The WS-BPEL definitions generated by CPFR Designer Tool are executed on an open source WS-BPEL engine which is called ActiveBPEL [3]. ActiveBPEL provides a robust runtime environment capable of executing process definitions created for the BPEL standard. It is completely based on the WS-BPEL standard. Additionally, the engine is expected to continually benefit from the contributions of both the Active Endpoints and the ActiveBPEL community at large. The engine provides a visual representation of the BPEL process deployed as shown in Figure 5.18.

5.2.2 An Open Source Enterprise Application - vtiger CRM

To be able to demonstrate the results of the thesis work, an enterprise application is needed through which the planning will be performed. For this we have chosen vtiger CRM [42]. vtiger CRM is an enterprise-ready Open Source CRM software mainly for small and medium businesses. vtiger CRM is built over proven, fast, and reliable LAMP/WAMP (Linux/Windows, Apache, MySQL, and PHP) technologies and other open source projects.

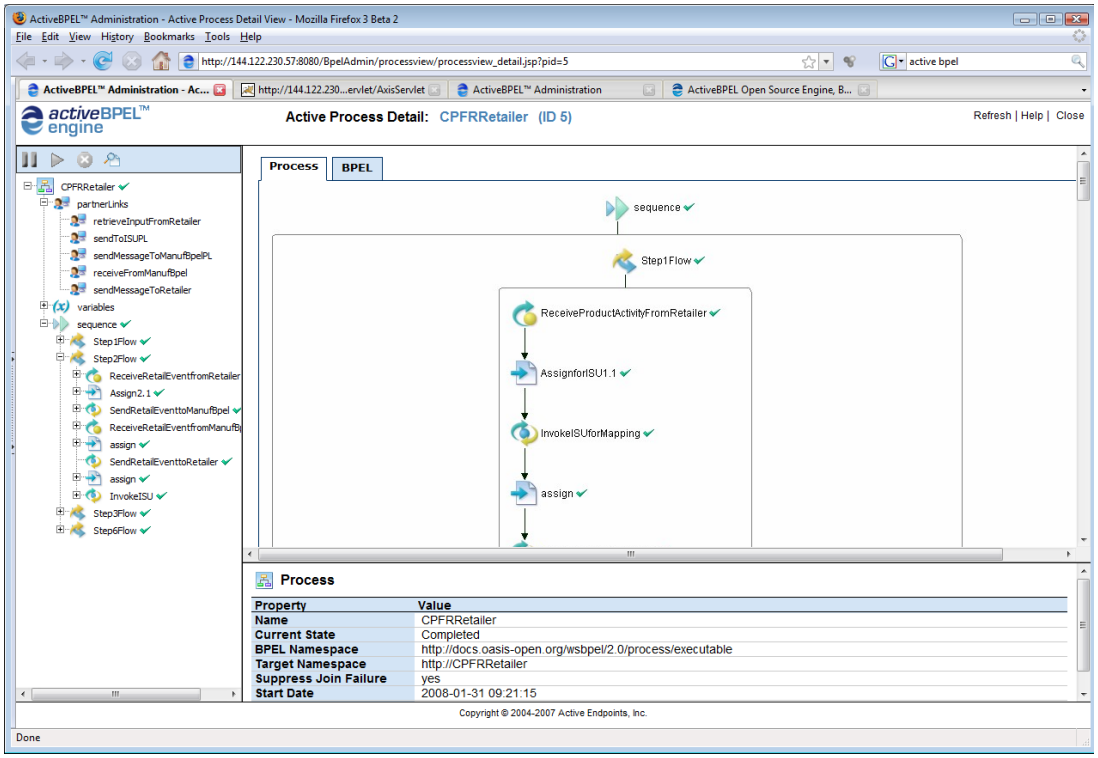

Figure 5.18: ActiveBPEL Community Edition Engine

vtiger CRM leverages the benefits of the Open Source software and adds more value to the end-users by providing many enterprise features, such as Sales force Automation, customer support & service, marketing automation, inventory management, multiple database support, security management, product customization, calendaring, E-mail integration, add-ons, and others. Within the scope of the iSURF project vtiger CRM application has been extended so that it can display the content of the messages received from the web services and send messages to the other Web Services. For this new menus have been added like shown in Figure 5.19.

5.2.3 Web Services Exposing the Functionalities of Enterprise Application

As mentioned in Section 5.1.1, the functionalities of the Enterprise Applications of the companies who will join to the collaboration are exposed as the web services. WS-BPEL processes communicate with the Enterprise Applications through the standard interfaces of the web services. In order to demonstrate the execution of an example CPFR Process, two web services have been implemented within this work: ManufacturerService and RetailerService.

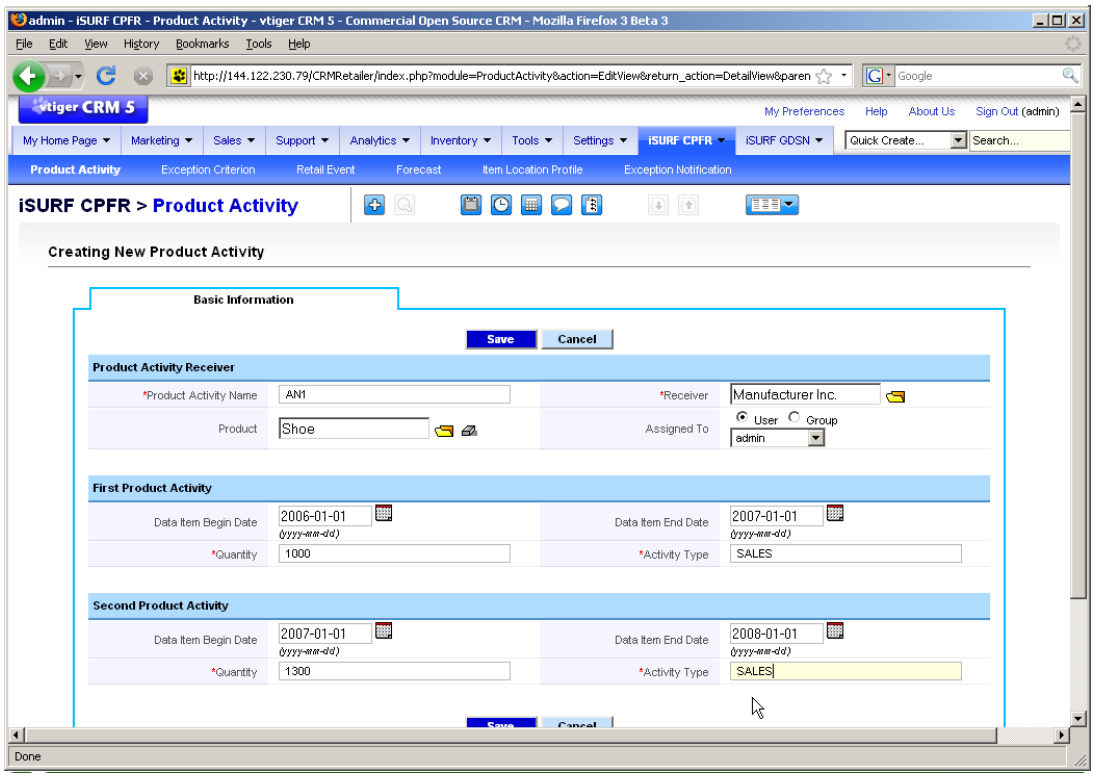

Figure 5.19: vtiger CRM Application

Both of these services have the following operations:

- receiveTerminationMessage
- receiveExceptionCriteriaMessage
- receiveExceptionNotificationMessage
- receiveSalesForecastMessage
- receiveSalesForecastRevisionMessage
- receiveOrderForecastMessage
- receiveOrderForecastRevisionMessage
- receiveTILPMessage
- receiveRetailEventMessage
- receiveInventoryStatusMessage
- receivePerformanceHistoryMessage

- receiveProductActiviyMessage
- receivePurchaseConditionsMessage
- receiveInventoryStatusMessage

All of the operations are one way. Except receiveTerminationMessage operation which takes string as input, the other operations take GS1 Messages as inputs. These web services provide communication with the vtiger CRM application of the manufacturer and the retailer.

5.2.4 iSURF Monitoring GUI

Again within the scope of the iSURF project, a monitoring GUI has been implemented. This application simply displays the messages exchanged during the execution. The messages are listed and when the document icon is clicked on one of the sides, the content of the document sent or received is displayed on the right side of the screen. A snapshot from the application is given in Figure 5.20.

