José Alexandre Pires Ferreira

Mestre em Engenharia Electrotécnica e de Computadores

Monitoring and Information Alignment in Pursuit of an IoT-Enabled Self-Sustainable Interoperability

Dissertação para obtenção do Grau de Doutor em
Engenharia Electrotécnica e de Computadores – Especialidade em Sistemas de Informação Industriais

Orientador: Doutor Ricardo Luís Rosa Jardim Gonçalves, Professor Auxiliar com Agregação, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa
Co-orientador: Doutor Carlos Manuel Melo Agostinho, Professor Auxiliar Convidado, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa

Júri:

Presidente:	Prof. Doutor Jorge Joaquim Pamiés Teixeira
Arguentes:	Prof. Doutor Raul Poler Escoto
	Prof. Doutor João Pedro Mendonça de Assunção da Silva
Vogais:	Prof. Doutor Jorge Joaquim Pamiés Teixeira
	Prof. Doutor Ricardo Luís Rosa Jardim Gonçalves
	Prof. Doutor Manuel Martins Barata
	Prof. Doutor Carlos Eduardo Dias Coutinho
	Prof Doutor José António Barata de Oliveira
	Doutora Maria do Carmo Correia Marques

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Dezembro 2017

Copyright

Monitoring and Information Alignment in Pursuit of an IoT-Enabled Self-Sustainable Interoperability

To my family

Acknowledgements

I would like to thank all the people who, in some way, supported me during the course of my course and this dissertation.

First, my family, who never gave up on me during this phase of my life, who supported me all those years and endured all the moments, lived during this doctorate.

To all my GRIS colleagues, especially to Carlos Agostinho, who helped me and believed me at this moment, although it often seemed that I was not going to arrive, pushing me to do a better job and successfully complete this dissertation.

My advisor, Dr. Ricardo Gonçalves for believing in my abilities and giving me his advice for the successful completion of this work, and the great opportunity to work in GRIS.

For my great friends, Ana Mamede and Carla Mamede for always giving me support and motivation to finally finish this dissertation, which gave me hope and believed in me all the time, made me believe in myself, although I do not always show how much I appreciate to both of you. To Catarina Ferreira, although we met a short time, you have been a good friend and helped me calm down whenever I need so thanks to three.

For my special friends, who shared all the hard work during the course, this took many years, but it was finally over. Thank you, Paulo Polaina, Francisco Cavaco and António Paulo, during all these years together, vacations and the great nights we spent together.

Finally, for all Badjoras those in recent years were great fellowship at work, dinners, and drinks, particularly André Coelho, Flávio Silva, João Lourenço, João Soares, João Ralo and Joaquim Pereira. For Carlos Raposo, Fábio Lopes and João Soares for the support they gave in this work, they were a great contribution and supported me to finish this dissertation.

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2011-2014) under grant agreement n° 285132 (http://cordis.europa.eu/project/rcn/99927_en.html) and the European Union Horizon 2020 Programme (H2020/2015-2017) under grant agreement nº 636909 (http://cordis.europa.eu/project/rcn/193440_en.html).

I would like to thank FCT – Foundation for Science and Technology funding through a Ph.D. scholarship (SFRH/BD/84803/2012).

Abstract

To remain competitive with big corporations, small and medium-sized enterprises (SMEs) often need to be more dynamic, adapt to new business situations, react faster, and thereby survive in today's global economy. To do so, SMEs normally seek to create consortiums, thus gaining access to new and more opportunities. However, this strategy may also lead to complications. Due to the different sources of enterprise models and semantics, organizations are experiencing difficulties in seamlessly exchanging vital information via electronic means. In their attempt to address this issue, most seek to achieve interoperability by establishing peer-to-peer mappings with different business partners, or by using neutral data standards to regulate communications in optimized networks. Moreover, systems are more and more dynamic, frequently changing to answer new customer's requirements, causing new interoperability problems and a reduction of efficiency. Another situation that is constantly changing is the devices used in the enterprises, as the Enterprise Information Systems, devices are used to register internal data, and to be used to monitor several aspects. These devices are constantly changing, following the evolution and growth of the market. So, it is important to monitor these devices and doing a model representation of them. This dissertation proposes a self-sustainable interoperable framework to monitor existing enterprise information systems and their devices, monitor the device/enterprise network for changes and automatically detecting model changes. With this, network harmonization disruptions are detected in a timely way, and possible solutions are suggested to regain the interoperable status, thus enhancing robustness for reaching sustainability of business networks along time.

Keywords

Self-Sustainable Interoperability; Model Morphisms; Multi-Agent Systems; Monitoring and Assessment; Knowledge Management; Sensing Enterprise; Internet of Things.

Resumo

Para manter a competitividade com as grandes corporações, as pequenas e médias empresas (PMEs) tem de se tornar mais dinâmicas, adaptar-se a novas situações, reagir mais depressa e, assim sobreviver na inconstante economia global de hoje. Para isso, as PMEs procuram criar consórcios, ganhando com isso, acesso a novas e mais oportunidades. No entanto, esta estratégia pode também conduzir a complicações. Devido às diferentes fontes de modelos empresariais e semântica, as organizações encontram dificuldades, quando começam a troca de informação eletrónica importante para o consórcio. As empresas tentam resolver estes problemas, procurando alcançar a interoperabilidade através da criação de mapeamentos ponto-a-ponto com os diferentes parceiros de negócios, ou usando padrões de dados neutros para regular as comunicações em redes otimizadas. Além disso, os sistemas são cada vez mais dinâmicos, dado que mudam frequentemente para responder ás necessidades dos clientes, causando novos problemas de interoperabilidade e uma redução de eficiência. Outra situação que está em constante mudança, são os dispositivos utilizados nas empresas como os sistemas de informação de empresas, os dispositivos são usados para registar os dados internos, e para ajudar na monitorização da empresa. Estes dispositivos estão em constante mudança, acompanhando a evolução do mercado. Portanto, é importante monitorizar esses dispositivos e representa-los em modelos. Esta dissertação propõe uma estrutura interoperável auto-sustentável, para monitorizar os sistemas de informação das empresas e os seus dispositivos, monitorizar a rede de empresas/dispositivos à procura de mudanças, e detetar automaticamente alterações ao modelo. Assim, as interrupções de harmonização na rede são detetadas a tempo, e as possíveis soluções são sugeridas para recuperar o status de interoperabilidade, reforçando assim a robustez para alcançar a sustentabilidade da rede de empresas ao longo do tempo.

Palavras-Chave

Interoperabilidade Auto-Sustentável; Modelo de Morfismos; Sistema de Multi-Agentes; Monitorização e Avaliação de Sistemas; Gestão de Conhecimento, Empresa Sensorial, Internet das Coisas.

Table of Contents

Table of Figures

List of Tables

List of Acronyms

#

MapT ► 5-Tupple Mapping

A

ATC ► Abstract Test Cases

ATL ► ATLAS Transformation Language

B

BP ► Blueprints

BPEL ► Business Process Execution Language

BPMN ► Business Process Management Notation

C

C^{4}IF ► Connection, Communication, Consolidation, Collaboration Interoperability Framework

CAS ► Complex Adaptive System

CAS-SIF ► CAS to support Sustainable Interoperability Framework

CEP ► Complex Event Processing

CII ► Component for Integration and Intelligence

CIM ► Computer Integrated Manufacturing

CIMOSA ► Computer Integrated Manufacturing Open System Architecture

CM ► Communicator Mediator

CPS ► Cyber-Physical Systems

D

DB ► Data Base

DEVMAN ► Device Management for Interoperability in a Physical Network

DMN ► Dynamic Manufacturer Network

DoDAF ► Department of Defense Architecture Framework

E

EA ► Enterprise Architecture

EC ► Enterprise Collaboration

EIS ► Enterprise Information Systems

EI ► Enterprise Interoperability

EIMM ► Enterprise Interoperability Maturity Model

EISB ► EI Science Base

EM ► Enterprise Modeling

EPL ► Event Processing Language

ERP ► Enterprise Resource Planning

ESB ► Enterprise Service Bus

F

FoF ► Factory of the Future

G

GERAM ►	Generalised Enterprise Reference Architecture and Methodology
GRIS ►	Group for Research in Interoperability of Systems

I

IEEE ►	Institute of Electrical and Electronics Engineers
IEC ►	International Electrotechnical Commission
IEM ►	Integrated Enterprise Modeling
IoT ►	Internet of Things
IoT-A ►	Internet of Things Architecture
IPyM ►	Interoperability Efficiency Pyramid Model
IS ►	Information System
ISO ►	International Organisation for Standardization (http://www.iso.org)
IT ►	Information Technology
IUT ►	Implementation Under Tests

K

KB ►	Knowledge Base
KPI ►	Key Performance Indicator

L

LC ►	Life-Cycle
LISI ►	Levels of Information System Interoperability
LSE ►	Liquid and Sensing Enterprise

M

MAS ►	Multi-Agent System
MBSE ►	Model-Based Systems Engineering
MDA ►	Model Driven Architecture
MDI ►	Model Driven Interoperability
MDSEA ►	Model-Driven Service Engineering Architecture
MEI ►	Minimal Effective Interoperability
MIRAI ►	Monitoring morphIsms to suppoRt sustAinable Interoperability of enterprise systems
MoMo ►	Model Morphism
MPI ►	Maximal Potential Interoperability
MTS ►	Methods for Testing and Specification

N

NAF ►	NATO Architecture Framework
NASA ►	National Aeronautics and Space Administration
NM ►	Network Monitor

O

OBM ►	OSMOSE Business Models
OMG ►	Object Management Group (http://www.omg.org)
OSMOSE ►	OSMOsis applications for the Sensing Enterprise

OTIM ►	OSMOSE Technology Independent Models
OTSM ►	OSMOSE Technology Specific Models

P

PLC ►	Product Life-Cycle
PI ►	Performance Indicator

S

SE ►	Sensing Enterprise
SM ►	System Monitor
SMAP ►	Semantic MAPping
SME ►	Small and Medium Enterprises
SOMF ►	Service-Oriented Modeling Framework
SQuaRE ►	Software Product Quality Requirement and Evaluation
SSIF ►	Self-Sustainable Interoperability Framework
SSN ►	Semantic Sensor Network
STEP ►	Standard for the Exchange of Product Data
SUT ►	System Under Test

T

TOGAF ►	The Open Group Architecture Framework
TTCN ►	Tree and Tabular Combined Notation

W

WS ►	Web Services

X

XML ►	Extensible Markup Language

1. INTRODUCTION

In an era of globalization, organizations cooperate with each other to reach new markets, increase production, and decrease production costs, hoping to achieve better results and, consequently increase sales. However, globalization implies working with organizations scattered over the world, thereby bringing problems. Sometimes they do not have a proper concern to manage the processes well, due to the often-daunting scale of production. Even good managers sometimes pass up the opportunity to use software solutions that support the management of network production. Such software is called Enterprise Information Systems (EIS) and are created to support the management of the various resources (personnel, material, equipment, process, etc.). Often this software is able to monitor production step by step, and sometimes identify problems early enough to afford the time to solve them. However, this does not oblige the different enterprises to use the same enterprise systems, and as a result, different enterprises often have their own system (ATHENA IP, 2007b).

The use of different EIS causes problems among the various enterprises in the network, as each one uses their own standard, complicating the exchange of information between them. This is a critical point when working with other enterprises because it can happen that the information is misinterpreted, causing delays in production and perhaps the loss of considerable profits. An example is the United States health system (CalgaryScientify, 2013), a system that is divided into several health sectors, and each contains its system, creating serious problems of interoperability whenever a patient needs their record to show to another doctor, whose objective is to support the elaboration of his health condition. It is estimated that in the case of the health system, attempting to solve the interoperability problem could save about $30 billion and at the same time improve patient care and hospital safety. The implications for the total potential cost savings of health care across the global community, if this issue can be addressed with barrier-breaking technology, are then massive.

In this example, it is possible to identify the problems that can occur when working in an Enterprise Collaboration (EC). Each enterprise has its own EIS, and consequently its own standards, models, and devices, creating friction when interacting with another enterprise. On the other hand, with each passing day, the need of sharing information is becoming more and more important for organizations. It is even more important that failures never occur in that information exchange. It is important to support the integration of the Product Life-Cycle (PLC) phases, since manufacturers, distributors, designers, retailers, and warehouses, all have their proprietary solutions. Maintaining interoperability between them is therefore crucial (Ricardo Jardim-Goncalves et al., 2011).

Maintaining good functional collaboration and a good interoperability status between enterprises is thus not an easy task. Yet, whenever it is achieved, it brings benefits to the organizations, including: shorter response times, reduced cycle times from order to cash, reduced inventories, and a reduction in the number of costly errors, keeping enterprises ahead of competition and positioning them for world-class performance (Ricardo Jardim-Goncalves et al., 2011), thus using enterprise systems to describe the PLC, with the aim of achieving better results. Reaching an interoperability status is a difficult task, but when it is achieved, it brings several advantages to enterprises, notably the communication that will be interoperable, avoiding the breaking on the harmonization. In this situation, an interoperable environment inside the network is created.

Achieving an interoperable environment is the aim of the enterprises, allowing them to enter and leave a network without problems and difficulties. However, only *"diamonds are forever"*, and in a network, several things can occur to disrupt or defeat the interoperability status. To avoid this, it is necessary to create a system that is able to monitor the network searching for problems, and every time that one is detected, be able to repair it in time, perhaps avoiding serious problems on the network. This scenario, in which the network is able to create an interoperability system and at the same time be sustainable, is called a "self-sustainable interoperable network".

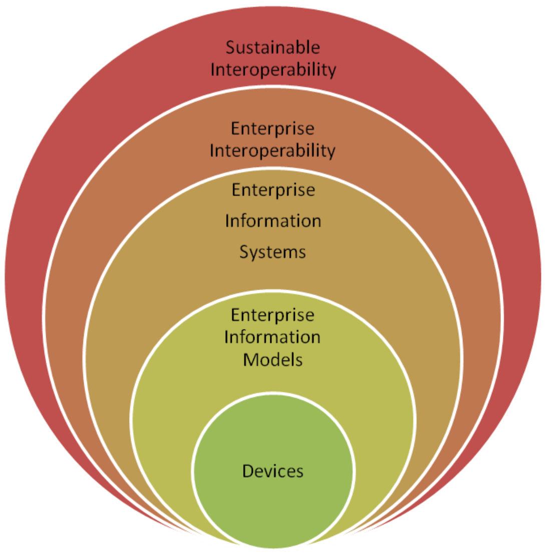

Sustainable Interoperability
Enterprise Interoperability
Enterprise Information Systems
Enterprise Information Models
Devices

Figure 1-1: Sustainable Interoperability Paradigm.

To better understand sustainable interoperability, one can follow Figure 1-1 that looks like an onion, in which each layer represents a better interoperability status. This onion starts in the center with the representation of the devices (first layer), i.e. objects, machines, or pieces of equipment made for some special purpose implemented a device[1]. For this purpose, throughout this work, a device are all resources that are related to the enterprise, e.g. models, sensors, machines, humans, etc. Since the devices are the various resources of the enterprise, they must be represented somewhere in a system, and the second layer represents the models that describe the devices, being the described in information models, e.g. as resources in their EIS, which also contemplate internal procedures, system data structures, enterprise organizational structure, etc. (second layer). The more complete these models are the more enterprises could benefit from good management decisions, along with operational traceability and quick identification of possible problems. EIS software (third layer) implements the enterprise information models, executes the enterprise services, manages information exchange, this is fundamental to allow the collaboration (inside and outside the enterprise) and create a good internal interoperability environment. As explained previously, every time that it is necessary to put different EIS (or devices) working together, some "granularity" between them is expected. To

[1] http://www.merriam-webster.com/dictionary/device

mitigate this situation, it is necessary to create interoperability solutions. At this point, the enterprises have reached the Enterprise Interoperability level, (the fourth). To continue working smoothly, the models must become static, since every time a change is made, an interoperability harmonization breaking in the system occurs. As that does not reflect the reality, a fifth layer is required where the network needs to be sustainable, allowing each enterprise to make modifications in their EIS, models, and devices. With these measures, a self-sustainable interoperable system is created, and promoting this is the motivation for this dissertation.

1.1. Research Framework and Motivation

The goal of this dissertation is to contribute to the creation of a sustainable interoperable environment in enterprise collaboration, creating a system that is able to maintain the interoperability status of the network every time that a problem occurs. Figure 1-2 shows an example of an EC, which is composed of three enterprises that cooperate among each other, in a sustainable interoperability environment. Throughout the life cycle of this collaboration, enterprises exchange information between them to achieve the common goal they have defined. Since the models and devices are different between enterprises, they had to create mappings for the exchange of information and thus create an interoperable environment on the network. Since an EC is not a static network, several situations can occur to disrupt the harmonization of the network. Such situations arise when an enterprise makes changes in its procedures, models, EIS, devices, or even when there is a new entry in the network. For these situations not to create inefficiencies, it is necessary for the system to be prepared to constantly receive feedback about its neighbors, propose reaction and adapt in real-time (Agostinho, 2012).