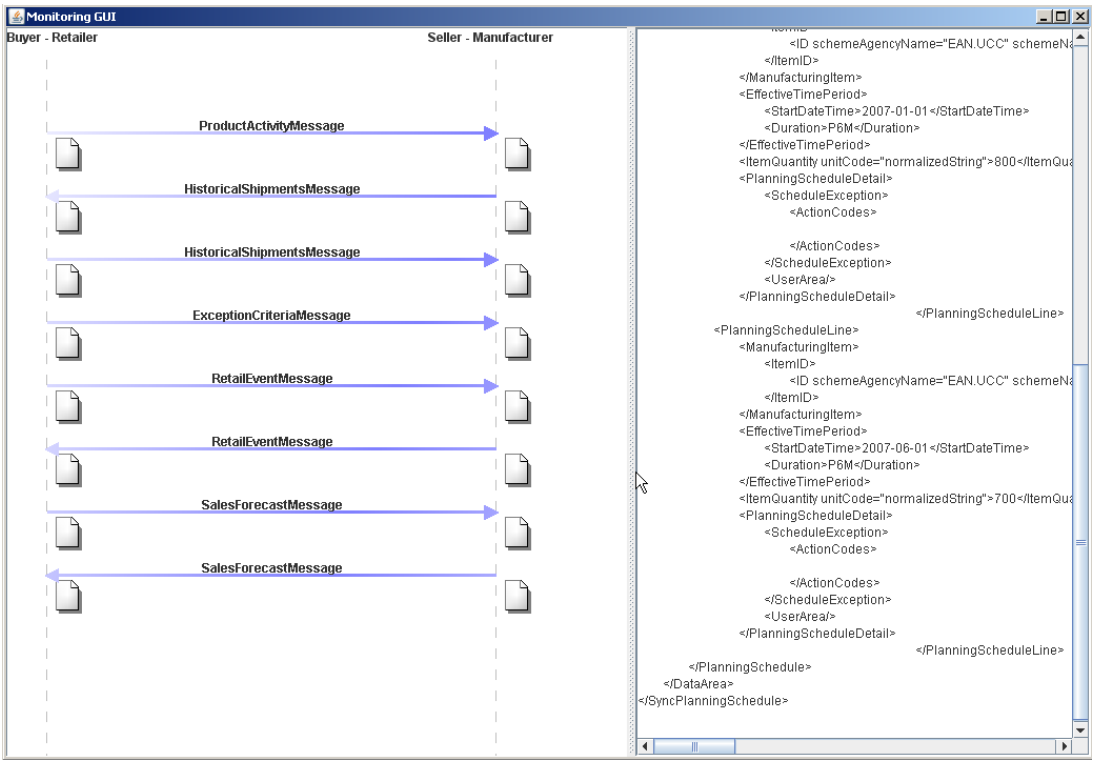

Figure 5.20: iSURF Monitoring GUI

5.2.5 General View of the Execution Environment

With all the components mentioned above a setup for demonstrating the execution of the automatically generated WS-BPEL process has been developed. The overall view of the setup environment is given in Figure 5.21. At the two sides, there are two CRM applications which need to communicate for the plan and the forecast generation. Moreover, this communication will bring much more benefits if it is done conforming to the CPFR guidelines.

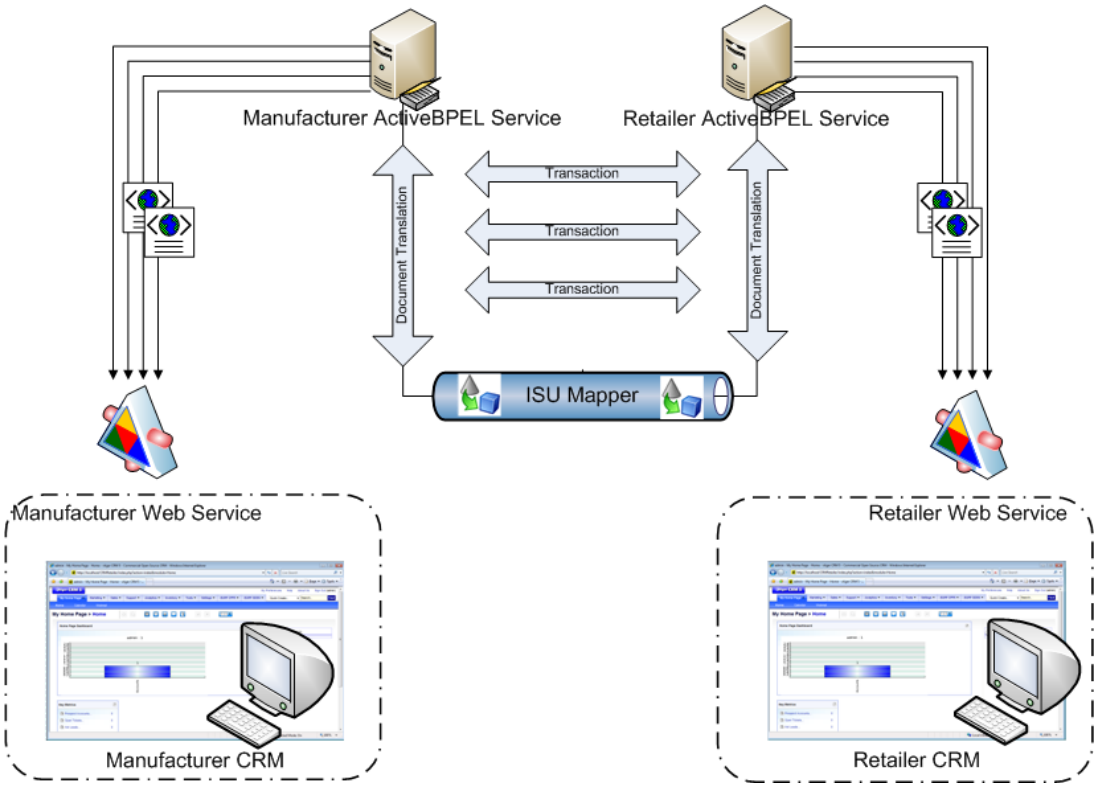

Figure 5.21: Setup of the Execution Environment

Using the CPFR Designer Tool the parties that want to collaborate, design a CPFR Process using one of the three templates given. The ebBP and WS-BPEL definitions are generated easily with the tool.

Returning back to the above figure, the legacy applications of the companies are wrapped with the Web Services. WS-BPEL processes generated use these web services for communicating with the applications. For instance, a message that will be sent from the Manufacturer to the Retailer follows the path:

- Manufacturer CRM ⇒ Manufacturer Web Service

- Manufacturer Web Service ⇒ Manufacturer WS-BPEL Process Service
- Manufacturer WS-BPEL Process Service ⇒ ISU Service (translation)
- ISU Service ⇒ Manufacturer WS-BPEL Process Service
- Manufacturer WS-BPEL Process Service ⇒ Retailer WS-BPEL Process Service
- Retailer WS-BPEL Process Service ⇒ Retailer Web Service
- Retailer Web Service ⇒ Retailer CRM

All the translations and business logic are handled in the background without needing the user to know.

CHAPTER 6

RELATED WORK

Collaborative Planning, Forecasting and Replenishment (CPFR®), the cross-industry initiative has started to be taken up in 2001 especially in North America by early adapters through several pilot projects.

Among many, a successful collaborative planning initiative is from a European chipmaker, STMicroelectronics [38]. Previously like many other companies, STMicroelectronics and its trading partners were using manual processes to match trading partner demand with ST's manufacturing capacity. Realizing that the long cycle time made ST and its trading partners very inflexible and unresponsive to changes in market demands, ST initiated the eChO pilot with its trading partners to implement a fully automated B2B-based collaborative planning and procurement solution based on RosettaNet standards. As a result ST experienced increased process efficiency, a decrease in planning man-hours by at least one full-time equivalent, reduced variability associated with infrequent data exchanges and long planning lead-times.

As reported in the "European CPFR Insights" prepared by Accenture [22] there are also pilot applications for Collaborative planning by the following European Companies: Condis, Delhaize, Elgeka, Eqos, Heinz, Hellas Spar Veropoulos, Henkel, JDA, Johnson &Johnson, Ketjuetu, Kimberley-Clark, Kraft, Marks & Spencer, Masterfoods, Nestlé, ONIA-NET, Procter & Gamble, SCA, Superdrug, Syncra.