When a sustainable interoperable system is achieved it becomes a system that allows an enterprise to enter and leave the network without causing problems to the network, and it will allow changes in the models. The task is divided into several steps that facilitate the goal, e.g. allow the system to evaluate the level of interoperability in the organizations, (in some cases interoperability may not exist). This evaluation is important to allow the system to react in three different ways:

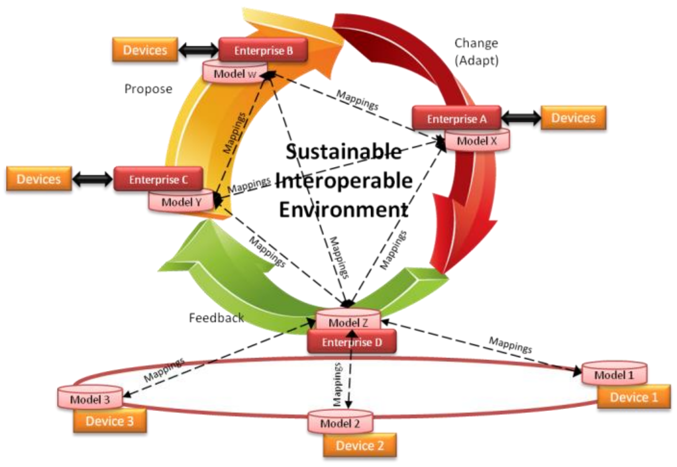


Figure 1-2: Sustainable Interoperable Environment.

1. Give support to maintain the interoperability in the network;
2. Improve the interoperability of the network, achieving a better level (reaching a better interoperability between enterprises);
3. In the case that interoperability does not exist, the system will provide measures to support the enterprises to reach that goal.

The motivation of this dissertation is to allow the creation of dynamic EC without having interoperability problems, making it possible for an enterprise to enter, leave, and/or make changes in their models without creating problems for the network partners. With this the enterprises will spend less, at the production line or by searching for a possible solution, resulting in that the financial loss is less and do not create so big impact to the enterprises. Another advantage of this system is to allow the monitoring of the network, by searching for changes that will create a harmonization breaking, or identify the problems by predicting them. This is important since it will identify mistakes before they occur, by solving them, thereby saving time and money, which are critical points for enterprises. The aim of this work is to create a self-sustainable interoperable environment that was proposed by Agostinho in (Agostinho, 2012) and it will be a continuation of his work by validating the concept, this will be made by creating a framework with the aim of reaching his ideal.

1.2. Research Method

The research method used in this dissertation is based on the classical research method (Camarinha-Matos, 2010) and comprises seven steps. The method begins with the search for the problem and ends with the interpretation of the results. If for some reason the results are unsatisfactory, this method allows one to return to the first step and experiment with a new approach. Each step of the method is described below:

1. **Research Question/Problem** - This is the most important step in research because it is here that the area of interest is defined. It is possible that several secondary questions appear in order to help define the main idea. These questions are located in Chapter 1.3.2;
2. **Background/Observation** - This is the section to do the research for the state of the art, by studying similar earlier works, presenting the literature review, and earlier projects with the goal of helping to start the dissertation. This is important because earlier works will have an impact on the new one, and help in a new approach. Revealing the existing ideas of other authors will, therefore, create and open the way to new solutions to be used in the development for this dissertation. This background is located in Chapter 2 and Chapter 3;
3. **Formulate Hypothesis** - Here is where the expected results of the research are discussed. It is important that the hypothesis is simple to understand, specific, conceptually clear, and measured. The hypothesis of this dissertation is in Chapter 1.3.3;
4. **Experiment Design** - This step reveals the detailed plan and steps of the experimental phase, which often includes the design of a prototype or even system architecture. The system architecture and methodology are presented in Chapter 4, Chapter 5, Chapter 6 and Chapter 7;

5. **Test Hypothesis/Collect Data** - In this phase the evaluation of the system architecture is made. This is done by testing and simulating different scenarios. In each test data for further analysis and hypothesis, validation is collected. The implementation scenario is described in Chapter 8.3 with the description of the industrial validation;
6. **Interpret/Analyze Results** - Here is where the results are analyzed and the veracity of the hypothesis is determined. If for some reason the results are not satisfactory, it is possible to change to a different approach and return to step 1. Moreover, when positive results are achieved, it is possible to look at the next steps, giving ideas for further research. Throughout this dissertation in each chapter an analysis of the chapter is made before what one intends to achieve in the hypothesis;
7. **Publish Findings & Transfer to Industry** - When good results are achieved and a good contribution to the scientific community is made, it is important to share the results with the community and in some cases transfer the technology to industry. Presenting the results to the scientific community is done using different means, such as a scientific paper, conference addresses, journals, and so on. In the case of the industry, it is very important to transmit the results in such a way that they can be implemented. The contribution to the scientific community is described in Chapter 8.1 and the contribution to the industry is described in Chapter 8.2.

1.3. Research Problem and Hypothesis

In this chapter, the research problems of this work are presented. Then the research questions are evaluated and presented, finishing the hypothesis and the approach that will be employed.

1.3.1. Problem Introduction

This doctoral work develops a novel framework to contribute to the sustainability of systems interoperability in a network of enterprises composed of models that represent their information, resources, and devices. In a Collaborative Network each partner needs to cooperate with the group, and as a result needs to be integrated into the conceptual, technical, and application levels and at the business level as well. In a first approach, three main issues arise that need to be solved:

1. **How to reach full integration between two or more enterprises?** This is a difficult task to achieve, as it is complicated to reach integration on all the three levels. Enterprises are usually able to achieve good interoperability on one or two levels but have challenges with the others. For example, an enterprise can reach a good technical interoperability but the conceptual interoperability has several problems;
2. **How to create interoperability between the different devices that enterprises use in their systems?** In an era of devices, where we can find devices at each corner a person go, it starts to be very important to enterprises to take advantage of these devices. However, their systems are not ready to use/control the devices, since each device have

their own standard and model. To allow them to cooperate, it is important to create an interoperability environment in this network;

3. **How to maintain the interoperability?** Markets are not static. They change due to customers' demands, forcing enterprises to follow them, by making changes in their models (of systems and devices), causing harmonization breakings occur in the system.

A way to avoid these issues is to give the system the capability of maintaining the interoperability of the network.

1.3.2. Research Questions

Based on the problems mentioned above, the following research questions are formulated:

Question:

Is it possible to make existing interoperable networks sustainable over time, responding spontaneously to harmonization breaking situations?

As the main research question is very extensive, it was necessary to divide it into four sub-questions to support to answer it. The four sub-questions are:

1. **What characterizes a sustainable interoperability network?**
2. **How can one identify and measure systems interoperability?**
3. **How will it be possible to create/achieve sustainable interoperability in a network?**
4. **How will the system react to poor integration?**

1.3.3. Hypothesis and Approach to Follow

Taking these Research Questions into consideration and in response to the analysis made with the Literature Review in Chapters 2 and Chapter 3 the following Hypothesis is made:

If a framework is able to assess the unity of a network (to verify how the interoperability is implemented), monitor the communications (with the aim of finding changes in the system), and be able to propose solutions to recover the interoperability, then the framework will be able to create a sustainable interoperability system in the network.

1.4. Industrial Supporting Framework

In order to validate the viability of the framework proposed during this work, it was necessary to implement it, having been divided by several validations, between academic and industrial validations. This dissertation was developed integrated into the Group for Research in Interoperability of Systems (GRIS) at UNINOVA and in the CRESCENDO[2], ENSEMBLE[3], IMAGINE[4], OSMOSE[5] and

[2] http://cordis.europa.eu/project/rcn/93958_en.html
[3] http://cordis.europa.eu/project/rcn/95531_en.html
[4] http://cordis.europa.eu/project/rcn/99927_en.html
[5] http://cordis.europa.eu/project/rcn/189013_en.html

C2NET[6] EC 7FP and H2020 Projects. This chapter describes the projects that were involved in this work. Given that in order to be able to describe the demonstrators it is necessary to explain the projects in order to be able to understand how the validations work. Thus, five projects are described.

1.4.1. IMAGINE Project

Nowadays, manufacturer's facilities are distributed all over the world, by creating a manufacturer network. This requires the ability to change various times the processes if needed during a product life-cycle, creating the need for dynamism, in order to adapt their models as faster as possible, to not lose time and money (EFFRA-UE, 2013). By having a framework that facilitates the integration between different enterprises, facilitating the production in a common and sharing product manufacturing requires an advanced and automatic search of partners for a Dynamic Manufacturer Network (DMN) creation and maintenance process. Four complementary stages compose the implementation of a DMN lifecycle, as illustrated in Figure 1-3 (Papazoglou et al., 2015), (Panopoulos et al., 2013):

1) The initial stage relates to the onboarding and administration procedures, and is where new partners or enterprises data is inserted in the Blueprints (BP) (the BP is described in detail in (Papazoglou et al., 2015)) (predefinition and implementation of data mappings are required), and made available to the DMN;
2) Network Analysis and Configuration represents the phase when the virtual production network is formed;
3) Network Design creates the layout of the DMN as the basis for the design of the detailed end-to-end manufacturing process;
4) Network Monitoring and Governance, which is the stage responsible for the deployment, monitoring, and governance of end-to-end processes in the DMN.

The IMAGINE project implemented a platform to embody the described DMN. Thus, the platform needs access to some specific resources' information of each enterprise to support the partners' choice through the best options depending on several criteria previously uploaded in the knowledge base (KB). Even though the upload of such information can be done manually through a web portal, the automatic data exchange among enterprises and the platform is relevant, and it can be achieved through a service-based customization of the platform capable of ensuring interoperability at technical and semantic level (this is made through the use of Web Services (WS) to get the information data from the enterprises and inserting it into the repository of the platform, by identifying if the data is correct and performing a semantic alignment in regard to the concepts). This approach will enrich the manufacturer's representation in the proposed platform, making them more responsive and agile in designing and producing a new generation of future-internet manufacturing orders, thereby boosting their productivity and competitiveness. Also, it will provide the core functionality and lay the foundations for the Factory of the Future (FoF) by tightening plant-to-enterprise integration at network level towards end-to-end manufacturing. To facilitate such end-to-end manufacturing, it proposed a

[6] http://cordis.europa.eu/project/rcn/193440_en.html

system adapter, which represents the mentioned platform extension and that will be responsible for the connectivity between legacy EIS and the generic platform itself.

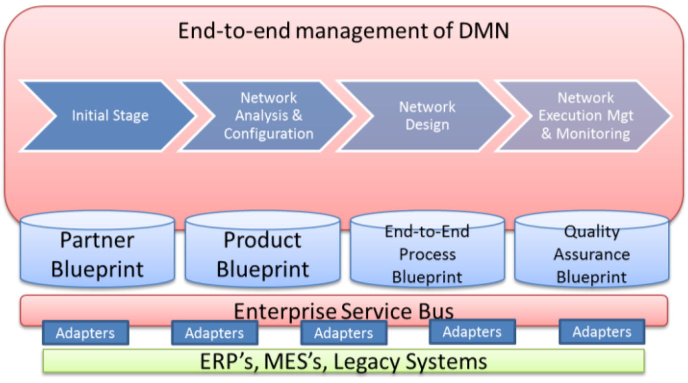

End-to-end management of DMN
Initial Stage
Network Analysis & Configuration
Network Design
Network Execution Mgt & Monitoring
Partner Blueprint
Product Blueprint
End-to-End Process Blueprint
Quality Assurance Blueprint
Enterprise Service Bus
Adapters
Adapters
Adapters
Adapters
Adapters
ERP's, MES's, Legacy Systems

Figure 1-3: FoF Integration Framework (Panopoulos et al., 2013).

The contribution that this project can bring to this work is related to the development of the adapter since this dissertation intends to support the management of the interoperability in a network of enterprises. One of the necessary points to achieve is to start in the register of enterprises, to be able to register an enterprise begins by knowing what type of data, devices, and services use, to allow the system to control it.

1.4.2. C2NET Project

The goal of the project is the creation of cloud-enabled tools for supporting the SMEs supply network optimization of manufacturing and logistic assets based on collaborative demand, production and delivery plans (Figure 1-4). The main objective of the project is to build a new architecture in the cloud to provide SMEs, affordable tools (in terms of cost and ease of use) to help overcome the current economic crisis, improving competitiveness in the economy world. At the same time, to create cloud tools to support optimization of manufacturing networks composed mainly of SMEs and their logistic assets through demand management, production, and supply plans, considering the Collaborative Network perspective. In the project is expected to generate a Cloud Architecture composed by (Andres et al., 2016):

- **The Data Collection Framework** – To provide software components and hardware devices for IoT-based continuous data collection from supply network resources. This supports the collaborative manufacturing functionality while taking advantage of Cloud environments, which can enable solutions that are highly scalable, available and fault-tolerant;
- **The Optimizer –** To support manufacturing networks in the optimization of manufacturing and logistics assets by the collaborative computation of production plans, replenishment plans, and delivery plans in order to achieve shorter delivery times, better speed and consistency of schedules, higher use of productive resources and energy savings;
- **The Collaboration Tools** – For providing the Collaborative Manufacturing Network Platform with a set of tools in charge of managing the agility of the collaborative processes;

- **The Cloud Platform** – To integrate the data module, the optimizers, and the collaborative tools in the cloud and allow the access to process optimization resources to all the participants in the value chain to support their decisions and process enhancement. It will provide the base for the integration of the different modules for generating a collaborative working environment for manufacturing network partners.

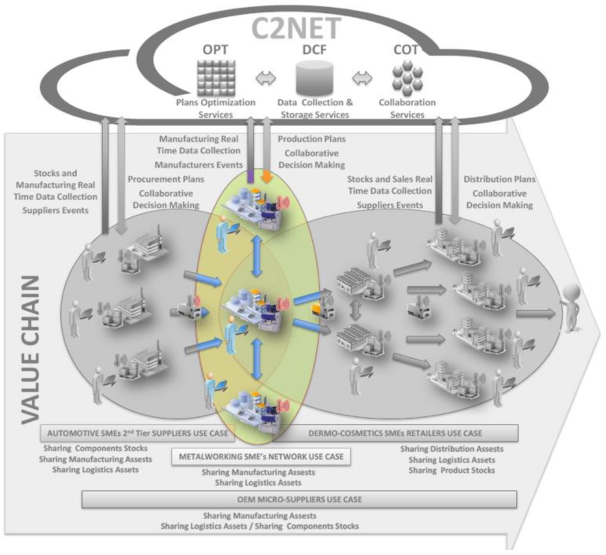

C2NET
OPT
DCF
COT
Plans Optimization Services
Data Collection & Storage Services
Collaboration Services
Stocks and Manufacturing Real Time Data Collection Suppliers Events
Manufacturing Real Time Data Collection Manufacturers Events
Procurement Plans Collaborative Decision Making
Production Plans Collaborative Decision Making
Stocks and Sales Real Time Data Collection Suppliers Events
Distribution Plans Collaborative Decision Making
VALUE CHAIN
AUTOMOTIVE SMEs 2nd Tier SUPPLIERS USE CASE
Sharing Components Stocks
Sharing Manufacturing Assests
Sharing Logistics Assets
METALWORKING SME's NETWORK USE CASE
Sharing Manufacturing Assests
Sharing Logistics Assets
DERMO-COSMETICS SMEs RETAILERS USE CASE
Sharing Distribution Assests
Sharing Logistics Assets
Sharing Product Stocks
OEM MICRO-SUPPLIERS USE CASE
Sharing Manufacturing Assests
Sharing Logistics Assets / Sharing Components Stocks

Figure 1-4: C2NET Objectives.

The contribution that this project can bring to this work is related to the registration of devices and how to manage them. This management involves data collection, fault monitoring, and fault tolerance. These are points that are interesting to maintain and to recover the interoperability of the network, whenever necessary.

1.4.3. OSMOSE Project

The OSMOsis applications for the Sensing Enterprise – OSMOSE project aimed at developing a reference architecture, a middleware and some prototypal applications for the Sensing-Liquid Enterprise, by interconnecting Real, Digital, and Virtual Worlds in the same way as a semi-permeable membrane permits the flow of liquid particles through itself (Agostinho et al., 2015). The worlds represent a way of organizing the structure of an entire manufacturing enterprise, and the business applications in three types of data management environments: **Real World (RW)** - related to data that comes directly from devices that is handled by physical components; **Digital World (DW)** - related to data management available in data and knowledge bases or Internet (big data); and **Virtual World (VW)** - related to specific management of data with the support of future projections or specific simulations (Spirito et al., 2014).

Following the LSE paradigm, osmosis processes are a special type of business processes used to moderate the information exchanged among the worlds. The six Osmosis processes considered are detailed in (Marques-Lucena et al., 2015):

- **Digitalization (RW-DW)** – Model and represent RW data in a computer-tractable form;

- **Actuation (DW-RW)** – Plan and implement highly distributed decision-making;
- **Enrichment (VW-DW)** – Extends the computational capabilities of the DW with annotations and projections coming from simulations and what-if hypothetical scenarios;
- **Simulation (DW-VW)** – Instantiate and run hypothetical VW scenarios based on historical data;
- **Virtualization (RW-VW)** – Provides real-time data for simulation of hypothetical simulations;
- **Augmentation (VW-RW)** – Annotates Real World objects with Virtual World information.

Model-Driven Paradigm for the LSE

A business process can be seen as a set of internal activities performed to serve a customer (Jacobson et al., 1994). It is characterized by being: a purposed activity; carried out collaboratively by a group; it often crosses functional boundaries; it is invariably driven by outside agents or customers (Ould & Ould, 1995). This means that to accomplish a business process, especially in manufacturing, it is necessary to involve several partners or user profiles and manage knowledge across different boundaries of the enterprise (Zdravkovic et al., 2013), much like the LSE.

To better align the implementation and support of a process lifecycle, a separation of concerns starting from business goals down to the consequent physical means to realize it is required (Yves Ducq et al., 2012). It can be accomplished if a model-driven approach is applied. Thus, instead of writing the code directly, such approach enables services to be firstly modeled with a high level of abstraction in a context-independent way. The main advantages of applying model-driven approaches are the improvement of the portability, interoperability, and reusability through the architectural separation of concerns (Grangel et al., 2008).