Taking into account the responses from the pilot projects, pioneering companies are achieving some improvements in their internal workflows through the adoption of Collaborative Planning mechanisms. This thesis work is expected to give a momentum to the pilot projects as for defining customized CPFR processes, it provides an easy to use add-on tool enabling the use of the existing systems of the companies. The tool reduces the effort on building the customized CPFR Processes and provides an automatic execution platform

for the implementation of the process. However, from the technology point of view, there is not many ready to use systems and tools in the market which are CPFR compliant for collaboration in the Supply Chain.

One of the tools that are used in pilot applications is i2 Collaborative Replenishment developed by i2 company [28]. i2 Collaborative Replenishment can enable companies to manage various aspects of their replenishment programs including collaborative forecast adjustments, customer demand changes, and program performance management. i2 Collaborative Replenishment is designed to offer forecast analysis, promotions collaboration, demand change liability/flexibility management, replenishment, and monitoring. The solution includes setup, planning and execution, and performance measurement components.

Another tool helping the supply chain companies during the implementation of CPFR is Vendor Managed Replenishment Solution developed by JDA [30]. From communications, negotiations and other key elements of CPFR, VMR enables vendors to execute forecasting and replenishment based on a retailer's point-of-sale data, while building optimum orders for distribution centers (DCs) and stores. Vendor Managed Replenishment presents the information needed by retail analysts so that they can take advantage of the best offers, manage exceptions, monitor promotions and perform additional buying and merchandising services for their customers.

The last example solution is the Predictive Trade Planning and Optimization tool from Oracle Demantra [34]. The tool basically provides consumer goods organizations with robust trade promotion and account planning, sales forecasting, and promotion optimization capabilities. It has the support for a Collaborative planning platform to enable that sales forecasts and event plans are synchronized across the marketing, the finance, the manufacturing, and the supply chain.

There are a limited number of tools for helping the companies regarding the CPFR and almost in all of these tools the functionalities for the CPFR are provided as a part of a supply chain management system. Therefore, these solutions do not provide the companies with the ability to design a joint CPFR Processes to be implemented with the other companies. Additionally, that the CPFR support being a part of the supply chain solution requires the supply chain companies to purchase those solutions and more importantly to abandon their existing solutions. Furthermore, existing solutions do not address the need of SMEs, and are very complicated. CPFR Designer Tool developed in this thesis, provides an add-on solution upon the existing systems on the level of services enabling the companies to continue with their existing systems. Additionally, for being able to use the tool, only knowledge on CPFR

guidelines are enough since the tool handles all the technical work (generation of ebBP definitions and WS-BPEL deployment package) in the background seamless to the user.

Regarding the ebBP specification, there are uses of it from different areas. Recently, Japan Electronics and Information Technology Industries Association (JEITA) [31] has developed a collaborative ebBP process definitions for the Japanese community. Additionally, an Italian knit wear has produced draft process definitions to show how the condition expressions can drive different usages of specific activities under different the conditions [32]. Criminal justice in the Netherlands has created an exemplary ebBP process definition [21].

WS-BPEL standard, on the other hand, has a strong industry support [7]. For example, IBM uses WS-BPEL in the core, for delivering Business Process Management (BPM) enabled by SOA. Additionally, SAP considers the process definition capabilities of WS-BPEL as one of the key building blocks for enterprise SOA and they plan to enhance the existing SAP NetWeaver support of BPEL4WS 1.1 with a WS-BPEL 2.0. Other companies following closely WS-BPEL standard are Active Endpoints, Adobe, BEA Systems, HP, Microsoft, Oracle, Rogue Wave Software, Sun Microsystems, TIBCO and webMethods.

This thesis work makes use of the WS-BPEL, a strong e-business industry standard in SOA. The WS-BPEL standard used on the CPFR Process execution significantly enhances the ability to offer service-based integration processes to CPFR Execution Platform users proving a service level add-on over their existing systems.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

Currently, there are many software applications handling planning, scheduling, material management, invoicing, workflow management within an organization. These applications include Warehouse Management Systems (WMS), Forecasting/Advanced Planning Systems (APS), Enterprise Resource Planning Systems (ERP), Material Requirements Planning (MRP), Distribution Requirements Planning (DRP) and Customer Relationship Management (CRM) which handle organization's day-to-day planning activities. However, these systems manage only the "internal supply chain".

On the other hand, companies need to plan across a wider span of activities and need to collaborate with their partners to optimize the "overall" profitability. This requires the collaborative planning within a supply chain and the exchange of planning data. Up to now there have been a number of efforts to provide a common medium for the collaborative planning and for the exchange of required planning data among the partners in the supply chain.

Collaborative Planning, Forecast and Replenishment (CPFR®) [9] is one of the most prominent initiatives on the Collaborative Planning which has been initiated by VICS (Voluntary Inter-industry Commerce Standard) on this need. The CPFR is a reference model providing a general framework for the collaborative aspects of planning, forecasting and replenishment processes. This reference model is a cycle, including the following collaboration tasks: Strategy & Planning, Demand & Supply Management, Execution and Analysis. The CPFR guidelines specify an iterative exchange of information between the parties in the supply chain.

The evolution of the CPFR into specific scenarios for the collaborative planning, the collaborative forecasting and the collaborative replenishment is not a cause for concern, but rather a necessary step in reengineering enterprise processes to take advantage of collabora-

tion. CPFR has always been a reference model, with many alternatives. For this reason, a customization utility is needed for companies to easily generate a CPFR process appropriate to their needs. CPFR is technology independent: it does not provide any guidelines on how the internal planning processes mostly implemented by the legacy enterprise planning systems will be integrated with the joint CPFR collaboration process. As an expected result of this, no messaging standards are mandated by the CPFR guidelines. All these issues bring about technological barriers to the companies which are intended to implement the CPFR guidelines. For implementing a CPFR process and for providing the integration with their underlying systems, the companies need to allocate significant resources for this. While this is difficult for event big companies, SMEs have great challenges hampering the adoption of the CPFR.

To address these technological challenges on the adoption of the CPFR, in this thesis work we have developed a Service Oriented Collaborative Supply Chain Planning Process Definition and Execution Platform. First of all, building blocks of the CPFR process have been defined in ebBP language which is a choreographic business process definition language. A CPFR Designer Tool have been developed which enables the companies visually define and customize the CPFR Processes generating the machine processable business process definitions in the background. Finally, the business process definition in ebBP has been converted to the WS-BPEL which is an executable business process definition for the orchestration and a CPFR Process Execution Environment has been provided enabling the execution of the CPFR Process generated in the WS-BPEL. The benefits of the developed Service Oriented Collaborative Supply Chain Planning Process Definition and Execution Environment Platform can be listed as:

- *An Open Mechanism for the definition and execution of a collaborative planning process:* As presented above, CPFR only provides guidelines, however does not mandate any technology for the definition and execution of planning process. For this reason companies have difficulties to define and deploy the CPFR solutions. The Service Oriented Open Platform provided for the definition and execution of the collaborative planning processes involves as many supply chain tiers as necessary.

- *Integration with the existing internal planning processes:* The generated CPFR processes are easily integrated with the existing internal planning processes executed by the enterprise legacy applications. The legacy systems do not need painstaking changes in order to conform to the inter-enterprise collaborative planning processes. It is suf-

ficient to only expose the functionalities of the existing legacy applications as the web services in order to provide a seamless, platform independent interoperation between the enterprise applications of multiple domains.

- *Easy Customization:* The CPFR Designer Tool allows trading partners in the supply chain to easily customize their planning business processes and planning documents according to their scenario specific needs. Thus the partners are not bound to predefined standards which may not respond to all of the needs of the collaboration.

As future work, the CPFR Designer Tool and the CPFR Process Execution Environment will be integrated to the iSURF project as the iSURF Service Oriented Collaborative Supply Chain Planning Process Definition and Execution Platform. The integration with the iSURF ISU has already been provided with the inclusion of the ISU Translation service within the generated CPFR WS-BPEL processes.

As a future implementation, whether it is necessary and practical to launch a new CPFR process from already executing CPFR Process will be discussed. If the analysis results in that it is practical, both the CPFR Designer Tool and CPFR Process Execution Environment will be extended to include this property. This property will be needed when there is an unrecoverable exception during the CPFR collaboration (e.g. seller for some reason cannot provide any product to the buyer and the buyer needs to find another seller automatically) and a new partner is needed to be found.

Another possible improvement on the CPFR Designer Tool might be during the generation of the WS-BPEL process. Instead of requesting an operation for each of the Message Exchange State in the CPFR Process, through semantic explanations of the services, this assignment might be performed automatically. In this way, for the user it is only required to select the web services to be used for the WS-BPEL process.

Another future work is that the integration of the CPFR Process Execution Environment and the automatically generated web services with the iSURF Legacy Web Service Wrapper Component will be tested. Currently, the manually developed web services are used since the iSURF Legacy Web Service Wrapper Component has not been implemented yet.

Furthermore, at the end, the communication between the exposed services of the legacy applications are planned to be realized within an Enterprise Service Bus (ESB). When the iSURF ESB is implemented, the compatibility of the currently generated WS-BPEL deployment package and the iSURF ESB will be inspected and the updates will be performed if necessary.

Finally, a pilot application [13] will be deployed in the premises Fratelli PIACENZA, a manufacturer of fine woolen fabrics and supplier to many world-leading apparel brand manufacturers, including Boss and INCO/Zegna. This pilot application will assess how much enhancement is provided to the as-is business model of PIACENZA's textile supply chain process through the iSURF components enabling the collaborative planning.

REFERENCES

[1] The Next Wave of Supply Chain Advantage: Collaborative Planning, Forecasting and Replenishment. Technical report, Industry Directions Inc. and Syncra Systems, April 2000. http://www.industrydirections.com/pdf/CPFRPublicReport.pdf, last visited on September 2008.

[2] Collaborative Planning, Forecasting and Replenishment Version 2.0. Global commerce initiative recommended guidelines, Voluntary Interindustry Commerce Standards (VICS) Association, June 2002. http://www.vics.org/docs/committees/cpfr-/CPFR_Tabs_061802.pdf, last visited on September 2008.

[3] The ActiveBPEL Community Edition Engine. http://www.activevos.com/community-open-source.php, last visited on September 2008.

[4] ActiveBPEL Community Edition Engine Download Package. http://64.119.177.125-/download/files/5.0/final/activebpel-5.0.2-bin.zip, last visited on September 2008.

[5] The Apache Software Foundation. http://www.apache.org/, last visited on September 2008.

[6] A Close Look at BPEL 2.0. http://www.sys-con.com/node/434430, last visited on September 2008.

[7] Support for WS-BPEL OASIS Standards. http://www.oasis-open.org/news/oasis-news-2007-04-12.php, last visited on September 2008.

[8] Common Object Requesting Broker Architecture (CORBA). http://en.wikipedia.org-/wiki/CORBA, last visited on September 2008.

[9] Collaborative Planning Forecasting and Replenishment Guidelines. http://-www.vics.org/committees/cpfr/, last visited on September 2008.

[10] CPFR Baseline Study - Manufacturer Profile, 2002. http://www.gmabrands.com-/industryaffairs/docs/cpfr.pdf, last visited on September 2008.

[11] VICS CPFR XML Messaging Model, 2001. http://xml.coverpages.org/CPFR-XMLMessagingModel0601a.pdf, last visited on September 2008.

[12] M. de Paula, J. Oliveira, J.M. de Souza, and J. Strauch. Improving design with collaborative planning, forecasting and replenishment through knowledge management and cscw. *Computer Supported Cooperative Work in Design, 2004. Proceedings. The 8th International Conference on*, 2:534–539 Vol.2, May 2004.

[13] A. Dogac, G. Laleci, A. Okcan, M. Olduz, M. Sesena, and A. Canepa. iSURF -An Interoperability Service Utility for Collaborative Supply Chain Planning across Multiple Domains: Textile Supply Chain Pilot. In *eChallenges Conference*, October 2008.

[14] A. Dogac, G. B. Laleci, M. Olduz, Y. Kabak, A. Okcan, and Tasyurt I. An Interoperability Service Utility for Collaborative Supply Chain Planning. In *International Conference on Concurrent Enterprising (ICE) Conference*, June 2008.

[15] European Article Number. http://en.wikipedia.org/wiki/European_Article_Number, last visited on September 2008.

[16] European Article Number Communication. http://www.gs1.org/productssolutions-/ecom/eancom/, last visited on September 2008.

[17] ebXML Business Process Technical Committee. ebXML Business Process Specification Schema Technical Specification v2.0.4. Specification, OASIS, December 2006. http://docs.oasis-open.org/ebxml-bp/2.0.4/ebxmlbp-v2.0.4-Spec-os-en.pdf, last visited on September 2008.

[18] ebXML Business Process Technical Committee. Web Services Business Process Execution Language Version 2.0. Standard, OASIS, April 2007. http://docs.oasis-open.org-/wsbpel/2.0/wsbpel-v2.0.pdf, last visited on September 2008.

[19] OASIS ebXML Business Process TC. http://www.oasis-open.org/committees/ebxml-bp/, last visited on September 2008.

[20] Electronic Product Code Standardization (EPCGlobal). http://www.epcglobalinc.org-/home, last visited on September 2008.

[21] The ePV example built by Dutch criminal justice system. http://www.oasis-open.org-/committees/download.php/16436/epv-Example2.zip, last visited on September 2008.

[22] European CPFR Insights. http://www.ecrnet.org/04-publications/blue_books-/pub_2002_cpfr_european_insights.pdf, last visited on September 2008.

[23] Adobe Flex 3. http://www.adobe.com/products/flex/, last visited on September 2008.

[24] European Commission ICT FP7. http://cordis.europa.eu/fp7/ict/, last visited on September 2008.

[25] Global Data Synchronization Network. http://www.gs1.org/productssolutions/gdsn/, last visited on September 2008.

[26] Global Standard One (GS1). http://www.gs1.org/, last visited on September 2008.

[27] Global Standard One XML. http://www.gs1.org/productssolutions/ecom/xml/, last visited on September 2008.

[28] i2 Collaborative Replenishment Solution. http://www.i2.com/Solution_library-/c_c_collaborative_replenishment.cfm, last visited on September 2008.

[29] iSURF Project. http://www.isurfproject.eu/, last visited on September 2008.

[30] JDA Vendor Replenishment Solution. http://www.jda.com/solutions/vendor-managed-replenishment.html, last visited on September 2008.

[31] Japan Electronics and Information Technology Industries Association (JEITA). http://-www.jeita.or.jp/english/, last visited on September 2008.

[32] Italian Knit Wear Draft Process Definitions. http://www.oasis-open.org/committees-/download.php/18365/CristianoNovelli_ebBP_May06.zip, last visited on September 2008.

[33] Dale Moberg and Monica J. Martin. The ebBP (ebXML Business Process Specification Schema), April 2006. http://www.oasis-open.org/committees/download.php-/17857/ebxmlbp-v2.0.3-WhitePaper-wd-r01-en.pdf, last visited on September 2008.

[34] Demantra Predictive Trade Planning and Optimization by Oracle. http://-www.oracle.com/applications/scm/demantra/predictive-trade-planning.html, last visited on September 2008.

[35] Intel and Shinko use RosettaNet standards to build forecast-to-cash procurement process, 2003. http://www.rosettanet.org, last visited on September 2008.

[36] Simple Object Access Protocol (SOAP). http://www.w3.org/TR/soap/, last visited on September 2008.

[37] Secure Sockets Layer (SSL). http://en.wikipedia.org/wiki/Secure_Sockets_Layer, last visited on September 2008.

[38] STMicroelectronics E-Chain Optimization Project: Achieving Streamlined Operations Through Collaborative Forecasting and Inventory Management. http://harvardbusinessonline.hbsp.harvard.edu-/relay.jhtml?name=itemdetail&id=GS36&referral=9026, last visited on September 2008.

[39] the VICS CPFR Advisory Team. CPFR: An Overview. Technical report, Voluntary Interindustry Commerce Standards (VICS) Association, May 2004. http://-committees.vics.org/committees/cpfr/CPFR_Overview_US-A4.pdf, last visited on September 2008.

[40] Uniform Code Council. http://www.uc-council.org/, last visited on September 2008.

[41] Voluntary Interindustry Commerce Standards. http://www.vics.org/, last visited on September 2008.

[42] Vtiger CRM. http://www.vtiger.com/, last visited on September 2008.

[43] The World Wide Web Consortium (W3C). http://www.w3.org/, last visited on September 2008.

[44] W3C Membership. http://www.w3.org/Consortium/membership, last visited on September 2008.

[45] WSBPEL Version 2.0 Primer, May 2007. http://docs.oasis-open.org/wsbpel/2.0-/Primer/wsbpel-v2.0-Primer.pdf, last visited on September 2008.

[46] Web Service Description Language (WSDL). http://www.w3.org/TR/wsdl, last visited on September 2008.