The work presented in this work was inspired by the one presented in (Yves Ducq et al., 2012), which adapted the model-driven concept to manufacturing services development, with the definition of Model-Driven Service Engineering Architecture (MDSEA) concept. It followed the Model Driven Architecture (MDA) and Model Driven Interoperability (MDI) principles (Lemrabet et al., 2010), supporting the modeling stage and guiding the transformation from the business requirements (Business Service Model, BSM) into detailed specification of components that need to be implemented (Technology Specific Models, TSM). This approach proposes that each model, retrieved by the model transformation from an upper-level model, should use a dedicated service modeling language, which represents the system containing the level of description needed. MDSEA was the chosen method because is already oriented to the development of services for business processes and identifies the concepts IT, Physical Mean and Human used to describe the processes.

However, for such approach to be successfully applied to the LSE concept, it should be enriched with the capability of representing concerns related with the LSE-enabled real, digital and virtual worlds of the Liquid-Sensing Enterprise (Agostinho & Jardim-Goncalves, 2015). Following this requirement, three levels of abstraction where adapted from the MDSEA (see Figure 1-5):

- **Osmosis Business Models (OBM)**, where the business case is defined. OBM extends the BSM in the sense that this abstraction level envisages meta-information not only about components (e.g. actors, resources, etc.) but also about activities and the world in which it is active (e.g. "scheduled maintenance" is an activity from the DW and "clean machine" is from the RW). The representation of the world in each activity is called OBM Annotated, enabling the system to identify osmotic processes;
- **OSMOSE Technology Independent Models (OTIM)**, which like the MDSEA TIM is complementing the upper-level model with detailed technology independent functionally. OTIM is optimized for the osmosis processes representation, detailing such behavior (OSMOSE membrane) and the interactions between the source and target world. For instance, in a digitalization process, OTIM represents three pools of activities (one the RW, one for the DW and one for the membrane);
- **OSMOSE Technology Specific Models (OTSM)**, which is the last level and consists in the instantiation and parameterization of the identified activities with services needed for the process execution.

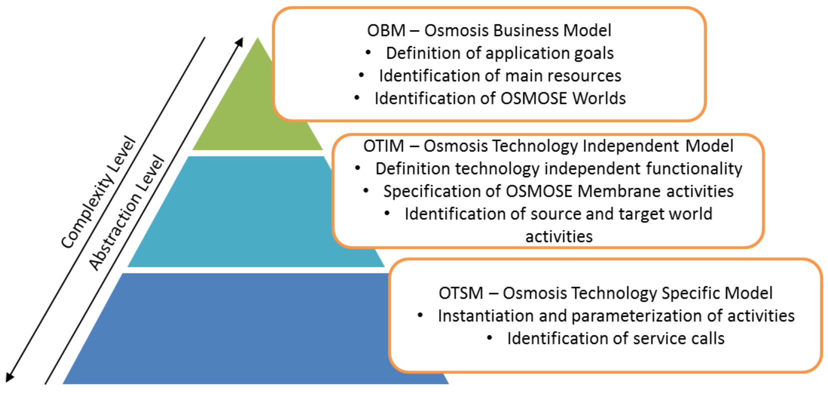

OBM – Osmosis Business Model
• Definition of application goals
• Identification of main resources
• Identification of OSMOSE Worlds
OTIM – Osmosis Technology Independent Model
• Definition technology independent functionality
• Specification of OSMOSE Membrane activities
• Identification of source and target world activities
OTSM – Osmosis Technology Specific Model
• Instantiation and parameterization of activities
• Identification of service calls
Complexity Level
Abstraction Level

Figure 1-5: OSMOSE Process Design Methodology.

The contribution that this project can bring to this work is related to the notion of sensing enterprise, allowing the enterprise to be aware of it surrounding enabling to follow with more detail the production. Gaining in this way the ability to identify complicated situations and to react when necessary, being able to maintain interoperability internally, and at the same time not compromising the production.

1.4.4. ENSEMBLE Project

The ENSEMBLE (Envisioning, Supporting and Promoting Future Internet Enterprise Systems Research through Scientific Collaboration) collaboration project, aims to coordinate and promote research activities in the domain of FInES and to systematically establish EI as a science. ENSEMBLE combines systemic approaches, scientific multi-disciplinarity, and innovative Web 2.0 collaboration tools with a community-driven mentality, in order to significantly increase the impact of the Future Internet Systems.

The project is not industry-driven, nevertheless, it is validating many of the work and concepts proposed, by incorporating them in the EI science base (EISB) problem and solution spaces formulation. The contribution that this project can bring to this work is related to the use of the EISB problem and solution spaces formulation to support evaluate the interoperability in the enterprises through an assessment framework.

1.4.5. CRESCENDO Project

The customers each day demand for better and more complex products, forcing the market expectations demanding a more efficient aircraft behavior, and in result forcing the enterprises to develop the products in a shorter time and with a more cost-effectiveness (CRESCENDO 2009). These force the enterprises to be global and in answer to this, the aircraft manufacturer & suppliers need collaboration solutions to work together in multi-disciplinary teams across the extended enterprise. These collaboration solutions have to follow all the development of the PLC, with good management to avoid rework and loss of time and money. One solution to avoid to do some rework is making often an analysis of the system to eliminate risks during all the phases and being able to accurately predict functional behavior. Another thing reduces the time consumption of time, and in consequence of money, is to simulate the products before it goes to the market, to avoid problems, see Figure 1-6. These are the challenges faced by the aeronautical industry and CRESCENDO proposes to solve this situation with the Behavioral Digital Aircraft.

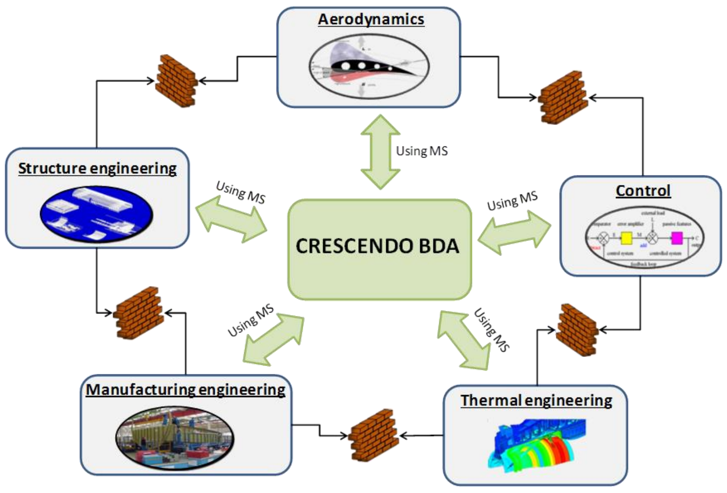


Figure 1-6: Current barriers and envisaged CRESCENDO solution.

Among the several components of the Behavioral Digital Aircraft, the Model Store is the core and will be the result that satisfies the CRESCENDO objective to organize modeling and simulation for multidisciplinary purposes. It proposes a common architecture of models that can be configured to provide the complete behavioral and functional view of the product and will be an enabler to organize and share all models in the Extended Enterprise. The Model Store will deliver innovation in terms of multi-level, ontology-based model architectures, supported by a semantic technology-based modeling and simulation dictionary, together with adapted modeling languages, management and control processes. These results will allow models associativity, evolution and context to be managed more effectively throughout the lifecycle. More specifically, the Model Store results will illustrate (CRESCENDO, 2010):

- How to link models developed for different purposes (or by different organizations), in a secure, federated, retrieval and storage system, so that the combined models can be used to simulate larger applications;
- How to link models in a hierarchical manner to provide consistent views of a product at differing levels of abstraction or life cycle stages;
- How to allow a model to understand its own behavior, input needs, and output capabilities, such that when a new element is added to the system, it will negotiate with the models of all the other elements of the system and fit in the overall common model's architecture.

The contribution that this project can bring to this work is related to the identification of a network in a collaboration, the notion that it must work together not only in production but in identifying solutions to avoid delays in production, one way is to predict functional events and communicate with them in order to avoid it, with this measure implemented in this work, can support to maintain the interoperability in the network through the network collaboration.

1.5. Dissertation Plan Outline

The problem to address has now been presented as has the purpose of this dissertation. Several topics leading to the hypothesis have been identified, for the current work, thus the remainder of the dissertation is organized into eight chapters:

Future Internet Systems Interoperability and **Interoperability Monitoring and Assessment** are state of the art issues and reside at the core of this work, representing the chapter two and three. In chapter four the **Framework to Enable IoT in a Sustainable Interoperable Environment** is presented, is the contribution of this dissertation. The chapters five, six and seven are chapters that describe the framework presented in this dissertation, then the chapter eight the **Implementation Testing and Hypothesis Validation** is made. The last chapter is the **Final Considerations and Future Work**.

Future Internet Systems Interoperability, the second chapter, begins by explaining how enterprise information systems work in a collaborative network, presenting problems and advantages in use, which will be important to reach the interoperability status. The aim is to alert the reader to the importance of system interoperability, and more importantly, the creation of a sustainable interoperability network. Throughout the chapter, various topics are covered that are important in the development of this work.

The third chapter is about **Interoperability Monitoring and Assessment**. Here will be addressed the topics about monitoring the interoperability of a network while performing an assessment. This chapter will help to answer one of the research questions since it will lead to the identification and measuring of the interoperability of a network. It will focus on the interoperability maturity methodologies, which are important in defining the level of interoperability in the enterprise. Next will be identified several assessment methods to allow the evaluation of the interoperability inside the enterprise (e.g. devices), giving the framework a critical capacity.

The fourth chapter is about the **Framework to Enable IoT in a Sustainable Interoperable Environment** is where the description of the contribution of this dissertation is made, being focused in how the framework works to solve the problem presented in this dissertation, and how it is going to answer to the research questions, by presenting the framework. To support the description of the framework, three chapters have been created that represent parts of it, which are the chapters five, six and seven. Chapter five describes **Dynamic Network Manager**, which is responsible for configuring enterprises. Being responsible for preparing the enterprise to enter the network and prepare for collaboration. Chapters six and seven describes the IoT Framework, however, they represent two different phases of this block, the design mode, and the runtime mode. In each of these two chapters, they presented the modules, describing their methodology and their components. In chapter six, the **IoT Framework – Design Mode** is described is responsible for identifying how each enterprise is represented to identify points in common with each other, allowing the creation of a bridge to connect them. In chapter seven, the **IoT Framework – Runtime Mode** is described, in this mode, it is where the network is working, with enterprises communicating with each other and running. As unforeseen events can occur, the network is constantly being monitored, so that the framework can react at the time and thus avoid larger problems. Explaining in this way how the sustainability of the interoperability is processed in each phase of the framework, aiming to be able to demonstrate to the reader the explanation of each phase of the framework until achieving the interoperability.

The eighth chapter is about the **Implementation Testing and Hypothesis Validation**, as the name implies, this chapter describes the implementations developed to demonstrate the various phases of the framework and to support its validity. This chapter ends with the validation of the hypothesis, where the hypothesis is discussed and at the same time answered, describing how it was validated. The last chapter is the **Final Considerations and Future Work** where is done a retrospective of the work developed throughout this dissertation, what was achieved, and what contributions it has brought to solve the problem that was being described during this work. Ending with the explanation of future work, which describes the work that is missing, and which is intended to continue to develop.

2. FUTURE INTERNET SYSTEMS INTEROPERABILITY

Enterprise Information Systems (EIS), as defined by the US Department of Energy (US Departement of Energy, 2011), are large-scale, integrated application-software packages that use the computational, data storage, and data transmission power of modern Information Technology (IT) to support processes, information flows, reporting, and data analytics within and between complex organizations. The integrated content may then be used to provide a configuration management solution throughout the life cycle in relation to the products, devices, assets, processes, and requirements of the entity (laboratory, facility, SDD, etc.). Commercial software packages promise the seamless integration of all the information flowing throughout the enterprise. The problem comes when there is the need to communicate with concurrent systems from different providers. If an organization has two systems and they cannot communicate with each other, then this will bring some problems for instance at the level of manufacturing, and customer responsiveness suffers as a result (Davenport, 1998).

Another point is related to the repositories in use by the different enterprises, which are used for collecting, generating, and storing massive amounts of data. Systems normally evolve along with the needs of the enterprises. Every time that a repository is needed, one is created, leading to a situation in which each enterprise has various repositories. Consequently, the information starts to spread across several different computer systems or devices, each housed in an individual function, business unit, region, factory, or office. These systems provide support to the activities for which they were created, but they represent one of the heaviest drags on business productivity and performance now in existence. EIS arise seeking to structure the systems and address this type of situations. Due to their modular nature, enterprises may install only the more appropriate modules. However, the system's complexity makes major modifications impracticable.

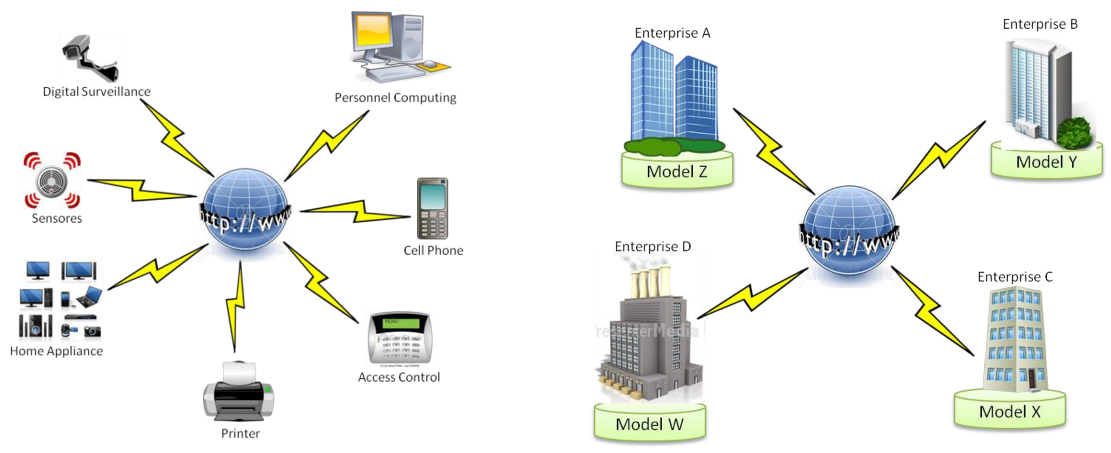


Figure 2-1: Examples of an Interoperability Network: a) Physical Processes Network; b) Collaborative Network.

Moreover, when working in a network, EIS problems are compounded even further, since there are hurdles on the path leading toward the integration of the enterprises. Organizations need a system or a product to work with other systems or products without special effort on the part of the customer. This is what IEEE (Geraci et al. 1991) defines as interoperability. Figure 2-1 presents some examples of an interoperability network. In a) a network composed of physical components is represented in which each component is communicating with the others. This follows the principle of the Cyber-Physical Systems (CPS), which represents the integration of computation, networking, and physical processes. The network monitors and controls the physical processes, receiving feedback from them and basing responses on that feedback. This allows for awareness of the state of each physical process (Asare et al., 2012). b) Illustrates an EC communicating via the web with the aim of producing a specific product. Each enterprise produces a part of the product, which is then assembled at Enterprise A. For this to run smoothly the systems need to be interoperable in order to avoid delays in the production. At the outset of the EC, an agreement is made to decide how the communications will be and how to share the information needed.

The lack of interoperability can cause great damage, modern enterprises have to be interoperable in terms of not only their IT systems, but also their business processes, their applications, and even their human and resources. Interoperability is a concern between organizational units or business processes within a large enterprise as well as within an enterprise network. The challenge lies in facilitating communication, cooperation, and coordination among these units and processes (Kotze & Neaga, 2010; Ray & Jones, 2006). In conclusion, from an enterprise architecture and a systems engineering perspective, operating in a networked environment places the requirements for interoperability alongside maintainability, reliability, the safety of a system (Kotze & Neaga, 2010).

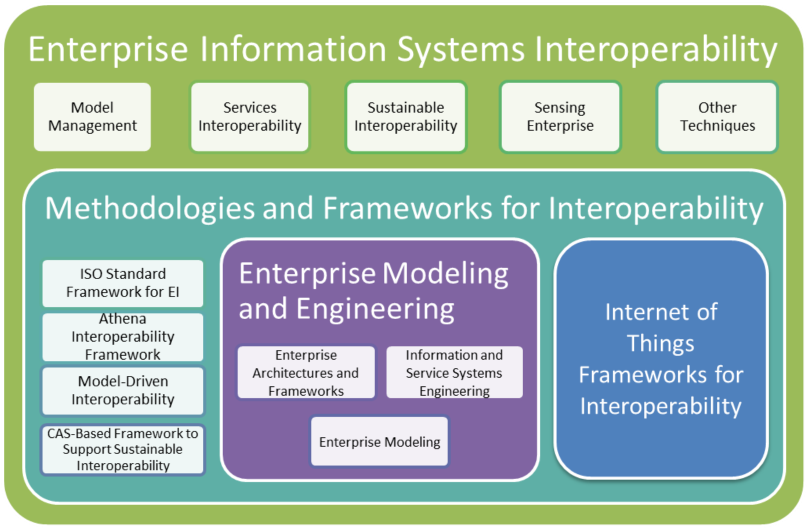

Enterprise Information Systems Interoperability
Model Management
Services Interoperability
Sustainable Interoperability
Sensing Enterprise
Other Techniques
Methodologies and Frameworks for Interoperability
ISO Standard Framework for EI
Athena Interoperability Framework
Model-Driven Interoperability
CAS-Based Framework to Support Sustainable Interoperability
Enterprise Modeling and Engineering
Enterprise Architectures and Frameworks
Information and Service Systems Engineering
Enterprise Modeling
Internet of Things Frameworks for Interoperability

Figure 2-2: Enterprise Information Systems Interoperability.

To support the structure of this chapter, a diagram was made with topics that will be discussed starting from the outside, there are several techniques to facilitate EIS interoperability (Figure 2-2). The topic self-sustainable interoperability and sensing enterprise are presented, explaining the advantages they bring to organizations. In order to obtain good interoperability level, the methodologies and frameworks for interoperability are presented, as the ISO Standard Framework for

EI, or the Athena Interoperability Framework. In the center are some Enterprise Modeling Techniques and Engineering Frameworks that are popular among enterprises that will be discussed. Also shown is the CAS-based framework to support the sustainable interoperability that will be adopted as the inspiration for this dissertation and the development of the framework to be presented in this work.

2.1. Enterprise Modeling and Engineering

Vernadat (2001) defines Enterprise Modeling (EM) as the efficient design, analysis, and optimization of enterprise operations requiring notations, formalisms, methods, and tools to depict the various facets of a business organization. An important point of using an EM is that it will help to describe the elements of the enterprise, including its functions, behavior, information, resource, organization or economic aspects of a business resource. This chapter presents and analyses several architectures and frameworks used for EM.

2.1.1. Enterprise Modeling Techniques

Several techniques for EM appear in the literature. The study of methods such as Integrated Enterprise Modeling (IEM), CIMOSA, and GRAI-GIM is important for this work since they allow the identification of how the enterprises are structured, supporting the framework to create patterns to create relations between the different models used by enterprises. Other frameworks exist, however, and are not necessarily less important than the ones discussed here.

IEM seeks to support the development of a unified model and to represent the different aspects of a manufacturing enterprise as views of the unified model (Kamath et al., 2003). Employing the object-oriented approach to describe information and functions of objects as views on a single model of the systems enterprise, the IEM creates two views of the system (Mertins & Jochem, 2004): **Information Model** (all processes and activities that are related to the production are described by functions and business processes as objects) and **Business Process Model** (tasks and business processes are executed on the objects).

Although being modeling techniques, the IEM and GRAI-GIM have different focuses. The first was developed to support processes (focused on describing the product with an object-oriented focus), while the second is a modeling technique to support the design of computer integrated manufacturing (CIM) systems, which means that is more focused on the manufacturing area, to support the specification of CIM systems. The GRAI-GIM was developed to support the modeling of a production management system in order to allow defining the specifications needed to select a software package for the entire manufacturing system and to support the design of such CIM systems. This model design has the following sub-systems (Reid & Preez, 1998): **Physical System** (Contains all components of a CIM system such as machines, workers, and techniques that are involved in the transformation of material and flow), **Decision System** (This system represents the decision-making hierarchy within a CIM system), **Information System** (This system is the connection between the physical and the decisional system, and with these systems and their environment) and Functional View (gives the possibility to address these 3 systems as 3 views).

Finally, CIMOSA is as the GRAI-GIM, a modeling technique to support the design of CIM systems. The major difference relies on the fact that CIMOSA describes physical attributes, instead of functional attributes of physical elements. Also, it probably better tool-supported at the industrial level than the other methods. CIMOSA (Computer Integrated Manufacturing Open System Architecture) is an open system architecture for CIM that defines a set of concepts and rules to facilitate the building of future CIM systems. It focuses on the standardization of activities and is defined in the project's three main results (PERA, 2001): **Modeling Framework** (It supports all phases of the CIM system life-cycle from requirements definition, through design specification, implementation description, and execution of the daily enterprise operation), **Integrated Infrastructure** (It provides specific information technology services for the execution of the Particular Implementation Model), **Event-Driven** (That is a process-based modeling approach with the goal of covering essential enterprise aspects in one integrated model. The main aspects are the functional, behavioral, resource, information and organizational).

2.1.2. Enterprise Architecture and Frameworks

Enterprises have the need to internally organize their structures, aiming to be more efficient and achieve their objectives more quickly. For that reason, they search for solutions to guide them in the best direction. An example is to create an Enterprise Architecture (EA) as an answer for their needs. This is important for them since an architecture is a framework of principles, guidelines, standards, models, and strategies that direct the design, construction, and deployment of business processes, resources, and information technology throughout the enterprise (Technology Training Limited, 2013). An architecture thus appears to be a good solution to be implemented by the enterprises, as it creates a high-level view of the system desired, describing the current and future features of the enterprises. It is developed having in mind: a picture of the current state; a blueprint or vision for the future; a roadmap on how to get there. Some of the better-known EA frameworks are presented next.

Zachman Framework is a two-dimensional classification scheme for descriptive representations of an Enterprise. It was derived through observation of descriptive representations (design artifacts) of various physical objects like airplanes, buildings, ships, computers, etc., in which it was empirically observed that the design artifacts (the descriptive representations, the product descriptions, and/or the engineering documentation) of complex products can be classified by the audience for which the artifact was constructed (the Perspective) as well as by the content or subject focus of the artifact (the Abstraction) (Zachman, 2003). So far, the methods of the previous section were described to give support to enterprises specifying their own CIM systems and their processes. Zachman is a framework to give support to structure and define an enterprise, as a whole.

Another very well-known EA framework is the TOGAF (The Open Group Architecture Framework). Zachman and TOGAF are very well known in industry and scientific community, alike. However, Zachman is very focused on classifying the various architectural artifacts inside of an enterprise, unlike TOGAF, which is a more step-by-step process, and a better guideline for enterprises. TOGAF gives more relevance to reference models (e.g. standards) than Zachman.

Indeed, TOGAF is used with the aim of improving business efficiency, ensuring consistent standards, methods, and communication among enterprise architecture professionals. It helps practitioners avoid being locked into proprietary methods, utilize resources more efficiently and effectively, and realize a greater return on investment (The Open Group, 2011). Covering the development of four related types of architecture, i.e. **Business**, **Data**, **Application,** and **Technology Architecture**, TOGAF has an excellent support provided by Open Group with plenty information available and open tools.

DoDAF was developed with the aim of supporting the specification of requirements and control of the development of architectures to enable managers at all levels to make key decisions more effectively through organized information sharing across institutions. Unlike the others, DoDAF is a specific framework created to support a specific organization. Thus, it starts with the specification of the requirements to be used on the development of targeted and domain architectures. NAF (NATO Architecture Framework) is a similar framework extended by adding views for bandwidth analysis, SOA, and standard configurations (Alghamdi, 2009; Ota & Gerz, 2011). This extension was divided into seven groups of views, i.e. **All View**, **Operational View**, **System View**, **Technical View**, **Capability View**, **Service-Oriented View**, and **Program View**.

GERAM is a complementary paradigm that provides a set of recommendations which are baseline requirements to support enterprise architecture and engineering. This baseline is to support enterprises to assess the different architectures/methodologies known and choose the one that is best suited to their enterprise business needs. Hence, it was developed to encompass and generalize the commonalities of various existing EA frameworks and reference architectures, by including all knowledge needed for enterprise engineering/integration (Saha, 2004), (IFIP-IFAC Task Force, 1999). It, therefore, differs from the other architectures/methodologies, since it just gives support to choose one instead of developing new.

2.1.3. Information and Service Systems Engineering

In this section, several methods and methodologies to support the Information and Service Systems Engineering are presented in order to support the structuring of information and services of enterprises. This topic is important for enterprises, since each year the need increases to plan, develop, and manage the enhancements of their infrastructure, products, and services, including marketing strategies for product and service offerings based on new, unexplored, or unforeseen customer needs with clearly differentiated value propositions (Pineda et al., 2005). These issues make it important to have Service System Engineering inside enterprises.

It is also necessary to have computer-based infrastructures, organizations, personnel, and components that collect, process, store, transmit, display, disseminate, and act on information (Silberberg & Mitzel, 2005). These are features that are the responsibility of the Information Systems Engineering, based on these two topics, the importance for enterprises to implement these ideas in their systems increases with every passing year. It is essential to allow them to increase production by avoiding problems inside their infrastructures, and at the same time, give them greater ability to integrate a network without interoperability problems, since they are using standards.

The Model-Based Systems Engineering (MBSE) is the formalized application of modeling to support the systems engineering processes, namely requirements, design, analysis, verification, and validation activities beginning at the conceptual design phase and continuing throughout development and later Life-Cycle (LC) stages (Operations & Crisp II, 2007; INCOSE, 2011), starting from the concept where the stakeholder's needs are identified until the disposal of the product, then merged with the development process used in the MBSE, namely (Nallon, 2003): **Requirement Models** (Represents the relationships between user requirements and/or model objects), **Behavior Models** (Represents the intended and unintended behaviors for a system of interest (e.g. a product), thus responding to functional requirements), **Parametric Model** (Used to reply to the non-functional requirements representing the formal relationships and constraints of the system and its components) and Structure Model (Which describe the enterprise and system level contexts from both logical and physical views).

The OMG introduced the Model-Driven Architecture (MDA), which was created with the intention to support the MDE in the specification of software systems based on a model transformation concept, and with the ability to address the complete development lifecycle, covering analysis and design, programming, testing, component assembly, as well as deployment and maintenance. It was created with the aim of achieving three main goals; portability, interoperability, and reusability (ATHENA, 2010; Truyen, 2006; Petzmann et al., 2007). Models and the transformation of these models are the focus of the MDA approach, and they can be more accurately described as layers of abstraction, with three layers a set of models can be constructed, each corresponding to a more focused view of the system. Next is a description of three models (Figure 2-3) (Truyen, 2006; Petzmann et al., 2007; Miller & Mukerji, 2001; Miller & Mukerji, 2003): **Computation Independent Model** (CIM - It is where the requirements for a system are designed/modeled, and it helps in presenting exactly what the system is expected to do. It is useful not only as an aid to understanding a problem but also as a source of a shared vocabulary for use in other models), **Platform Independent Model** (PIM - It is where the view of a system from the platform independent viewpoint is designed/modeled. The goal is producing models that can be transformed into an arbitrary system platform), and **Platform Specific Model** (PSM - It is a view of a specific platform. The PSM merges the specification of the PSM with the details of a particular platform).

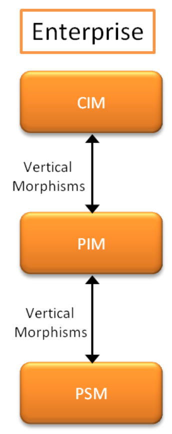


Figure 2-3: MDA's Conceptualization Levels and Transformations.

As an end goal, MDA software engineering approach must produce a code which reflects CIM business requirements and that can be run on specific platforms. This code should be achieved as automatic as possible.

Inspired by the web services and MDA/MDI, the Model Driven Service Engineering Architecture (MDSEA) was created to support the needs of modeling the three types of service system domain component (IT, Organization/Human, and Physical Means). The MDSEA was considered as an adaptation and extension of MDA/MDI to the engineering context of product-related services in the virtual enterprise environment. MDSEA proposes a framework for service systems modeling based on three abstraction levels (MSEE Project, 2012; Bazoun et al., 2013): **Business Service Model** (BSM - specifies the models at the global level, describing the service running inside a single enterprise or inside a set of enterprises as well as the links between these enterprises. The models at the BSM level must be independent of the future technologies that will be used for the various resources and must reflect the business perspective of the service system), **Technology Independent Model** (TIM - delivers models at the second level of abstraction independent from the technology used to implement the system. TIM levels represent the same system but with more detailed specifications. It gives detailed specifications of the structure and functionality of the service system which do not include technological details), and **Technology Specific Model** (TSM - enhances the specifications of the TIM model with details that specify how the implementation of the system uses a particular type of technology (such as IT applications, Machine technology, or a specific person)).

Another modeling paradigm based in service is the Service-Oriented Modeling Framework (SOMF), which is a model-driven engineering methodology whose discipline-specific modeling language and best practices focus on software design and distinct architecture activities employed during stages of software development lifecycle (Methodologies Corporation, 2011). SOMF was developed for any type of software development life cycle, starting at the conceptualization phase, supporting design and architecture activities, and extending modeling best practices for service operations in a production environment. To achieve these objectives, the SOMF has six distinct software development disciplines offering corresponding models whose language notation guides practitioners' design, architect, and support for a service ecosystem (Methodologies Corporation, 2011): Service-Oriented Conceptualization Model; Service-Oriented Discovery and Analysis Model; Service-Oriented Business Integration Model; Service-Oriented Logical Design Model; Service-Oriented Software Architecture Model; and Cloud Computing Toolbox Model.

Therefore, SOMF is a system engineering following the paradigms of SOA, which distinguishes it from the rest of the architectures/methodologies presented here. Being based on SOA means that is a framework oriented to software and web services.

2.1.4. Analysis of Different Techniques and Frameworks for EIS specifications and Modeling

The several methodologies/frameworks presented are different in nature. Some are more directed to information modeling of the enterprise, others to the information system, while others are more holistic and address the enterprise as a whole, treating requirements, product, process, and software in an integrated manner. All have pros and cons and, in the end, their adoption depends on the cost-benefit, as well as the learning curve for the tools implementing each paradigm.

Figure 2-4 depicts the coverage of each of the described architectures and methodologies in the operative levels of an EIS, i.e., Business, Process and Service (ISO, 2011a). It illustrates the models and concept used by each paradigm to manage each level of information. From this analysis, it is observed that all addressed architectures and methodologies tackles Process and Service level, whilst MDA, MDSEA, TOGAF, and SOMF additionally embrace the Business level with the high level

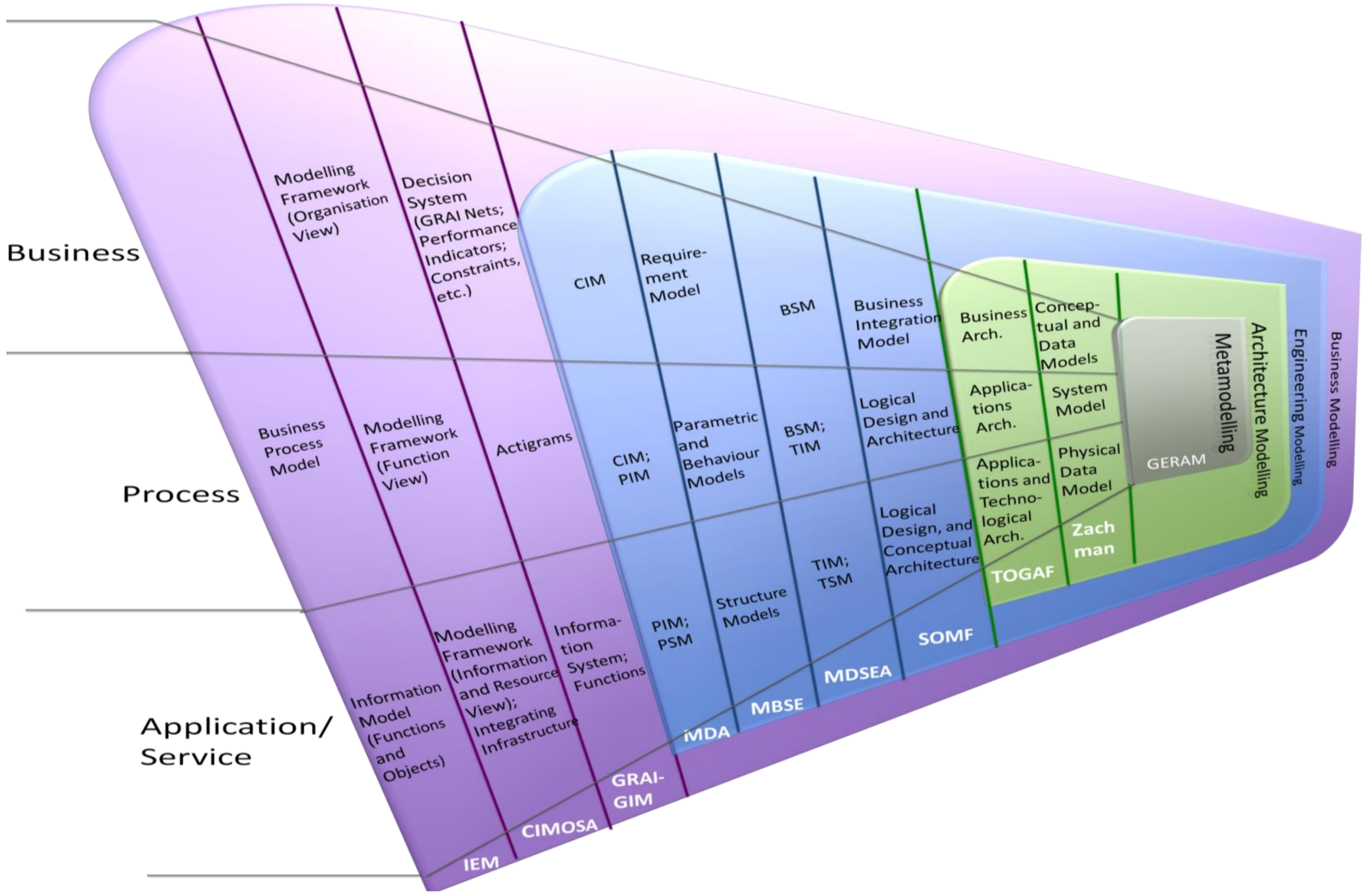

Business
Process
Application/ Service
Modelling Framework (Organisation View)
Decision System (GRAI Nets; Performance Indicators; Constraints, etc.)
Business Process Model
Modelling Framework (Function View)
Actigrams
Information Model (Functions and Objects)
Modelling Framework (Information and Resource View); Integrating Infrastructure
Information System; Functions
IEM
CIMOSA
GRAI-GIM
CIM
Require-ment Model
BSM
Business Integration Model
CIM; PIM
Parametric and Behaviour Models
BSM; TIM
Logical Design and Architecture
PIM; PSM
Structure Models
TIM; TSM
Logical Design, and Conceptual Architecture
MDA
MBSE
MDSEA
SOMF
Business Arch.
Concep-tual and Data Models
Applica-tions Arch.
System Model
Applica-tions and Techno-logical Arch.
Physical Data Model
TOGAF
Zach man
GERAM
Metamodelling
Architecture Modelling
Engineering Modelling
Business Modelling

Figure 2-4: Coverage of Modeling Architectures and Methodologies.

in detail. This means that these methodologies enable interoperability at all levels inside of an organization. In terms nature of data managed, Zachman and TOGAF are both focused on the whole enterprise data. CIMOSA and GRAI-GIM are meant to handle CIM data, while IEM is emphasizing more on the enterprise process data. MBSE is providing a strong emphasis on the product data and its lifecycle, whilst MDSEA is targeting the manufacturing services. Finally, MDA and SOMF are both managing software-related data. In conclusion, all were developed with different purposes, yet, in the end, all of them share the same goal, i.e. to provide guidelines for the enterprises to properly manage their own EIS (Kotze & Neaga, 2010), enabling an interoperable environment inside the enterprise. Yet, none is focused on the network.