[47] Extensible Markup Language (XML). http://www.w3.org/XML/, last visited on September 2008.

[48] Extensible Markup Language (XML) 1.0. http://www.w3.org/TR/xslt, last visited on September 2008.

[49] The XML Path Language (XPath). http://www.w3.org/TR/xpath, last visited on September 2008.

[50] XML Schema (XSD). http://www.w3.org/XML/Schema, last visited on September 2008.

[51] The Extensible Stylesheet Language Family (XSL). http://www.w3.org/Style/XSL/, last visited on September 2008.

[52] XSL Transformations (XSLT). http://www.w3.org/TR/xslt, last visited on September 2008.

APPENDIX A

AN EXAMPLE EBBP DEFINITION GENERATED BY CPFR DESIGNER TOOL

```
<?xml version="1.0" encoding="utf-8"?>
<ProcessSpecification nameID="iSURF_BusinessCollaborations"
        uuid="urn:metu_srdc:names:specification:isurf_businesscollaborations:schema"
        name="iSURF Business Collaborations" xmlns:xi="http://www.w3.org/2001/XInclude"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema_instance">
  <BusinessDocument nameID="Retail_Event_Document" name="Retail Event">
    <Specification type="schema" nameID="Retail_Event_Specification" location="Performance_History"
           name="Retail Event Specification"/>
  </BusinessDocument>
  <BusinessDocument nameID="Trade_Item_Location_Profile_Document" name="Trade Item Location Profile">
    <Specification type="schema" nameID="Trade_Item_Location_Profile_Specification"
           location="Trade_Item_Location_Profile" name="Trade Item Location Profile Specification"/>
  </BusinessDocument>
  <BusinessDocument nameID="Termination_Message_Document" name="Termination Message">
    <Specification type="schema" nameID="Termination_Message_Specification"
          location="Termination_Message" name="Termination Message Specification"/>
  </BusinessDocument>
  <Notification nameID="iSURF_Exception_Criteria_Transaction" name="Exception Criteria Transaction"
          isGuaranteedDeliveryRequired="true">
    <RequestingRole nameID="iSURF_ECT_Initiator" name="Initiator"/>
    <RespondingRole nameID="iSURF_ECT_Responder" name="Responder"/>
    <RequestingBusinessActivity nameID="iSURF_ECT_ResBA"
          name="Exception Criteria Responding Business Activity" isAuthorizationRequired="true"
          isNonRepudiationRequired="true">
      <Documentation>This is the messaging for Exception Criteria exchange.</Documentation>
      <DocumentEnvelope nameID="iSURF_ECT_DocEnv" businessDocumentRef="Exception_Criteria_Document"
        name="Exception Criteria Document" isAuthenticated="persistent" isConfidential="persistent"/>
      <ReceiptAcknowledgement name="ra2" nameID="iSURF_ECT_RBA_RA" signalDefinitionRef="ra2"/>
      <ReceiptAcknowledgementException name="rae2" nameID="iSURF_ECT_RBA_RAE" signalDefinitionRef="rae2"/>
```

```
    <AcceptanceAcknowledgement name="aa2" nameID="iSURF_ECT_RBA_AA" signalDefinitionRef="aa2"/>
    <AcceptanceAcknowledgementException name="aae2" nameID="iSURF_ECT_RBA_AAE"
          signalDefinitionRef="aae2"/>
  </RequestingBusinessActivity>
  <RespondingBusinessActivity nameID="iSURF_ECT_ResBA"
          name="Exception Criteria Responding Business Activity"/>
</Notification>
<Notification nameID="iSURF_Retail_Event_Transaction" name="Retail Event Transaction"
          isGuaranteedDeliveryRequired="true">
  <RequestingRole nameID="iSURF_RET_Initiator" name="Initiator"/>
  <RespondingRole nameID="iSURF_RET_Responder" name="Responder"/>
  <RequestingBusinessActivity nameID="iSURF_RET_ResBA" name="Retail Event Responding Business Activity"
          isAuthorizationRequired="true" isNonRepudiationRequired="true">
    <Documentation>This is the messaging for Retail Event exchange.</Documentation>
    <DocumentEnvelope nameID="iSURF_RET_DocEnv" businessDocumentRef="Retail_Event_Document"
          name="Retail Event Document" isAuthenticated="persistent" isConfidential="persistent"/>
    <ReceiptAcknowledgement name="ra2" nameID="iSURF_RET_RBA_RA" signalDefinitionRef="ra2"/>
    <ReceiptAcknowledgementException name="rae2" nameID="iSURF_RET_RBA_RAE" signalDefinitionRef="rae2"/>
    <AcceptanceAcknowledgement name="aa2" nameID="iSURF_RET_RBA_AA" signalDefinitionRef="aa2"/>
    <AcceptanceAcknowledgementException name="aae2" nameID="iSURF_RET_RBA_AAE"
          signalDefinitionRef="aae2"/>
  </RequestingBusinessActivity>
  <RespondingBusinessActivity nameID="iSURF_ECT_ResBA"
          name="Exception Criteria Responding Business Activity"/>
</Notification>
<Notification nameID="iSURF_Trade_Item_Location_Profile_Transaction"
          name="Trade Item Location Profile Transaction" isGuaranteedDeliveryRequired="true">
  <RequestingRole nameID="iSURF_TILPT_Initiator" name="Initiator"/>
  <RespondingRole nameID="iSURF_TILPT_Responder" name="Responder"/>
  <RequestingBusinessActivity nameID="iSURF_TILPT_ResBA"
          name="Trade Item Location Profile Responding Business Activity"
          isAuthorizationRequired="true" isNonRepudiationRequired="true">
    <Documentation>This is the messaging for Trade Item Location Profile exchange.</Documentation>
    <DocumentEnvelope nameID="iSURF_TILPT_DocEnv"
          businessDocumentRef="Trade_Item_Location_Profile_Document"
          name="Trade Item Location Profile Document" isAuthenticated="persistent"
          isConfidential="persistent"/>
    <ReceiptAcknowledgement name="ra2" nameID="iSURF_TILPT_RBA_RA" signalDefinitionRef="ra2"/>
    <ReceiptAcknowledgementException name="rae2" nameID="iSURF_TILPT_RBA_RAE"
          signalDefinitionRef="rae2"/>
    <AcceptanceAcknowledgement name="aa2" nameID="iSURF_TILPT_RBA_AA" signalDefinitionRef="aa2"/>
    <AcceptanceAcknowledgementException name="aae2" nameID="iSURF_TILPT_RBA_AAE"
          signalDefinitionRef="aae2"/>
  </RequestingBusinessActivity>
  <RespondingBusinessActivity nameID="iSURF_ECT_ResBA"
          name="Exception Criteria Responding Business Activity"/>
</Notification>
<Notification nameID="iSURF_Termination_Message_Transaction"
```

```
        name="Termination Message Transaction" isGuaranteedDeliveryRequired="true">
  <RequestingRole nameID="iSURF_TMT_Initiator" name="Initiator"/>
  <RespondingRole nameID="iSURF_TMT_Responder" name="Responder"/>
  <RequestingBusinessActivity nameID="iSURF_TMT_ResBA"
        name="Termination Message Responding Business Activity"
        isAuthorizationRequired="true" isNonRepudiationRequired="true">
    <Documentation>This is the messaging for Termination Message exchange.</Documentation>
    <DocumentEnvelope nameID="iSURF_TMT_DocEnv" businessDocumentRef="Termination_Message_Document"
        name="Termination Message Document" isAuthenticated="persistent" isConfidential="persistent"/>
    <ReceiptAcknowledgement name="ra2" nameID="iSURF_TMT_RBA_RA" signalDefinitionRef="ra2"/>
    <ReceiptAcknowledgementException name="rae2" nameID="iSURF_TMT_RBA_RAE" signalDefinitionRef="rae2"/>
    <AcceptanceAcknowledgement name="aa2" nameID="iSURF_TMT_RBA_AA" signalDefinitionRef="aa2"/>
    <AcceptanceAcknowledgementException name="aae2" nameID="iSURF_TMT_RBA_AAE"
        signalDefinitionRef="aae2"/>
  </RequestingBusinessActivity>
  <RespondingBusinessActivity nameID="iSURF_ECT_ResBA"
        name="Exception Criteria Responding Business Activity"/>
</Notification>
<BusinessCollaboration name="Buyer Order Forecast" nameID="Buyer_Order_Forecast">
  <Role name="ABC Stores" nameID="IDBuyer"/>
  <Role name="XYZ Manufacturing" nameID="IDSeller"/>
  <TimeToPerform/>
  <BusinessTransactionActivity name="Send Exception Criteria BTA-S2B"
        nameID="iSURF_Send_EC_BTA" businessTransactionRef="iSURF_Exception_Criteria_Transaction">
    <Documentation>CPFR Step 1</Documentation>
    <Documentation>Exception Criteria</Documentation>
    <TimeToPerform/>
    <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_ECT_Initiator"/>
    <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_ECT_Responder"/>
  </BusinessTransactionActivity>
  <BusinessTransactionActivity name="Send Exception Criteria BTA-S2B"
        nameID="iSURF_Send_ECR_BTA_S2B" businessTransactionRef="iSURF_Exception_Criteria_Transaction">
    <Documentation>CPFR Step 1</Documentation>
    <Documentation>Exception Criteria Revision</Documentation>
    <TimeToPerform/>
    <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_ECT_Initiator"/>
    <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_ECT_Responder"/>
  </BusinessTransactionActivity>
  <BusinessTransactionActivity name="Send Exception Criteria BTA-B2S"
        nameID="iSURF_Send_ECR_BTA_B2S" businessTransactionRef="iSURF_Exception_Criteria_Transaction">
    <Documentation>CPFR Step 1</Documentation>
    <Documentation>Exception Criteria Revision</Documentation>
    <TimeToPerform/>
    <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_ECT_Responder"/>
    <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_ECT_Initiator"/>
  </BusinessTransactionActivity>
  <BusinessTransactionActivity name="Send Retail Event BTA-S2B"
        nameID="iSURF_Send_RE_BTA" businessTransactionRef="iSURF_Retail_Event_Transaction">