2.2. Internet of Things Frameworks

Internet of Things (IoT) is the communication between objects, devices and people, enabling in the near future, that communications and information processing to be ubiquitous and performed by IoT systems (Bermudez-edo et al., 2015). This trend leads to a rapid development of the internet and technology, and consequently, the amount of information has increased exponentially (online and technologically generated). A lack of standardization and common vocabulary has continued to generate heterogeneity, which strongly hinders information exchange and communication (Ding & Fensel, 2001). In order to be able to control these conditions caused whenever working with an IoT network. These models have to take into account that there are a wide variety of sensors or devices that communicate using different protocols and technologies, which often measure different things in very different ways and are not capable of exchanging information with other devices easily, due to being developed for specific situations. The correct standard configuration, regarding efficient ontologies of these components, is believed to be the key for a successful implementation, because it provides the needed tools and contexts to successfully integrate and configure sensors in order to manage them, or give the capability to the sensors of managing themselves, and finally monitor the network as a whole. These models propose a high-level reference architecture or ontology to identify, procedures, configuration, and relationships between different entities of different domains inside an IoT network.

The Internet of Things Architecture (IoT-A) is a guideline to support the users in the design of their own IoT solution, addressing the structural concerns related to the IoT and proposing an Architectural Reference Model (ARM), starting with the definition of an initial set of key building blocks regarding functionality, scalability, performance, deployment and security, with the purpose to eventually derive into a large set of concrete IoT-Architectures (M. Unis et al., 2013). The project instead of defining a new approach, based on work already done. Due to this choice, backward-compatibility is ensured and the solutions adopted are already established in the field. The IoT-A propose the ARM that provides the highest abstraction level for defining an IoT model, by following the IoT Architecture and the Guidelines references, the user is able to design the model, views, and perspectives of the IoT network. Including in the model is a tree with the group of communication protocols (such as WiFi, ZigBee, IPv6, and RFID) and device technologies (such as sensors, actuators, and tags). These protocols and devices are in fact the base of an efficient and interoperable IoT model because by achieving complete understanding and commitment between them, full connectivity is reached and exchanging information can become an easy task.

The IoT Reference Model aims at establishing a common grounding and a common language for IoT architectures and IoT systems (M. Unis et al., 2013). It consists of several sub-models: **Domain Model** (which describes all the relevant concepts in the IoT like devices, services, and virtual entities, and also the relationships between them. It also provides a common lexicon and taxonomy of the IoT domain (Muller, 2008)), **Information Model** (defines the structure (relations and attributes) of IoT related information in a system on a conceptual level. It can be seen as a meta-model that provides a structure for the information to be handled by IoT systems (Meissner Unis et al., 2013)), **Functional Model** (is an abstract framework for understanding the Functionality Groups (FG) that is the process to identify and relate each FG to the others. With this process, the complexity of the system is divided into smaller and more manageable parts), **Communication Model** (introduces concepts for handling the complexity of communication in heterogeneous IoT environments. It aims at the current paradigms for connecting elements and, for each case, to create an interoperable network capable of efficiently exchanging information) and the **Trust, Security, and Privacy Model**. The Figure 2-5 describes how concepts and aspects of each model are used to support others.

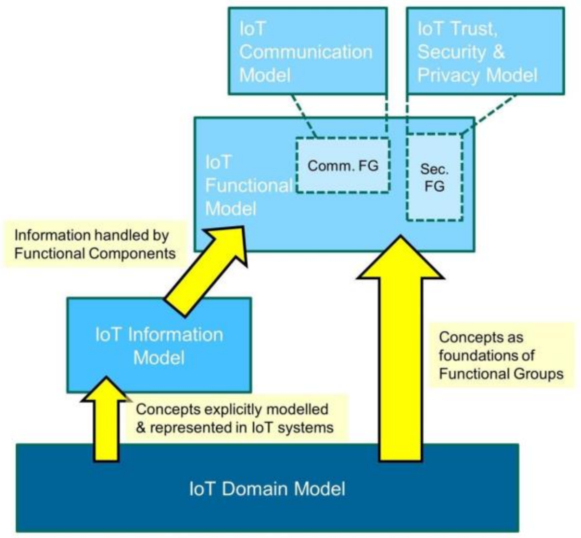

IoT Communication Model
IoT Trust, Security & Privacy Model
IoT Functional Model
Comm. FG
Sec. FG
Information handled by Functional Components
IoT Information Model
Concepts as foundations of Functional Groups
Concepts explicitly modelled & represented in IoT systems
IoT Domain Model

Figure 2-5: Interaction of all sub-models in the IoT Reference Model (M. Unis et al., 2013).

IoT-A is a very broad and complete model for organizing an IoT-Oriented Architecture. The reference model and sub-models, define the basic concepts, terminologies, and relationships in the IoT ARM. This architecture is complete and represents one step further in defining the IoT reference architecture, but it lacks the applicability that other more specific ontologies have proposed due to a more care for specific implementations and real-world situations.

Another IoT model is the Semantic Sensor Network (SSN) Ontology[7] uses an OWL[8] (a language designed to represent complex knowledge about objects, groups of objects and their relationships) ontology to describe sensors, their capabilities, measurement processes and resultant observations. This ontology contains only concepts and relations directly relevant to the sensors, leaving aside concepts related to other domains. In this way, the ontology is better positioned to provide modularity and reusability. The key specifications of the sensor information are based on their accuracy, the observations and methods used for sensing, the concepts for operating and, related with the structure for field deployments, the deployment lifetime and sensing purpose. The ten conceptual

[7] https://www.w3.org/2005/Incubator/ssn/ssnx/ssn
[8] https://www.w3.org/OWL/

modules, key concepts, and relations are shown in **Error! Reference source not found.** (Compton et al., 2012).

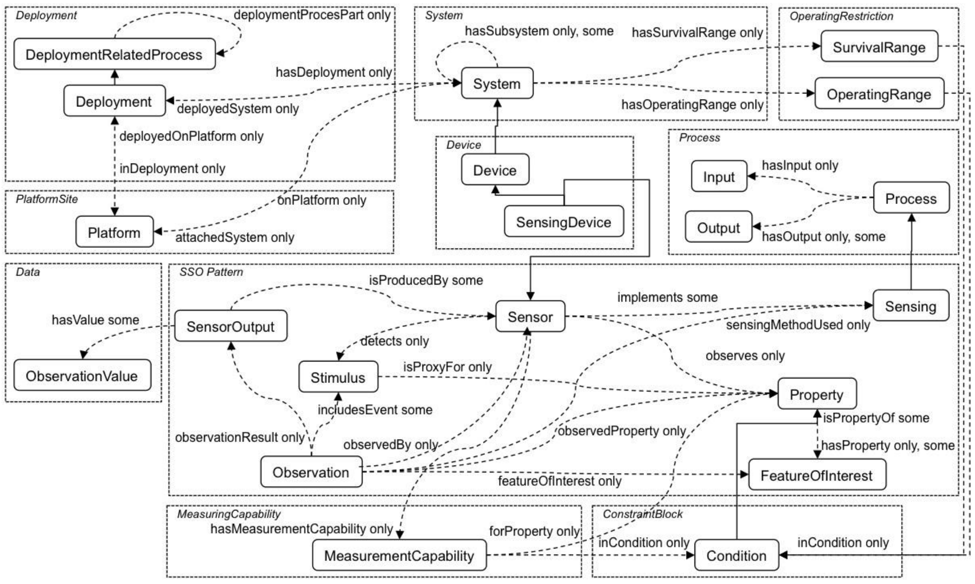


Figure 2-6: The SSN Ontology (Compton et al., 2012).

The ontology can be seen from four main perspectives (Compton et al., 2012): A **sensor perspective**, with a focus on what senses, how it senses, and what is sensed; A **data or observation perspective**, with a focus on observations and related metadata; A **system perspective**, with a focus on systems of sensors and deployments; A **feature and property perspective**, focusing on what senses a particular property or what observations have been made about a property. This OWL ontology describes a specific straightforward approach to sensors, sensing, and measurement capabilities of sensors, observations that result from sensing and deployments, or field applications, in which the sensors are used. The configuration, in this ontology, can be represented as an adaptive process which provides an interesting base for the objective of this work. The network is visualized in modules to provide better reusability and adaptation based on sensor observation. It lacks additional information regarding sensor relative entities and the domains where they are used, which means that this ontology has to be complemented with other ontologies or models in order to develop an efficient practical solution.

The IoT Lite is a lightweight ontology, based on the SSN Ontology, to represent IoT resources, entities and services (Bermudez-edo et al., 2015). By creating a representation, in a more lightweight approach than previously existed in other ontologies, it is possible to achieve an ontology able to provide shorter response time and thus create a more efficient structure for systems.

This ontology describes IoT concepts into three classes. Objects, System or Resources and Services. The devices are also split into, but not restricted to, three classes: sensing devices, actuating devices and tag devices. The services in the system are described with an availability or access control and a coverage (area covered by the IoT device). The **Error! Reference source not found.** depicts the concepts of the ontology and the main relationships between them. IoT-Lite is meant to be used with a quantity taxonomy, that allows discovery and interoperability of IoT resources in heterogeneous platforms using a common vocabulary (Bermudez-edo et al., 2015).The IoT Lite Ontology is a meta-ontology designed to be extended in order to represent IoT concepts. It also focuses more on sensing and establishes a high-level concept on actuation, which allows any future developments or adaptations in this area. It is a lighter view of the SSN Ontology, ideal for environments or specific situations in such environments, that require fast and easy processing (Bermudez-edo et al., 2015).

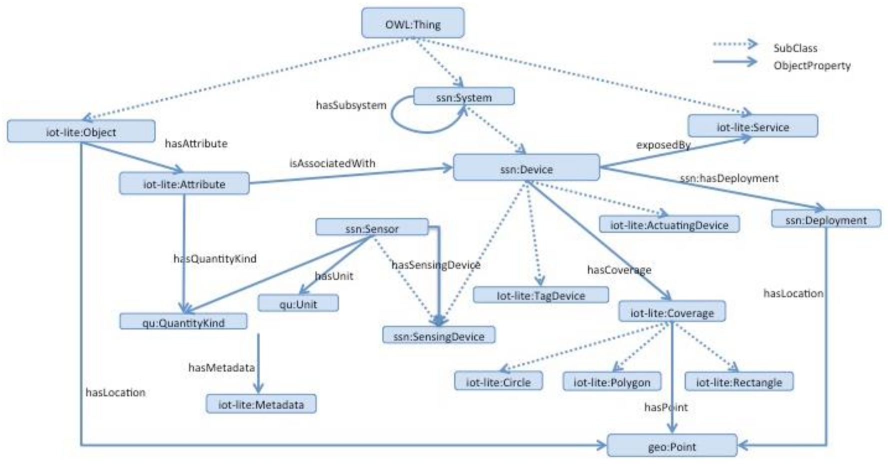


Figure 2-7: IoT-Lite Ontology (Bermudez-edo et al., 2015).

Another project that developed an IoT model was the OSMOSE Project[9] is working to design and develop a reference architecture for modeling and managing the referred sensing liquid enterprise integrating (Lampathaki et al., 2015), a view of three interconnected worlds: the **Real World (RW)**; the **Digital World (DW)**; and the **Virtual World (VW)**. The worlds represent three types of data management environments (Initiative et al., 2015), (Agostinho et al., 2015). OSMOSE follows the osmosis metaphor concept, which relates to the process of passing the molecules from a less concentrated solution (individual perception of each world) to a more concentrated one. Following the sensing liquid enterprise paradigm, osmosis processes are a special type of business processes used to moderate the information exchanged among the worlds. Since the notion of osmosis processes is conceptual, they can be modeled using the same strategies of regular business processes.

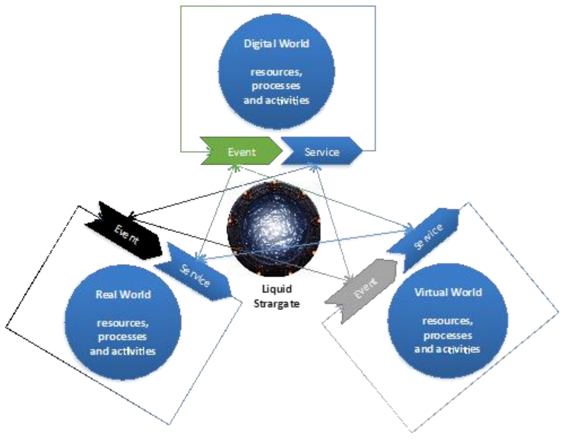

Digital World
resources, processes and activities
Event
Service
Event
Service
Service
Event
Liquid Strargate
Real World
resources, processes and activities
Virtual World
resources, processes and activities

Figure 2-8: Event-Service workflow across the OSMOSE Worlds.

[9] www.osmose-project.eu

The six Osmosis processes (see **Error! Reference source not found.**) considered are (Marques-Lucena et al., 2015): **Digitalization (RW-DW)** – Model and representation of real world data in a computer-tractable form; **Actuation (DW-RW)** – Plan and implement highly distributed decision-making; **Enrichment (VW-DW)** – Extends the computational and experiential capabilities of the Digital World annotations and projections coming from simulations and what-if hypothetical scenarios; **Simulation (DW-VW)** – Instantiate and run hypothetical future scenarios fed by Digital World data; **Virtualization (RW-VW)** – Provides data for simulation of hypothetical simulations from the real world and runs the simulation; and **Augmentation (VW-RW)** – Annotates Real World objects with Virtual World information.

2.2.1. Analysis of Different Internet of Things Frameworks

Semantic technologies are viewed today as a key technology to solve the problems of interoperability and integration within the heterogeneous world of ubiquitously interconnected objects and systems (Katasonov et al., 2006). A relevant problem for IoT related semantic descriptions is that they are not as widely adopted as expected (Bermudez-edo et al., 2015) and in the result of that, the IoT landscape nowadays appears to be highly fragmented. One of the main concerns users and developers have is that semantics increase complexity and processing time and, therefore, they are unsuitable for dynamic and responsive environments such as the IoT. A model so complete as the IoT-A tends to be difficult to apply due to existing various aspects and constraints that the developer has to deal with before even starting implementing. Some more specific models like the SSN Ontology and IoT Lite, are incisive on the problem they want to solve. By developing such ontologies for more specific solutions, or even consider them protocols (to enforce his use), creates a bigger opportunity for the scientific community to excel in the field they wish to improve, instead of dissolving into a solution, usually not giving importance to minor but still important problems, for the immensity that is the IoT environment. Nevertheless, semantic technologies are widely claimed to be a qualitatively stronger approach to interoperability than contemporary standards-based approaches (Lassila, 2005). The models described so far have been chosen because they are open source, others models could have been studied in the case of not being open source as the IBM Bluemix, Cisco IoT System, Microsoft Azure, Google Brillo, Z-Wave.

Table 2-1: IoT Models Comparison.

	W3C SSN	IoT-A	IoT Lite	OSMOSE
Interoperability	✓	✓	✓	✓
Device Discovery and Management	✓	×	×	×
Context-Awareness	✓	×	✓	✓
Scalability	✓	×	✓	✓
Management of Large Volumes of Data	×	×	×	✓
Security, Privacy, and Integrity	×	✓	×	✓
Dynamic Adaptation	✓	×	✓	×
Fast Processing	×	×	✓	✓

In Table 2-1, a comparison of the studied models is provided for better understanding. Some ontologies excel where others present weaker or inexistent solutions, which means that different applications may need different ontologies or a combination of some of them.

2.3. Methodologies and Frameworks for Interoperability

The most accurate way to achieve EI is to use standards. They are the base for creating a more homogeneous network, and are great enablers to the agreement of terminology, thus allowing communication and cooperation between software components, processes, organization units, and humans (Chen & Vernadat, 2002). Standardization initiatives, supported by standardization bodies (such as ISO17, IEC18), developed by industrial communities (e.g. IEEE19) or by European projects, have been seeking to contribute to data exchange and systems communications. However, each focuses on one particular aspect of interoperability without aligning their enterprise knowledge and skills for taking advantage of seamless cooperation (Panetto, 2007).

Many standardization groups exist, supported by local governments and international communities. Nonetheless, even among them, they have replication of efforts, while the full potential benefits could only be achieved if interoperability were underpinned by a coherent set of open, and internationally accepted ICT standards (Ray & Jones, 2006). As a result, for many projects such as ATHENA and INTEROP, ISO defined a standard framework for enterprise interoperability, is presented next.

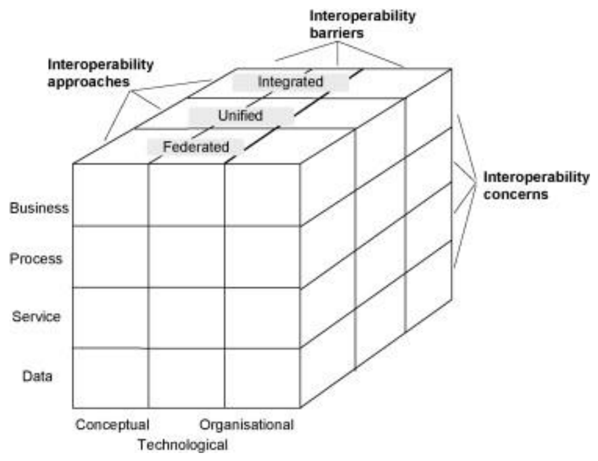


Figure 2-9: EI framework (ISO, 2011a).

ISO Standard Framework for EI, the ISO standard 11354 (ISO, 2011a) defines a holistic framework to structure the interoperability requirements to enable communication rather than defining the communication itself, and it is thus independent of specific technologies. It defines concerns, barriers, and approaches relative to EI. In a cube in which the three axes represent concerns, barriers, and approaches (**Error! Reference source not found.**). This graphic representation classifies the system and provides the right approach to solve specific problems.

The interoperability concerns describe the kinds of concerns that are relevant for enterprise interoperability (ISO, 2011b). These were adapted from the ATHENA Interoperability Framework (AIF), which is a framework created with the aim of capturing the research elements and solutions to interoperability issues (ATHENA Project, 2007). These different levels are represented in **Error! Reference source not found.** and are the four concerns used in the ISO 11354. The four concerns are Enterprise/Business, Process, Service, and Information/Data, and are important to achieve enterprise interoperability (ISO, 2011a): **Data Interoperability Concern** (Interoperability of data refers to the ability of all kinds of entities to exchange data and to relate different data models on to the other), **Service Interoperability Concern** (Interoperability of service refers to the ability of partners to request, provide, and utilize each other's services), **Process Interoperability Concern** (Interoperability of process refers to the ability of partners and other entities needed for process operation to exchange information), and **Business Interoperability Concerns** (Interoperability of business refers to the ability of enterprises to cooperate with partners for the conduct of business through interaction at various levels of their respective organizations).

Many interoperability issues are specific to particular application domains, while barriers are generic incompatibilities and mismatch that obstruct the sharing and exchange of information, these are the interoperability barriers. Following the framework for enterprise interoperability proposed in the ISO standard 11354, EI can be a difficult process due to incompatibilities and mismatches caused by three barriers: **Conceptual Barriers** (need to be detailed in terms of the syntactic (applies when different people or systems use different ways to represent information), semantic (applies when the meaning of exchanged information is not sufficiently similar), and semiotic (applies when participating entities interpret the information differently in different contexts) incompatibilities of exchanged items (ISO, 2011a)), **Technological Barriers** (represent the incompatibilities that affect the ability to exchange information. In information systems this barrier is caused by interfaces that do not allow the exchange of information to occur correctly or at all), and **Organizational Barriers** (is caused by human-related issues that directly affect the interoperation between enterprises. Organizations with different structures, methods, and policies need an increased effort to interoperate because there is a need to find mappings between partners to soften the heterogeneity).

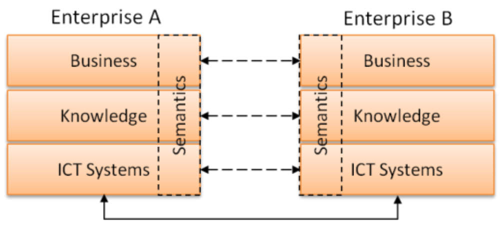


Figure 2-10: Interoperability at all layers of the enterprises (ATHENA IP 2007a).

Finally, the interoperability approaches (ISO, 2011a): **Integrated Approach** (In this approach there is a need for some kind of agreement about a common way to represent the information, ensuring a common syntax and semantic so the exchanged entities can be interpreted in the same manner by all enterprises involved in the process), **Unified Approach** (In this approach there is a need to define a meta-level structure that will allow for making mappings from one system to another. With those mappings defined it will be possible to make a translation between participating entities but at the cost of possible loss of information) and **Federated Approach** (In this approach there is no need for common meta-models to interoperate. All the actors in the interoperability process adjust their operation dynamically, using a priori information about the capabilities of the entities to be involved in the exchange).

From 2004 to 2007, the ATHENA Interoperability Framework was developed with the aim to adopt a holistic perspective on interoperability, in order to achieve genuine interoperation between enterprises. This framework (**Error! Reference source not found.**) is designed to analyze and understand the business needs and technical requirements addressing interoperability across business, knowledge, application, and data layers, following the perspective presented in IDEAS Interoperability Framework (Athena IP, 2004; ATHENA, 2010), which is described below: **Business Layer** (is located at the top of the framework. It represents all issues related to the organization, and the operations of an enterprise are addressed), **Knowledge Layer** (is responsible for acquiring profound and broad knowledge of the enterprise), **ICT System Layer** (It focuses on the ICT solutions that allow an enterprise to operate, make decisions, and exchange information within and outside its boundaries), and **Semantic** (This concerns the capture and representation of the actual meaning of concepts, thus promoting understanding). In this framework, the interoperability is fulfilled when all the layers are achieved by enterprises.

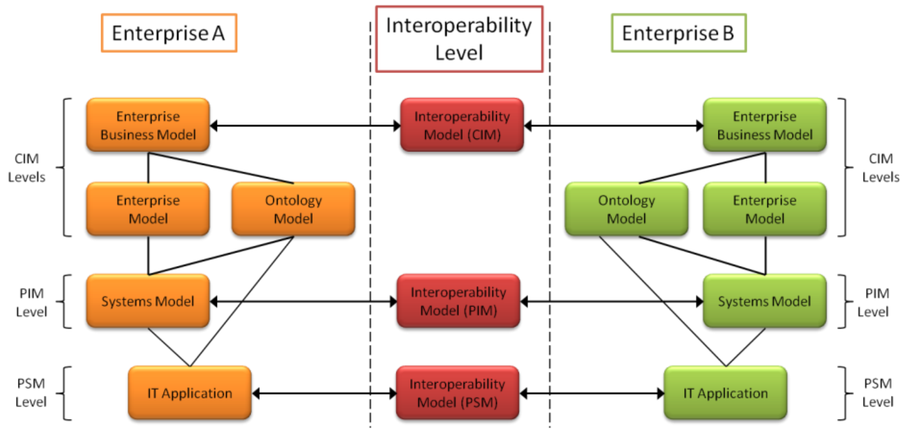


Figure 2-11: Reference Model for MDI (Chen et al., 2008).

To achieve interoperability it is required to consider the interaction in all the layers of the organization. As in the AIF, those are business, knowledge, and ICT (Information and Communication Technology) levels, are supported by semantics (see **Error! Reference source not found.**). To help in this matter a model-driven interoperability was introduced combining the guidelines of AIF and the EI standard with the MDD practices presented in the section 2.1.3.2. Model-Driven Interoperability (MDI) generates software, providing many advantages by improving portability, interoperability, and reusability through the architectural separation of concerns (Bourey et al., 2006).

The Enterprise Interoperability Science Base (EISB) was created to describe a scientific base to define and structure enterprise interoperability. Although there is no common structure or content to create a science base of neighboring sciences, it was developed a methodology, which might be applied in defining a science base, emerged based on the application of generally accepted scientific principles. This was the main concept to provide the basis of a methodology for the definition of the

EISB (ENSEMBLE Project, 2011). The definition and objectives of a science base for Enterprise Interoperability were also analyzed, supporting to define the structure of the EISB and including formalized problem and solution space as well as structured EI domain knowledge divided into twelve main Scientific Areas of EI.

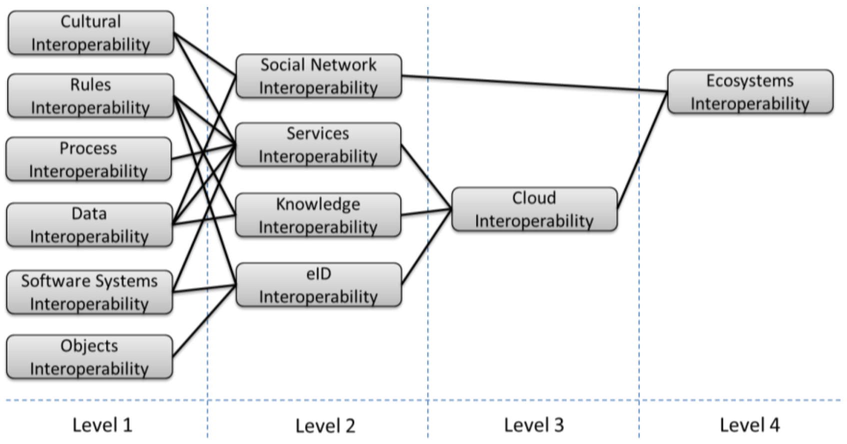

Cultural Interoperability
Rules Interoperability
Process Interoperability
Data Interoperability
Software Systems Interoperability
Objects Interoperability
Social Network Interoperability
Services Interoperability
Knowledge Interoperability
eID Interoperability
Cloud Interoperability
Ecosystems Interoperability
Level 1
Level 2
Level 3
Level 4

Figure 2-12: EI Scientific Areas.

These twelve Scientific Areas are used to support the managing of systems that need to cooperate, those are defined along four levels of complexity, where the higher levels are a composition of the lower ones, as shown in **Error! Reference source not found.** (Lampathaki et al., 2012)**.** The twelve areas are: Data Interoperability; Process Interoperability; Rules Interoperability; Objects Interoperability; Software Interoperability; Cultural Interoperability; Knowledge Interoperability; Services Interoperability; Social Networks Interoperability; Electronic Identity Interoperability; Cloud Interoperability; and Ecosystems Interoperability.

The presented methodologies and frameworks have all a common purpose, identify and help to define interoperability in an enterprise. To indicate the kind of interoperability that exist in an enterprise it is possible to know if it is possible to interoperate with other and at the same time to identify possible failures that can be found when the enterprises want to interoperate. Thus, it needs to identify which method best suits to the work that the authors wish to achieve, in addition to these methods, there are others that also help achieve interoperability and it needs to be identified, which is the case of management models, service interoperability, self-sustainable interoperability, the sensing enterprise and the concept harmonization breaking, which are described during this chapter.

2.3.1. Model Management

Currently, enterprises are increasing their use of information systems in order to improve their management. But the integration of these systems can bring some problems, such as the design, integration, and maintenance of complex application artifacts, including application programs, databases, websites, workflow scripts, formatted messages, and user interfaces (Bernstein, 2003). Enterprises have the need to store data to use to their benefit and to take the best benefits it must be organized. By using models to capture reality, expert knowledge, and salient features of complex systems (Sundaram & Wolf, 2009).

The use of models has become an integral part of the successful decision making in modem organizations. It would be difficult to find managers in major organizations who have not benefited from meaningful insights into problems through the creation and use of models. Models are physical or mathematical abstractions that, although simplified, reflect the key interactions of the system variables (Bharadwaj et al., 1993). However, the management of models involves a wide variety of functions that include the creation and editing of a model, querying and updating a model base, executing models, and generating reports. These functions are analogous to those of a database management system in which the objects of interest are data records instead of models. It is therefore not surprising to see a strong influence of DB technology in the creation of model management systems (Bharadwaj et al., 1993). For model management to support different models, meta-models and transformation between the different models are employed.

Modeling is the process of converting our perceived views of reality into a representation of them, and meta-modeling is the process of specifying the requirements to be met by the modeling or establishing the specification which the modeling process should fulfill. As shown in **Error! Reference source not found.**, a model implies that the modeler abstracts properties from things in order to obtain a representation of the physical world. This process of abstraction can be applied to modeling itself, to obtain a model of the modeling process, which is called a meta-model. The meta-model defines concepts of a given domain, their relationships, and forms a modeling language used to create executable domain models (Smolik, 2000; Van Gigch, 1991).

The main goal of the meta-model is to specify what is important to define in a model about the system. Since it is the meta-model that specifies what elements may be contained in the model and how they relate to each other. When used in the real world, the meta-model is a specification of a domain-specific modeling language, a language used to express the requirements of the system and define exactly how it should realize them (Smolik, 2000). To help with of well-formalized models, MDE provides developers with uniform modeling notations for a wide range of modeling activities. The UML become the standard of MDE, along with several technologies related to modeling (Bran Selic, 2003; B Selic, 2003; Sacevski & Veseli, 2007).

A way of representing what two models have in common is the concept of morphism, which is described in mathematics as an abstraction of a structure-preserving map between two mathematical structures (Ogren, 2000). Recently, this concept has gained some meaning in computer science, more exactly in systems interoperability. This new meaning of morphism describes the relationships (e.g. mapping, merging, transformation, etc.) between two or more IS specifications.

In the pursuit of a sustainable interoperability solution, authors address morphisms, namely, mapping representations to reach an interoperable state between EISs. They specify the relationship between two information models, either structural or conceptual as is the case of ontologies. There exist several classes of morphisms, and in this subchapter, a selection of these classes are described.

Two classes of morphisms exist, those that modify the operand, and those that do not. The first is classified as model-altering and the second as non-altering. Below is a description of these two classifications (Agostinho et al., 2007; Agostinho et al., 2010): **Model non-altering morphism**

(Supported in the traditional model-mappings, in this class of morphism changes to the source models are not applied. The relationship is identified among two or more existing models, pointing to similarities and convergences), and **Model altering morphism** (Use a kind of transformation function to transform the source model (operand) into a set of rules (operator), thus modifying the targeted output. The model altering can be divided into two categories, model transformation and model merging. Model transformation occurs when a source model (A) is modified by a function F in order to obtain a new output model (B) expressed, or not, in the same language).

Given two models M1 and M2, Bernstein in (Bernstein, 2003) describes a morphism over M1 and M2 has a binary relationship over the objects of the two models. Thus, a mapping between models M1 and M2 is a model, map12, and two morphisms, one between map12 and M1 and another between map12 and M2. A mapping is, therefore, a relationship between two models. This relationship is made one meta-level higher than the transform, so the mapping describes the rules used for the transformation (ATHENA, 2010). In most cases, the model has a higher entity, and above that one (represented by the relating entities and attributes), these morphisms that are related with a higher morphism are called the sub-morphisms of that one. An example is in **Error! Reference source not found.**, where the Person is the higher morphism, and the relating entities are the Name and Age.

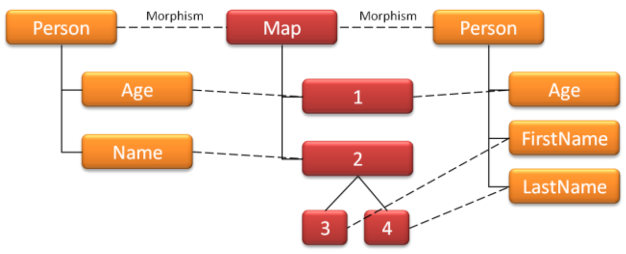


Figure 2-13: Example of mappings between two models.

A mapping defines the concept of a relationship between models. This means that instead of representing the relationship as a set of pairs, (of objects), a mapping represents it as a set of objects (each of which can relate objects in the two models). In **Error! Reference source not found.** both information structures represent models to describe a person. As an example of potential mapping difficulties, when it is necessary to establish a relationship between the two entities, conflicts between them can occur. In that case, it is about the name, since in the second model it is divided into the first and last name. Because of this issue, it is important to complement mappings with specific knowledge that identifies whether they are complete, and if not complete, then identify the differences as well as possible.

Another type is the model transformation technologies that take a model as input and generate another model as output. This is made by defining rules at the meta-model level (Benguria & Larrucea, 2007). An example is in **Error! Reference source not found.**, where we find the rules in the middle of the two models, which identifies the transformation. Then, a direct transformation is made between model A with model B. The transformation is normally used when it is necessary to pass information from one enterprise to another. The information is passed to the exact model that the other enterprise uses, avoiding interoperability issues.

2.3.2. Service Interoperability

The service interoperability concept is directly connected with that of Service Oriented Architectures (SOA), namely with the planning and customization of services to achieve software interoperability (Agostinho, 2012). Sakao and Shimomura, in (Sakao & Shimomura, 2007), define service as an activity through which a provider causes a receiver to change from an existing state to a new state that the receiver desires, where both a content and a channel are means to realize the service. This definition fits well with the requirements of this work.

SOA combines the capability of invoking remote objects and functions (services) using specific tools for dynamic service discovery as if they were local, placing an emphasis on interoperability. Services are supplied by a provider and delivered through a service channel to a receiver, in which it is important to have complete integration between them (ORACLE, 2009).

Consequently, it is of utmost importance to create an interoperable channel that harmonizes different services available. This is one of the points of this dissertation, i.e., to create an integrator channel between the different providers. Since each can be using different technologies, and if they are not integrated, an error can occur in the communication. This situation will cause an increase in the interoperability failures related to the increase of the complexity of the service. Due to this fact, it is important to have good interoperability in the service, independent of the type of service (ORACLE, 2009).

Thus, it should define what channels to use to integrate the various enterprise throughout this chapter several proposals are made, starting with SOA that in literature several descriptions to describe it are made. For example the W3C[10] describe SOA as "a set of components which can be invoked, and whose interface descriptions can be published and discovered", while the Open Group (The Open Group, 2013) describe SOA as "an architectural style that supports service - orientation - a way of thinking in terms of services, service-based development and the outcomes of services".