```

```
  <Documentation>CPFR Step 2</Documentation>
  <Documentation>Retail Event</Documentation>
  <TimeToPerform/>
  <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_RET_Initiator"/>
  <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_RET_Responder"/>
</BusinessTransactionActivity>
<BusinessTransactionActivity name="Send Exception Criteria BTA-B2S"
        nameID="iSURF_Send_RER_BTA_B2S" businessTransactionRef="iSURF_Exception_Criteria_Transaction">
  <Documentation>CPFR Step 2</Documentation>
  <Documentation>Retail Event Revision</Documentation>
  <TimeToPerform/>
  <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_ECT_Responder"/>
  <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_ECT_Initiator"/>
</BusinessTransactionActivity>
<BusinessTransactionActivity name="Send Exception Criteria BTA-S2B"
        nameID="iSURF_Send_RER_BTA_S2B" businessTransactionRef="iSURF_Exception_Criteria_Transaction">
  <Documentation>CPFR Step 2</Documentation>
  <Documentation>Retail Event Revision</Documentation>
  <TimeToPerform/>
  <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_ECT_Initiator"/>
  <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_ECT_Responder"/>
</BusinessTransactionActivity>
<BusinessTransactionActivity name="Send Trade Item Location Profile BTA-S2B"
        nameID="iSURF_Send_TILP_BTA"
        businessTransactionRef="iSURF_Trade_Item_Location_Profile_Transaction">
  <Documentation>CPFR Step 2</Documentation>
  <Documentation>TILP</Documentation>
  <TimeToPerform/>
  <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_TILPT_Initiator"/>
  <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_TILPT_Responder"/>
</BusinessTransactionActivity>
<BusinessTransactionActivity name="Send Exception Criteria BTA-B2S"
        nameID="iSURF_Send_TILPR_BTA_B2S"
        businessTransactionRef="iSURF_Exception_Criteria_Transaction">
  <Documentation>CPFR Step 2</Documentation>
  <Documentation>TILPR</Documentation>
  <TimeToPerform/>
  <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_ECT_Responder"/>
  <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_ECT_Initiator"/>
</BusinessTransactionActivity>
<BusinessTransactionActivity name="Send Exception Criteria BTA-S2B"
        nameID="iSURF_Send_TILPR_BTA_S2B"
        businessTransactionRef="iSURF_Exception_Criteria_Transaction">
  <Documentation>CPFR Step 2</Documentation>
  <Documentation>TILPR</Documentation>
  <TimeToPerform/>
  <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_ECT_Initiator"/>
  <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_ECT_Responder"/>
```

```
</BusinessTransactionActivity>
<BusinessTransactionActivity name="Send Termination Message BTA-S2B"
        nameID="iSURF_Send_TM_BTA"
        businessTransactionRef="iSURF_Termination_Message_Transaction">
  <Documentation>CPFR Step 2</Documentation>
  <Documentation>Termination Message</Documentation>
  <TimeToPerform duration="PT4H"/>
  <Performs currentRoleRef="IDSeller" performsRoleRef="iSURF_TMT_Initiator"/>
  <Performs currentRoleRef="IDBuyer" performsRoleRef="iSURF_TMT_Responder"/>
</BusinessTransactionActivity>
<Success name="Success" nameID="iSURF_COF_Success"/>
<Failure name="Failure" nameID="iSURF_COF_Failure"/>
<Start>
  <ToLink toBusinessStateRef="iSURF_Send_EC_BTA"/>
</Start>
<Transition nameID="TR0">
  <FromLink fromBusinessStateRef="iSURF_Send_EC_BTA"/>
  <ToLink toBusinessStateRef="iSURF_Send_ECR_BTA_B2S"/>
</Transition>
<Variable name="EC Accepted" nameID="EC_Accepted"
        businessTransactionActivityRef="iSURF_Send_ECR_BTA_S2B"
        businessDocumentRef="Exception_Criteria_Revision_Document">
  <ConditionExpression expressionLanguage="XPath1"
        expression="//ExceptionCriteria[@status='APPROVED']"/>
</Variable>
<Variable name="EC Not Accepted" nameID="EC_Not_Accepted"
        businessTransactionActivityRef="iSURF_Send_ECR_BTA_S2B"
        businessDocumentRef="Exception_Criteria_Revision_Document">
  <ConditionExpression expressionLanguage="XPath1"
        expression="//ExceptionCriteria[@status!='APPROVED']"/>
</Variable>
<Decision nameID="D0">
  <FromLink fromBusinessStateRef="iSURF_Send_ECR_BTA_S2B"/>
  <ToLink toBusinessStateRef="iSURF_Send_RE_BTA">
    <ConditionExpression expressionLanguage="XPath1" expression="EC_Accepted"/>
  </ToLink>
  <ToLink toBusinessStateRef="iSURF_Send_ECR_BTA_B2S">
    <ConditionExpression expressionLanguage="XPath1" expression="EC_Not_Accepted"/>
  </ToLink>
</Decision>
<Transition nameID="TR1">
  <FromLink fromBusinessStateRef="iSURF_Send_ECR_BTA_B2S"/>
  <ToLink toBusinessStateRef="iSURF_Send_ECR_BTA_S2B"/>
</Transition>
<Transition nameID="TR2">
  <FromLink fromBusinessStateRef="iSURF_Send_RE_BTA"/>
  <ToLink toBusinessStateRef="iSURF_Send_RER_BTA_B2S"/>
</Transition>
```

```
<Transition nameID="TR3">
  <FromLink fromBusinessStateRef="iSURF_Send_RER_BTA_B2S"/>
  <ToLink toBusinessStateRef="iSURF_Send_RER_BTA_S2B"/>
</Transition>
<Variable name="RE Accepted" nameID="RE_Accepted"
        businessTransactionActivityRef="iSURF_Send_RER_BTA_S2B"
        businessDocumentRef="Retail_Event_Revision_Document">
  <ConditionExpression expressionLanguage="XPath1"
        expression="//RetailEvent[@status='APPROVED']"/>
</Variable>
<Variable name="RE Not Accepted" nameID="RE_Not_Accepted"
        businessTransactionActivityRef="iSURF_Send_RER_BTA_S2B"
        businessDocumentRef="Retail_Event_Revision_Document">
  <ConditionExpression expressionLanguage="XPath1"
        expression="//RetailEvent[@status!='APPROVED']"/>
</Variable>
<Decision nameID="D1">
  <FromLink fromBusinessStateRef="iSURF_Send_RER_BTA_S2B"/>
  <ToLink toBusinessStateRef="iSURF_Send_TILP_BTA">
    <ConditionExpression expressionLanguage="XPath1" expression="RE_Accepted"/>
  </ToLink>
  <ToLink toBusinessStateRef="iSURF_Send_RER_BTA_B2S">
    <ConditionExpression expressionLanguage="XPath1" expression="RE_Not_Accepted"/>
  </ToLink>
</Decision>
<Transition nameID="TR4">
  <FromLink fromBusinessStateRef="iSURF_Send_TILP_BTA"/>
  <ToLink toBusinessStateRef="iSURF_Send_TILPR_BTA_B2S"/>
</Transition>
<Transition nameID="TR5">
  <FromLink fromBusinessStateRef="iSURF_Send_TILPR_BTA_B2S"/>
  <ToLink toBusinessStateRef="iSURF_Send_TILPR_BTA_S2B"/>
</Transition>
<Variable name="TILP Accepted" nameID="TILP_Accepted"
        businessTransactionActivityRef="iSURF_Send_TILPR_BTA_S2B"
        businessDocumentRef="Trade_Item_Location_Profile_Revision_Document">
  <ConditionExpression expressionLanguage="XPath1"
        expression="//TradeItemLocationProfile[@status='APPROVED']"/>
</Variable>
<Variable name="TILP Not Accepted" nameID="TILP_Not_Accepted"
        businessTransactionActivityRef="iSURF_Send_TILPR_BTA_S2B"
        businessDocumentRef="Trade_Item_Location_Profile_Revision_Document">
  <ConditionExpression expressionLanguage="XPath1"
        expression="//TradeItemLocationProfile[@status!='APPROVED']"/>
</Variable>
<Decision nameID="D2">
  <FromLink fromBusinessStateRef="iSURF_Send_TILPR_BTA_S2B"/>
  <ToLink toBusinessStateRef="iSURF_Send_TM_BTA">
```

```
        <ConditionExpression expressionLanguage="XPath1" expression="TILP_Accepted"/>
      </ToLink>
      <ToLink toBusinessStateRef="iSURF_Send_TILPR_BTA_B2S">
        <ConditionExpression expressionLanguage="XPath1" expression="TILP_Not_Accepted"/>
      </ToLink>
    </Decision>
    <Decision name="Finalize Planning Decision" nameID="D3">
      <Documentation/>
      <FromLink fromBusinessStateRef="iSURF_Send_TM_BTA"/>
      <ToLink toBusinessStateRef="iSURF_COF_Success">
        <ConditionExpression expressionLanguage="ConditionGuardValue" expression="iSURF_COF_Success"/>
      </ToLink>
      <ToLink toBusinessStateRef="iSURF_Send_EC_BTA">
        <ConditionExpression expressionLanguage="ConditionGuardValue" expression="iSURF_COF_Failure"/>
      </ToLink>
    </Decision>
  </BusinessCollaboration>
  <Signal name="ReceiptAcknowledgement" nameID="ra2">
    <Specification location="http://docs.oasis-open.org/ebxmlbp/ebbp-signals-2.0"
            name="ReceiptAcknowledgement" nameID="rabpss2"/>
  </Signal>
  <Signal name="ReceiptAcknowledgementException" nameID="rae2">
    <Specification location="http://docs.oasis-open.org/ebxmlbp/ebbp-signals-2.0"
            name="ReceiptAcknowledgementException" nameID="raebpss2"/>
  </Signal>
  <Signal name="AcceptanceAcknowledgement" nameID="aa2">
    <Specification location="http://docs.oasis-open.org/ebxmlbp/ebbp-signals-2.0"
            name="AcceptanceAcknowledgement" nameID="aabpss2"/>
  </Signal>
  <Signal name="AcceptanceAcknowledgementException" nameID="aae2">
    <Specification location="http://docs.oasis-open.org/ebxmlbp/ebbp-signals-2.0"
            name="AcceptanceAcknowledgementException" nameID="aaebpss2"/>
  </Signal>
  <Signal name="GeneralException" nameID="ge2">
    <Specification location="http://docs.oasis-open.org/ebxmlbp/ebbp-signals-2.0"
            name="GeneralException" nameID="gebpss2"/>
  </Signal>
</ProcessSpecification>
```