For simplicity, in this dissertation considers SOA as a set of components that are published somewhere on the web, that allows an enterprise to manage services. These will have a considerable impact on the manner in which to manage the software life cycle, ranging from the specification of requirements as services, the design of services, acquisition and outsourcing as services, to asset management of services, and so on (Microsoft, 2004). It is for this reason that it is very important to specify the SOA before implementing the WS.

In recent years WS has been increasingly used by enterprises to communicate with each other through networks. Due to the importance of the WS, several organizations have been giving it more attention. The reason behind this growing importance is related to the use of web-applications by the platforms that do not have access to the web using web browsers. The web-applications are built around the web browser standards and can be used by any browser on any platform. Web Services

[10] http://www.w3.org/TR/ws-gloss/

have two types of uses (W3Schools, 2013): **Reusable application-components** (There are several applications on the web, so by using WS it is possible that it be shared by several enterprises. Avoiding the need to make new ones, over and over), **Connect existing software** (Web services can help to solve the interoperability problem by giving different applications a way to link their data. With web services one can exchange data between different applications and different platforms).

With the increased use of WS by enterprises in order to provide better service to its customers to meet their demands, enterprises need to continue to innovate in their products, in response to this demand, cloud computing appeared, providing several kinds of services to customers, allowing each person to customize according to their wishes and needs.

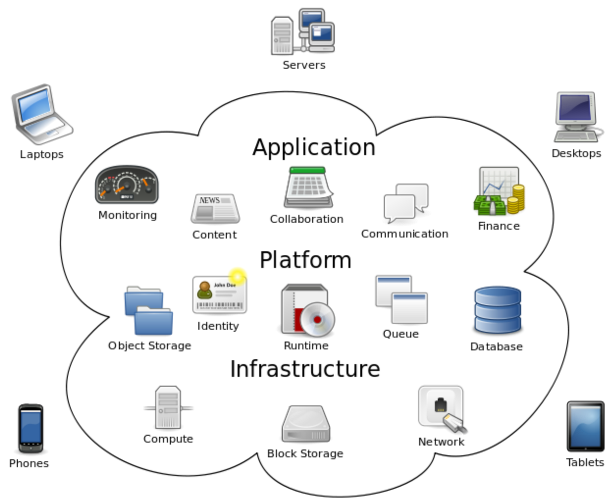


Figure 2-14: Cloud functionalities (WikiMedia, 2009).

In recent years cloud computing has seen tremendous growth, by providing a vast range of services to clients. These services include the provision of music, movies, television series, etc., and it is expected that the cloud model will continue to grow in the future. Mell & Grance in (Mell & Grance, 2011) define cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction". The cloud thus provides access to different resources that can be of different types, such as hardware, software, and data resources. **Error! Reference source not found.** shows an example of cloud computing.

Another way to provide services to customers is through the Internet of Services (IoS), which aims to use the internet for new ways of value creation in the services sector. In (Terzidis et al., 2012) the authors describe IoS as a commercial transaction in which one party grants temporary access to resources of another party in order to perform a prescribed function and a related benefit. Resources may be a human workforce and skills, technical systems, information, consumables, land, and others.

It is important for the enterprises to have the capacity to compete with each other. With every passing day, this competitiveness grows more dependent on the capacity of each enterprise to answer to customers' needs, and consequently, to be able to adapt to the changes required (Service Web 3.0

Project, 2009). Another important feature is to maintain the integration intact, in order to assure that the services continue to communicate with each other.

2.3.3. Self-Sustainable Interoperability and the Sensing Enterprise

It is important today for enterprises to improve the way they use their resources, with the aim of obtaining more profitability from them. Another aspect of this addresses the cases of enterprises working in a network, and in which those enterprises wish to reach to the market as quickly as possible. With this in mind, this chapter addresses the issues of Self-Sustainable Interoperability and Sensing Enterprise. These are two current topics aimed at supporting enterprises in these issues.

2.3.3.1. Self-Sustainable Interoperability

The United Nations Organization defines sustainability as the "development that meets the needs of the present without compromising the ability of future generations to meet their own needs" (United Nations, 1987). In other words, what we do in the present will affect the future, as in the case of an enterprise that chooses a system without thinking if the system will be the best one in the future. Perhaps the models in use are not the best ones, or perhaps the price is the best solution (greatest concern) today. These issues can lead to serious problems in the future, and as a result, creating a sustainable environment in the enterprise is a major concern.

Creating a sustainable environment in the enterprise will bring benefit(s) by making the system more robust and resilient, thereby boosting the system's ability to answer to and solve problems that may occur, and to adapt to changes as necessary. Interoperation should be addressed at different levels. The diversity, heterogeneity, and autonomy of software components, application solutions, business processes, and the business context of an enterprise must be considered. Interoperability's definition has been revisited a number of times (ATHENA IP, 2007b).

In some cases (e.g. collaborative product development), Systems Engineering (SE) is a special case of EI, as it requires integration of all disciplines and areas of expertise from different enterprises into a team effort forming a structured development process that proceeds from concept to production to operation and then disposal. It also considers both the business and technical requirements of all customers, with the goal of providing a quality product that meets all user demands (INCOSE, 2011).

Such engineering models may be defined in different languages or semantics, and thus morphisms are needed to describe and formalize the relationships between them, and sustainability methodologies are needed in order to cope with market dynamics: manufacturing systems are constantly adapting to new market and customer requirements, thus answering the need to respond with faster and better quality production; and new organizations are constantly entering and leaving networks, leading to a constant fluctuation and evolution of system models. All these factors make interoperability difficult to maintain (Agostinho & Jardim-Goncalves, 2009), and there may be some problems arising these situations, these can include:

- **Different legacy systems** - In the world several legacy systems used by different enterprises exist. Each has its own features and characteristics. Some of them follow standards with the aim of allowing the enterprises to communicate with each other. In some cases, the systems were developed many years ago using technologies that obsolete, not being prepared for integration with another system. Yet, enterprises are afraid of change for new software, since are risky and may not be successful. However, interoperability is difficult to achieve due to the different systems;
- **Distribution around the world** - Big corporations started to distribute their manufacturing facilities around the world, with the aim of having cheaper products. This requires changes in their legacy systems, forcing the enterprises to be dynamic in order to adapt their models to those changes. One way to solve this problem is to have a Dynamic Manufacturer Network (DMN) that will manage the network, centralizing the system. Even though there are different systems on the network, it is needed to create a central EIS/model that will create an interoperable environment in enterprises, harmonizing the communication between the different systems, avoiding misunderstandings between them;
- **Culture** - In an international network, one of the biggest problems is culture. Each culture has its own philosophies and objectives that influence each person, and as a result, the models and system will be different, creating interoperability issues;
- **Language** - Language is always a problem, as it invariably causes people to mismatch their meanings in a conversation. In a system, the problem is the same – a semantic problem. To solve this problem, the systems need to be integrated to allow them to understand each other;
- **Communication** - Communication is a problem that can be related to culture and language, as different cultures and languages can create friction in communication. If enterprises do not agree on the standard to be used in the communication, this can cause considerable problems between them;
- **Devices** – The management of devices is a tricky business since existing thousands of devices and different aspects that can create difficulties in the moment of co-operating with other device/entity. Today exist several brands of devices, each brand follows their own protocols and technologies in their devices. Since they have their own protocols allows the cooperation between devices of the same brand, yet creates interoperability problems when using devices of different brands. Looking careful with the devices and legacy systems, they have the same problems, different protocols, technologies, architectures. Yet, they are at different levels, one at the software level another at hardware level. Although of the similarity of their problems, the approach to solve it is different.

Despite these problems being different between them, they have a point in common. Every time, that one of these problems occurs it will create problems in the network, and the interoperability is reached, the set of organizations within a network demonstrate stability exchanging e-messages following established laws (i.e. mapping morphisms). Therefore, at this point, networks display symmetry and organizations are operating regularly. However, that may change according to the environment. If just one of the network members adapts to a new requirement, the harmony is broken, and the network begins experiencing interoperability problems. This is called harmonization breaking (Agostinho, 2012).

However, the ideal way to prevent these situations from happening during the time that the enterprises are cooperating, thereby maintain interoperability on the network. One way this happens is to identify the shortcomings in time, avoiding an impact on the partnership and delayed production. To mitigate these problems, it is necessary to constantly monitor the network, looking for changes and to maintain the interoperability, creating a Sustainable Interoperability Environment. Agostinho and Jardim-Gonçalves in (Agostinho & Jardim-Goncalves, 2009) proposed the CAS to Support Sustainable Interoperability Framework (CAS-SIF). The Self-Sustainable Interoperability Framework is inspired by CAS-SIF and follows its paradigm. The goal is to provide systems with the capacity for self-recognition in the face of environmental (i.e. the market) adversities. However, to avoid falling into chaotic behavior, CAS-SIF uses an adaptive intelligence at both the network and systems level. In Figure 2-15 the framework is represented and decomposed into four layers:

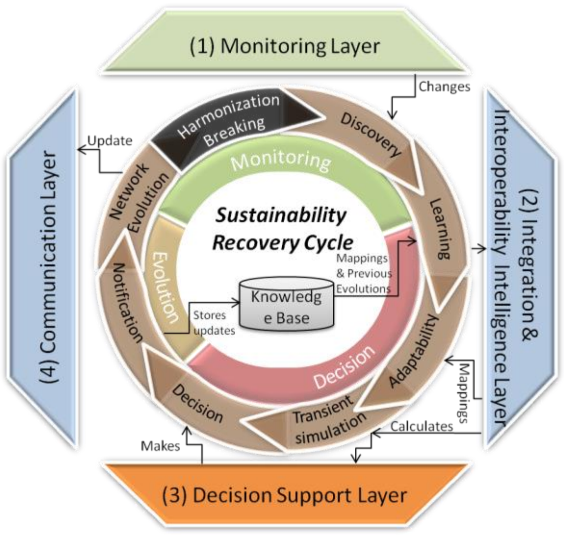


Figure 2-15: CAS-SIF Framework (Agostinho & Jardim-Goncalves, 2009).

- **Monitoring** – The monitoring layer addresses different stages, from capturing information to its analysis, and is structured into specific components in order to meet a set of requirements. The whole process has to be carried out balancing extensibility and self-description capabilities with compactness. When the interoperability is established by the organization, stability is achieved by exchanging e-messages following established laws. But sometimes small fluctuations acting on a system may cause it to cross a critical point and behave unexpectedly, which is designated as harmonization breaking. Figure 2-15 shows that this is detected by the monitoring layer, which is then responsible for analyzing it and discovering the changes that caused the anomaly;
- **Integration & Interoperability Intelligence** – After detecting the harmonization breaking, a learning process should be triggered to learn more about the changes that have occurred, and the nodes adaptation required. CAS-SIF enables the adaptation of systems and the

optimization of the maintenance process through dynamic model morphisms, using knowledge representation technologies applied to the model management domain, such as the tuple described above. As in Figure 2-15, each enterprise's Communication Mediator provides information about similar mapping and previous evolutions that have occurred so that the system can learn and propose, in the adaptability process, a new or updated mapping;

- **Decision Support** – Changes at the internal or interface structures of the organization's information systems can lead to unexpected situations. In these cases, the CAS-SIF must consider some kind of decision support to allow the manager or any other decision maker to take the final word regarding whether or not to execute the mapping proposed in the adaptation. As illustrated, this layer calculates a transient indicator to understand how the network will suffer from the proposed (re)adaptation, and support the decision;
- **Communication** – The CAS-SIF framework is meant to be implemented among the several network nodes that are adopting this level of interoperability. In this way, this layer is responsible for the re-adaptation of a network node, and the communication to the entire business network in such a way that it causes minimal disruption to the other members of the network.

The CAS-SIF will support the enterprise to reach a sustainable interoperability environment. The sustainability recovery is made along the adaptive organization lifecycle (monitoring, decision, evolution), considering monitoring and decision support capabilities in the specification of a framework to implement the conceptual solution's second package. This framework must exhibit:

- **Discovery capabilities** – Detecting when enterprise systems are updated in the network, driving into harmonization breaking;
- **Learning and adaptability** – After detecting the harmonization breaking a learning process should be triggered to acquire knowledge about the changes that have occurred, and the nodes adaptation required, by means of morphisms knowledge bases defined at interoperability establishment. It should enable the adaptation of systems and the optimization of the maintenance process, using knowledge representation technologies, applied to the model management domain, namely dynamic MoMo;
- **Transient simulation and decision** – To understand how a network, as an integrated complex system, will evolve during the transient period, and how adaptations, if implementation is decided, affect the overall system and network behavior;
- **Notification/Communication** – Provides information about the way the network nodes should react so that they obtain information for their own needed adaptations and the information system, and for the entire network to evolve toward the interoperable state (equilibrium) as swiftly as possible.

2.3.3.1.1. Intelligent System Methodologies to Support a Self-Sustainable Interoperability System

In order to have a self-sustainable interoperability environment inside a network of enterprises, i.e., to have a network in which enterprises may enter, leave and evolve without breaking the harmonization of the network, it is necessary to implement intelligent system methodologies capable of acting/reacting accordingly with the information monitoring. As each enterprise has its own legacy systems and standards, a study is made on Enterprise Modeling and Engineering during this chapter (Chapter 2.1), where techniques and architectures to support the modeling of the enterprises' systems were mentioned. The importance of this study was to identify how their systems work and their specifications to identify main points, to be used as a roadmap for the integration between them. Another point of this study was to identify the level of interoperability of each legacy system. A study was made on Monitoring and Evaluation (in Chapter 3) to be used as procedures for identifying and monitoring the level of interoperability of an enterprise. The ultimate goal is to know the status of each enterprise, and if it is needed to propose solutions to improve the network interoperability. This will help to realize intelligent solutions towards the objectives of this work.

In this section, some principles about the systems engineering have been analyzed. However, knowing how they work is not enough to achieve the objectives proposed in this work. To create the Self-Sustainable Interoperability Framework proposed in the hypothesis is needed to identify how a system can recover from a change and adapt to that odd circumstance. Some concepts and methodologies are needed to follow to allow reaching the desired goal, some of these points are described here. It is needed to identify the legacy system and models in use, therefore gathering baseline knowledge. Then, the systems are evaluated based on the interoperability maturity. Finally, it is important to identify the steps to be implemented to create the interoperability of the enterprise within the network using one (or more) of the possible methodologies and frameworks for interoperability. As soon as interoperability is achieved, it becomes necessary to maintain it, and for that reason, it is necessary to create an adaptive system so that the methodology will be able to adapt to the diversities that it encounters along the enterprise's network lifecycle. This system can include four phases: 1) Plan; 2) Execution; 3) Evaluation; 4) Learning; embedded within a monitoring module, with the possible entry points for self-adaptation. The self-adaptive system starts to make a plan to the second self-adaptation entry point, motivated by "reaction", relies on the premise that every time that a problem is found, it must be solved. For that reason, an evaluation of the system is made to identify what kind of interoperability problem occurred, and then plan again searching for a solution. Of course, in these cycles, the self-adaptive system has learned with possible mistakes in order to avoid its repetition in the future.

The system, therefore, needs to be prepared to adapt to what is happening in the environment. In summary, the main research aim is to create a proactive adaptation and predictive monitoring methodology, based on the monitoring of the network. This opens new topics to study, namely:

- **Machine Learning** – It needs to give to the architecture the feature to take advantage or past occurrences to improve future actions. The culture and language are interoperable problems, which need to be addressed. Thus, the learning ability is needed to allow the framework to respond to problems appropriately;

- **Planning** – It needs to plan the future to evolve in response to an existing problem. The architecture will identify problems, and with them generate goals to recover the state of harmonization;
- **Self-Adaptive System** – This is the accumulation of all the points discussed until now. It will provide the system to adapt to adversities and to evolve over time, avoiding and minimizing harmonization breakings in the network.

The ideal self-sustainable interoperability system is a system that is able to act independently, to discover a problem and being able to decide and recover from the harmonization breaking. However, in some situations, the machine needs the support of the human in the decision. Yet, it is possible to give suggestions to the human, and then the human selects the best option, in some cases, the human can purpose a better option. At this point the machine learning will be used, the system will encounter a problem and it will try to solve, in the case that is not possible, it will ask for support. In these situations, the system will learn over time. The planning is used in the situations that it identified the solution and it will prepare a goal for the plan.

2.3.3.2. The Sensing Enterprise

The Sensing Enterprise (SE) concept was created by the FInES Cluster (FInES Cluster, 2009) with the support of the European Commission, upon the advent of the Augmented Internet (FInES Cluster, 2011). The community acknowledged the fact that enterprises are desperately in need of innovative ideas to adapt, remain competitive, or sometimes simply survive in the digital era. SE is an attempt to reconcile traditional non-native "Internet-friendly" organizations with the tremendous possibilities offered by the cyber worlds (from the clouds to the dust). It refers to "an enterprise anticipating future decisions by using multi-dimensional information captured through physical and virtual objects and providing added value information to enhance its global context awareness" (Santucci et al., 2012). The SE concept envisions the enterprise as a smart complex entity capable of sensing and reacting to stimuli, by integrating decentralized intelligence, context awareness, dynamic configurability, and sensorial technology into its decision-making process (Danila et al., 2013).