APPENDIX B

AN EXAMPLE BPEL DEFINITION GENERATED BY CPFR DESIGNER TOOL FOR THE MANUFACTURER

```
<?xml version="1.0" encoding="utf-8"?>
<process name="CPFRManufacturer"
         targetNamespace="http://CPFRManufacturer"
         ext:createTargetXPath="yes"
         suppressJoinFailure="yes"
         xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
         xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
         xmlns:isu="http://www.srdc.com.tr/isu/"
         xmlns:ext="http://www.activebpel.org/2006/09/bpel/extension/query_handling"
         xmlns:manubpel="http://www.srdc.com.tr/manufacturer/bpel"
         xmlns:manufacturer="http://www.srdc.com.tr/manufacturer/"
         xmlns:retailer="http://www.srdc.com.tr/retailer/"
         xmlns:retbpel="http://www.srdc.com.tr/retailer/bpel"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <extensions>
    <extension mustUnderstand="yes"
         namespace="http://www.activebpel.org/2006/09/bpel/extension/query_handling"/>
  </extensions>
  <partnerLinks>
    <partnerLink name="sendMessageToISUPL" partnerLinkType="manubpel:sendMessageToISUPLType"
         partnerRole="sendMessageToISURole"/>
    <partnerLink name="sendMessageToRetailerBpelPL"
         partnerLinkType="manubpel:sendMessageToRetailerBpelPLType"
         partnerRole="sendMessageToRetailerBpelRole"/>
    <partnerLink name="sendMessagetoManufPL" partnerLinkType="manubpel:sendMessagetoManufPLType"
         partnerRole="sendMessageToManufacturerRole"/>
    <partnerLink name="receiveDataFromRetailerBpelPL"
         partnerLinkType="manubpel:receiveDataFromRetailerBpelPLType"
         myRole="receiveDataFromRetailerBpelRole"/>
```

```
  <partnerLink name="receiveDataFromManufPL"
       partnerLinkType="manubpel:receiveDataFromManufPLType" myRole="retrieveDataFromManufacturerRole"/>
</partnerLinks>
<variables>
  <variable name="translateDocumentRequest" messageType="xsd:string"/>
  <variable name="translateDocumentResponse" messageType="xsd:string"/>
  <variable name="op_Buyer" messageType="retbpel:Exception_Criteria"/>
  <variable name="op_Seller" messageType="manubpel:Exception_Criteria"/>
  <variable name="op_Buyer" messageType="retbpel:Exception_Criteria"/>
  <variable name="op_Seller" messageType="manubpel:Exception_Criteria"/>
  <variable name="op_Buyer" messageType="retbpel:Exception_Criteria"/>
  <variable name="op_Seller" messageType="manubpel:Exception_Criteria"/>
  <variable name="op_Buyer" messageType="retbpel:Retail_Event"/>
  <variable name="op_Seller" messageType="manubpel:Retail_Event"/>
  <variable name="op_Buyer" messageType="retbpel:Exception_Criteria"/>
  <variable name="op_Seller" messageType="manubpel:Exception_Criteria"/>
  <variable name="op_Buyer" messageType="retbpel:Exception_Criteria"/>
  <variable name="op_Seller" messageType="manubpel:Exception_Criteria"/>
  <variable name="op_Buyer" messageType="retbpel:Trade_Item_Location_Profile"/>
  <variable name="op_Seller" messageType="manubpel:Trade_Item_Location_Profile"/>
  <variable name="op_Buyer" messageType="retbpel:Exception_Criteria"/>
  <variable name="op_Seller" messageType="manubpel:Exception_Criteria"/>
  <variable name="op_Buyer" messageType="retbpel:Exception_Criteria"/>
  <variable name="op_Seller" messageType="manubpel:Exception_Criteria"/>
  <variable name="op_Buyer" messageType="retbpel:Termination_Message"/>
  <variable name="op_Seller" messageType="manubpel:Termination_Message"/>
  <variable name="op_arrived_Seller" type="xsd:boolean"/>
</variables>
<sequence>
  <repeatUntil>
    <sequence>
      <receive name="receive_iSURF_Send_EC_BTA" partnerLink="receiveDataFromManufPL"
          operation="opfromManufacturer" variable="op_Seller"/>
      <assign>
        <copy>
          <from part="message" variable="op_Seller"/>
          <to part="message" variable="translateDocumentRequest"/>
        </copy>
        <copy>
          <from>'Seller'</from>
          <to part="from" variable="translateDocumentRequest"/>
        </copy>
      </assign>
      <invoke name="InvokeISUforMapping" partnerLink="sendMessageToISUPL"
          operation="translateDocument" inputVariable="translateDocumentRequest"
          outputVariable="translateDocumentResponse"/>
      <assign>
        <copy>
```

```
    <from part="message" variable="translateDocumentResponse"/>
    <to part="message" variable="op_Buyer"/>
  </copy>
</assign>
<invoke name="InvokeRetailerBPEL" partnerLink="sendMessageToRetailerBpelPL"
    operation="opfromManufBPEL" inputVariable="op_Buyer"/>
<repeatUntil>
  <sequence>
    <receive name="receive_iSURF_Send_ECR_BTA_B2S" partnerLink="receiveDataFromRetailerBpelPL"
        operation="opfromRetBPEL" variable="op_Seller"/>
    <invoke name="InvokeManufBPEL" partnerLink="sendMessagetoManufPL" operation="op"
           inputVariable="op_Seller"/>
    <receive name="receive_iSURF_Send_ECR_BTA_S2B" partnerLink="receiveDataFromManufPL"
            operation="opfromManufacturer" variable="op_Seller"/>
    <assign>
      <copy>
        <from part="message" variable="op_Seller"/>
        <to part="message" variable="translateDocumentRequest"/>
      </copy>
      <copy>
        <from>'Seller'</from>
        <to part="from" variable="translateDocumentRequest"/>
      </copy>
    </assign>
    <invoke name="InvokeISUforMapping" partnerLink="sendMessageToISUPL"
           operation="translateDocument" inputVariable="translateDocumentRequest"
           outputVariable="translateDocumentResponse"/>
    <assign>
      <copy>
        <from part="message" variable="translateDocumentResponse"/>
        <to part="message" variable="op_Buyer"/>
      </copy>
    </assign>
    <invoke name="InvokeRetailerBPEL" partnerLink="sendMessageToRetailerBpelPL"
           operation="opfromManufBPEL" inputVariable="op_Buyer"/>
  </sequence>
  <condition>//ExceptionCriteria[@status!='APPROVED']</condition>
</repeatUntil>
<receive name="receive_iSURF_Send_RE_BTA" partnerLink="receiveDataFromManufPL"
        operation="opfromManufacturer" variable="op_Seller"/>
<assign>
  <copy>
    <from part="message" variable="op_Seller"/>
    <to part="message" variable="translateDocumentRequest"/>
  </copy>
  <copy>
    <from>'Seller'</from>
    <to part="from" variable="translateDocumentRequest"/>
```

```
  </copy>
</assign>
<invoke name="InvokeISUforMapping" partnerLink="sendMessageToISUPL"
        operation="translateDocument" inputVariable="translateDocumentRequest"
        outputVariable="translateDocumentResponse"/>
<assign>
  <copy>
    <from part="message" variable="translateDocumentResponse"/>
    <to part="message" variable="op_Buyer"/>
  </copy>
</assign>
<invoke name="InvokeRetailerBPEL" partnerLink="sendMessageToRetailerBpelPL"
        operation="opfromManufBPEL" inputVariable="op_Buyer"/>
<repeatUntil>
  <sequence>
    <receive name="receive_iSURF_Send_RER_BTA_B2S" partnerLink="receiveDataFromRetailerBpelPL"
             operation="opfromRetBPEL" variable="op_Seller"/>
    <invoke name="InvokeManufBPEL" partnerLink="sendMessagetoManufPL" operation="op"
            inputVariable="op_Seller"/>
    <receive name="receive_iSURF_Send_RER_BTA_S2B" partnerLink="receiveDataFromManufPL"
             operation="opfromManufacturer" variable="op_Seller"/>
    <assign>
      <copy>
        <from part="message" variable="op_Seller"/>
        <to part="message" variable="translateDocumentRequest"/>
      </copy>
      <copy>
        <from>'Seller'</from>
        <to part="from" variable="translateDocumentRequest"/>
      </copy>
    </assign>
    <invoke name="InvokeISUforMapping" partnerLink="sendMessageToISUPL"
            operation="translateDocument" inputVariable="translateDocumentRequest"
            outputVariable="translateDocumentResponse"/>
    <assign>
      <copy>
        <from part="message" variable="translateDocumentResponse"/>
        <to part="message" variable="op_Buyer"/>
      </copy>
    </assign>
    <invoke name="InvokeRetailerBPEL" partnerLink="sendMessageToRetailerBpelPL"
            operation="opfromManufBPEL" inputVariable="op_Buyer"/>
  </sequence>
  <condition>//RetailEvent[@status!='APPROVED']</condition>
</repeatUntil>
<receive name="receive_iSURF_Send_TILP_BTA" partnerLink="receiveDataFromManufPL"
         operation="opfromManufacturer" variable="op_Seller"/>
<assign>
```

```
  <copy>
    <from part="message" variable="op_Seller"/>
    <to part="message" variable="translateDocumentRequest"/>
  </copy>
  <copy>
    <from>'Seller'</from>
    <to part="from" variable="translateDocumentRequest"/>
  </copy>
</assign>
<invoke name="InvokeISUforMapping" partnerLink="sendMessageToISUPL"
        operation="translateDocument" inputVariable="translateDocumentRequest"
        outputVariable="translateDocumentResponse"/>
<assign>
  <copy>
    <from part="message" variable="translateDocumentResponse"/>
    <to part="message" variable="op_Buyer"/>
  </copy>
</assign>
<invoke name="InvokeRetailerBPEL" partnerLink="sendMessageToRetailerBpelPL"
        operation="opfromManufBPEL" inputVariable="op_Buyer"/>
<repeatUntil>
  <sequence>
    <receive name="receive_iSURF_Send_TILPR_BTA_B2S" partnerLink="receiveDataFromRetailerBpelPL"
             operation="opfromRetBPEL" variable="op_Seller"/>
    <invoke name="InvokeManufBPEL" partnerLink="sendMessagetoManufPL"
            operation="op" inputVariable="op_Seller"/>
    <receive name="receive_iSURF_Send_TILPR_BTA_S2B" partnerLink="receiveDataFromManufPL"
             operation="opfromManufacturer" variable="op_Seller"/>
    <assign>
      <copy>
        <from part="message" variable="op_Seller"/>
        <to part="message" variable="translateDocumentRequest"/>
      </copy>
      <copy>
        <from>'Seller'</from>
        <to part="from" variable="translateDocumentRequest"/>
      </copy>
    </assign>
    <invoke name="InvokeISUforMapping" partnerLink="sendMessageToISUPL"
            operation="translateDocument" inputVariable="translateDocumentRequest"
            outputVariable="translateDocumentResponse"/>
    <assign>
      <copy>
        <from part="message" variable="translateDocumentResponse"/>
        <to part="message" variable="op_Buyer"/>
      </copy>
    </assign>
    <invoke name="InvokeRetailerBPEL" partnerLink="sendMessageToRetailerBpelPL"
```

```
            operation="opfromManufBPEL" inputVariable="op_Buyer"/>
    </sequence>
    <condition>//TradeItemLocationProfile[@status!='APPROVED']</condition>
  </repeatUntil>
  <assign>
    <copy>
      <from expression="true()"/>
      <to variable="op_arrived_Seller"/>
    </copy>
  </assign>
  <pick>
    <onMessage partnerLink="receiveDataFromManufPL" operation="opfromManufacturer"
              variable="op_Seller">
      <sequence>
        <assign>
          <copy>
            <from part="message" variable="op_Seller"/>
            <to part="message" variable="translateDocumentRequest"/>
          </copy>
          <copy>
            <from>'Seller'</from>
            <to part="from" variable="translateDocumentRequest"/>
          </copy>
        </assign>
        <invoke name="InvokeISUforMapping" partnerLink="sendMessageToISUPL"
                operation="translateDocument" inputVariable="translateDocumentRequest"
                outputVariable="translateDocumentResponse"/>
        <assign>
          <copy>
            <from part="message" variable="translateDocumentResponse"/>
            <to part="message" variable="op_Buyer"/>
          </copy>
        </assign>
        <invoke name="InvokeRetailerBPEL" partnerLink="sendMessageToRetailerBpelPL"
                operation="opfromManufBPEL" inputVariable="op_Buyer"/>
      </sequence>
    </onMessage>
    <onAlarm>
      <for>'PT4H'</for>
      <assign>
        <copy>
          <from expression="false()"/>
          <to variable="op_arrived_Seller"/>
        </copy>
      </assign>
    </onAlarm>
  </pick>
</sequence>
```

```
      <condition>op_arrived_Seller</condition>
    </repeatUntil>
  </sequence>
</process>
```