SE seeks to extend the notion of IoT by leveraging the power and ubiquity of ICT networks and systems, and the characteristics and properties of smart objects. Another feature of SE has to do with the types of sensors (such as RFID, sensors, actuators, smart cards, etc.), which will be an integrated part of the enterprise as object information. These characteristics will give to the enterprise systems a sense and alertness to the environment, giving the possibility in real time to know what is happening across the enterprise and enterprise networks. In another point, the enterprise systems will have the capacity to respond and react to different sets of business stimuli (FInES Cluster, 2012). Such systems become self-adaptable, self-organized, self-optimizing, self-configuring, self-protecting, self-healing, self-describing, and self-discovering. In short, the Sensing System will acquire a multiplicity of self-* properties that can be summarized as "self-reinvention" (Man-Sze, 2012). Sensing Enterprise brings several benefits to the enterprise systems (FInES Cluster, 2012), including the following:

- **Context awareness** – The Sensing Enterprise receives real-time information about its environment. This information will be "multi-dimensional", originating from a vast array of sources via a potentially unlimited number of devices. Real-time information from both the physical and virtual surroundings of the enterprise is constantly fed into its decision-making process;
- **Dynamic configurability** – Dynamic configurability is needed to support a system with a great deal of flexibility in how it operates. Since IoT systems are potentially made of thousands of nodes and devices, their configuration is complex and is critical for the integrity, robustness, and security of the enterprise system;
- **Information requirement and processing** – With the availability of massive amounts of information on tap, the enterprise is a dynamically configurable collection of smart components informed by smart objects, and will need to act upon such information and react to changing contexts in real time and on a scale which is unprecedented;
- **Multi-identity enterprises and enterprise entities** – IoT potentially transforms our notion of the enterprise and enterprise entities. An enterprise that operates within and straddles the worlds of the physical, virtual, and digital will have multiple identities served by multiple interacting enterprise systems. It may have specific business objectives within each of those worlds, and the information required will be different;
- **Relationships between humans and objects** – The Sensing Enterprise posit potential new relationships between people and objects as collaborating and possibly competing partners in an enterprise environment. As indicated, they may even become "equal" partners in decision-making. The interactions between people, their digital representations, and objects will be a key issue.

Following the idea and challenges behind the SE, in (Agostinho et al., 2014) the authors identify several new research areas that can be related to the State-of-the-Art of the Sensing Enterprise:

- **Cyber-Physical Systems (CPS)** – These are engineered systems in which physical components are tightly intertwined with computational elements. Borders are "liquefied" and it is believed that these systems will revolutionize not only production but also mobility and healthcare, by facilitating the communication between intelligent context-aware entities (Nikolaus, 2013);
- **Smart Tags & RFID** – These are important components for CPS and Wireless Sensor Network development, and are already widespread in many industries from automotive assembly lines, to pharmaceuticals, to clothing, etc.;
- **Ambient Connectivity** – Brings the ability to assume connectivity of anything, anywhere, and anytime independently of the means and providers (Frankston, 2009; Frankston, 2013);
- **Competitive and Customer Intelligence** – Applying methods such as crawling, scraping, or data mining to gather information about the surrounding environment (including

competitors and markets), as well as knowledge of technological developments (Gilad, 2008);

- **Model-Based Systems and Service Engineering** – These promote modeling to address many of the limitations of the traditional document-based engineering approach. They provide a more rigorous means for capturing and integrating requirements, design, analysis, verification, and validation throughout the system's later life cycle phases (Operations & Crisp II, 2007; Y. Ducq et al., 2012). This enables more understanding between development teams and the other stakeholders, as well as traceability features that facilitate properties such as backward compatibility, in which the enterprises of the future should be able to interoperate with non-evolved enterprises;
- **Self-Sustainable Interoperability** – The principles have already been explained and can here be enlarged into becoming the "glue" to the previous areas. Connected to other interoperability research (e.g. EI, MDI, etc.), it also includes other relevant topics such as complexity management, model and service matching, transformation, monitoring, and strategic decision-making. It can be an important asset in the development of FInES, and especially the SE, as enterprises will need to permanently adapt to meet their requirements while maintaining interoperability.

All of these domains are closely related, and the SE cannot exist without all. The research presented in this work is focused on self-sustainable interoperability, namely on identifying a way to create a sustainable environment inside an enterprise network. In this regard, this work is connected with SE since both of them seek to harmonize the enterprise systems on a network.

2.3.3.3. Internet of Things

The IoT has reached many different players and gained further recognition, by means of application areas, Smart Cities, Smart Car and Mobility, Smart Home and Assisted Living, Smart Industries, Public Safety, Energy & Environmental Protection, Agriculture, and Tourism as part of a future IoT Ecosystem. Each year IoT is more important to our lives (Vermesan & Friess, 2013).

Complementary to IoS, IoT is a concept and a paradigm that considers pervasive presence in the environment of a variety of things/objects/devices, which through wireless and wired connections and unique addressing schemes are able to interact with each other and cooperate with other things/objects/devices to create new applications/services and reach common goals (Vermesan & Friess, 2013). The end users are spread around the world and go to the IoT to access different things (objects/devices) that are represented. This adds a new dimension to the world of information and communication technologies (ICTs): with anytime, anyplace connectivity for anyone, we will now have connectivity for anything (International Telecommunication Union, 2005).

2.4. Analysis of State of the Art towards Hypothesis

Despite the existence of several EIS and solutions for integration in the market, frequently enterprises continue to use traditional legacy systems and proprietary models for engineering analysis, simulation, reliability, costing, PLM, etc. When looking to SMEs, this situation is worse, since frequently they do not have the funds to manage changes and evolutions and may lag behind their competitors. This, therefore raise a sustainable interoperability problem for enterprises every time they work in a collaborative network because if enterprises use a specific EIS that are not supported or interoperable with others, they become more resilient to change and evolve with the networks. The same happens if they use physical devices that will not be possible to replace or reconfigure freely, since each brand of the device has their own characteristics, complicating the integration with other devices. Enterprise researchers know about this problem and have searched for solutions. The most common way to integrate models between different partners without "touching" their existing systems is to establish P2P mappings between the different models (Agostinho & Jardim-Goncalves, 2009), this solution with transformation between models enables the creation of a homogeneous system in which it is possible to send data from one side to another without problem in interpreting the data. So also, it facilitates the problem related to the communication, as normally enterprises need to “negotiate” what kind of data to send and its structure. However, overlooks the use of standards in their systems, which would create a more homogeneous network. Despite having several propositions to solve the interoperability issue, few authors address the subject of how to maintain the interoperability status, and this remains an open issue.

This chapter supported to prepare the answer to three research questions presented in Chapter 1.3.2. The first question is answered throughout the dissertation, partially answering in this chapter. By identifying existing EIS and IoT frameworks and how they work, one step towards the characterization of an interoperable network is identified, while methodologies and frameworks are identified to support in the identification of the interoperability in the enterprises. The second question is studied in this chapter with the study on methodologies and frameworks to identify and evaluate interoperability in an enterprise, aiming to identify one or more methodologies or frameworks to use in this work. The fourth question is also partly answered in this chapter, since by identifying the EIS, IoT frameworks, and methodologies/frameworks for interoperability support to identify if an enterprise has a low integration, so it can try to identify ways to react in these situations.

3. Interoperability Monitoring and Assessment

A relevant problem for the interoperability monitoring and assessment is the identification and measurement of the level of interoperability in a network. This is an important step to establish a baseline and identify the gap to a sustainable interoperability environment. Monitoring and assessment will lead the system to discover what kind of interoperability services are/need to be implemented in the enterprises, and at the same time give the possibility of preparing new measures to improve. Based on the compatibility of different information models, assessment methods can measure the interoperability between two EIS, and categorize it according to a predefined maturity scheme.

To support the monitoring and assessment of interoperability the several topics were identified and are structured in Figure 3-1. First, different interoperability maturity schemes are presented in the right part of the figure; these interoperability maturities can be used as a way to categorize the interoperability between systems. Then, several run-time monitoring paradigms and evaluation strategies are represented in the middle and left part of the figure. These will allow the system to monitor and assess the maturities that are implemented. Since all systems are different, each will be a different case, so it is necessary to personalize the evaluation for each situation. To do so, one must choose performance indicators (KPIs) that will be used in the evaluation and will ascertain if the system is following them. These KPIs can be of different data types, depending on the needs of the users, and will be specified at the outset of the evaluation. For this purpose, data management is very important, since it will manage the data that is acquired in the monitoring phase, and which will facilitate the processing of the KPIs.

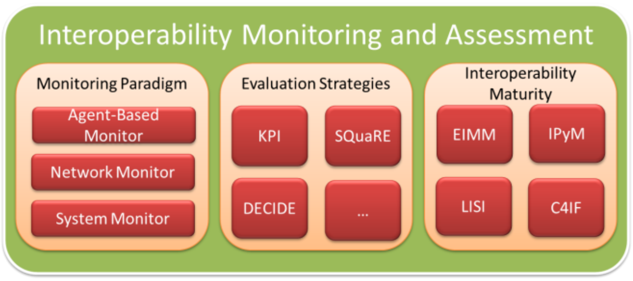


Figure 3-1: Interoperability Monitoring and Assessment.

For the monitoring, different paradigms exist, as the System Monitor (SM), in which the EIS is being monitored, providing data to the SM, which is saving and displaying it to users. With this system, it is possible to identify changes performed that will avoid problems in the future. Another paradigm is the Network Monitor (NM). It is similar to the SM, but in this situation, instead of monitoring a system, it is monitoring a network. Using the same principles as an SM, the information collection function of a network management will allow monitoring network devices that are located in remote locations. Due to this fact, it is necessary to remotely monitor each node of the network, and since more people communicate using networks, new networks have appeared each year, making this task more complex (Wong, 1997).

3.1. Interoperability Maturity

In the past few years, several authors have presented different solutions to reach the interoperability status. Normally, interoperability types follow a scale of advancement, in which the higher a type is placed in the scale, the more advanced is the interoperability maturity achieved. For this reason, the interoperability types are sometimes called levels, in which to reach an upper level of interoperability, all previous levels have to be successfully addressed (Peristeras & Tarabanis 2006).

Maturity levels describe the stages through which they are defined, implemented, and improved. They have the objective of providing a guide to select improvement strategies by determining the current capabilities of specific processes and identifying the issues most critical to quality and process within a particular domain, such as software engineering (C4ISR 1998). Several typologies are discussed next.

Connection, Communication, Consolidation, Collaboration Interoperability Framework (C^4IF) was presented in (Peristeras & Tarabanis, 2006). It is a framework that uses the basic linguistics concepts to the domain of Information System (IS) communication. With these concepts four maturity layers were defined: **Connection** (Refers to the ability of information systems to exchange signals. To succeed in this, a physical contact/ connection should be established between two (or more) systems); **Communication** (Refers to the ability to exchange data in ISs. To succeed in this, a predefined data format and/or schema needs to be accepted by the interlocutors; In this type at least two levels of communication exist, the first is the exchange based on a commonly accepted data format, the second is more advanced, and the exchange includes data); **Consolidation** (Refers to the ability of ISs to understand data. To succeed in this, a commonly accepted meaning for the data needs to be established between the interlocutors); and **Collaboration** (Refers to the ability of systems to act together. Action results in changes in the real world. To succeed in this, a commonly accepted understanding of performing functions/services/processes/actions needs to be established between the interlocutors or ISs. One of the greatest advantages is that the 3 areas are considered separately).

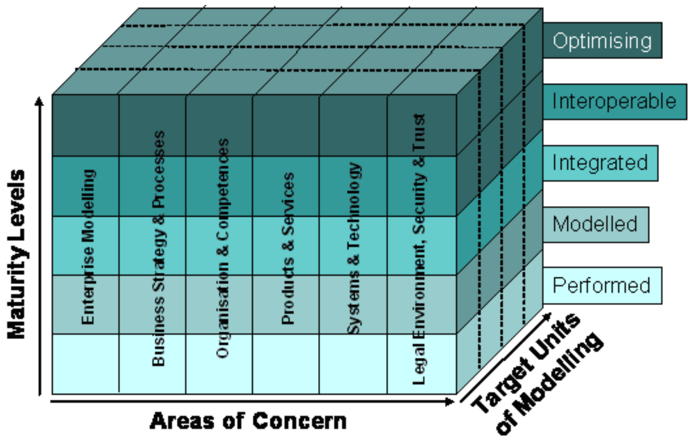


Figure 3-2: The Three Dimensions of the EIMM (ATHENA IP, 2012).

An early classification was defined in the Levels of Information System Interoperability (LISI) (Vida et al., 2012), which focuses on assessing systems against increasing levels of sophistication that focus on exchanging and sharing information and services throughout the system's lifecycle. This occurs through the five layers described below: **Level 0 – Isolated Interoperability in a Manual Environment** (These systems cover the wide range of isolated, or stand-alone, systems. The direct electronic connection is not allowed or is available, so the only interface between these systems is by manually re-keying or via extractable, common media); **Level 1 – Connected Interoperability in a Peer-to-Peer Environment** (These are capable of being linked electronically and providing some form of simple electronic exchanges, with a certain limitation. They are capable of passing homogeneous data types, such as voice, simple e-mail, or fixed graphic files such as GIF or TIFF images between workstations); **Level 2 – Functional Interoperability in a Distributed Environment** (These systems reside on local networks that allow data sets to be passed from system to system. There is an increase of complexity in media exchanges with the use of the formal data models); **Level 3 – Domain-Based Interoperability in an Integrated Environment** (These systems are capable of being connected via wide area networks (WANs) that allow multiple users to access data. A domain-based data model is present (logical and physical) that is understood, accepted, and implemented across a functional area or group of organizations that comprise a domain. To express the increase of capabilities between the levels); and **Level 4 – Enterprise-Based Interoperability in a Universal Environment** (These systems are capable of operating using a distributed global information space across multiple domains. Multiple users can access and interact with complex data simultaneously, and these data are shared and distributed throughout the system).

Another classification is the Enterprise Interoperability Maturity Model (EIMM), which is a maturity model and an application procedure to perform assessments for interoperability maturity, developed in the ATHENA Project. It was developed with two objectives in mind, first to identify the capabilities for interoperability of an enterprise and second to derive an adequate modeling concept (ATHENA IP, 2007a).

The aim of the EIMM is to perform the assessment that will outline the steps to be implemented by the rest of the framework (ATHENA IP, 2007a). Regarding the maturity itself, it has five maturity scales. These maturity levels are represented in **Error! Reference source not found.** and described below (ATHENA IP, 2007a): **Performed** (Enterprise modeling and collaboration is done but in an *ad-hoc* and chaotic manner. The organization collaborates with external entities (suppliers, administration, customers), but the relationships are not planned thoughtfully); **Modeled** (Enterprise modeling and collaboration are done in a similar way each time, the technique has been found to be applicable. Defined meta-models and approaches are applied, responsibilities are defined, people understand the enterprise model and know how to execute it, and network technologies are used to collaborate); **Integrated** (The enterprise modeling process has been formally documented, communicated, and is consistently in use. The organization uses a defined methodology and infrastructure for enterprise modeling, the different dimensions are integrated among themselves and the model is traceable to the enterprise systems. There is a knowledge base used to improve the models, and business collaboration is facilitated through interoperability technologies, use of standards, and externalization of part of the enterprise models); **Interoperable** (Enterprise models support dynamic interoperability and adaptation to changes and evolution of external entities. The workplaces of the people are seamlessly adapted to the enterprise model); and **Optimizing** (Enterprise models allow the organization to react and adapt to changes in the business environment in an agile, flexible, and responsive manner. Enterprise systems are systematically traced to enterprise models and innovative technologies are continuously researched and applied to improve interoperability).

Finally, in 2009, Agostinho and Jardim-Goncalves (Carlos Agostinho & Ricardo Jardim-Goncalves 2009) proposed another approach to interoperability levels classification that divides the interoperability types into five layers, called Interoperability Efficiency Pyramid Model (IPyM), in which enterprises are evaluated and given a level of the model depending on the type of interoperability implemented in their systems. This allows enterprises to improve their interoperability systems, and then reach a better level of the pyramid, which means that they have a better interoperability implemented and will facilitate the integration of a network.

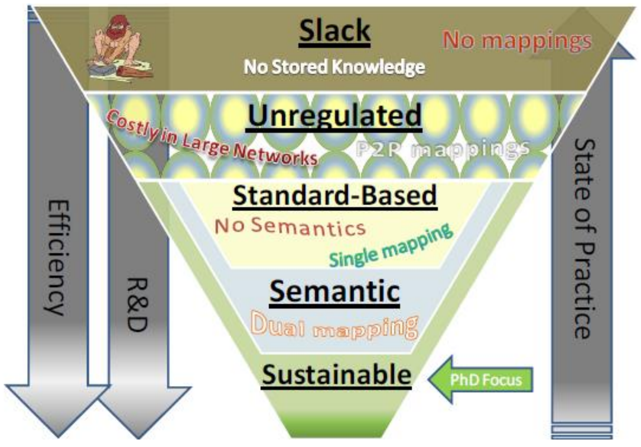
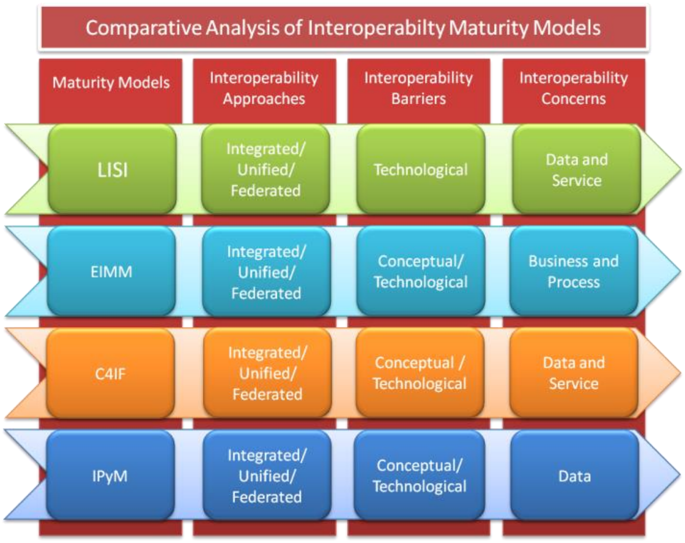


Figure 3-3: Interoperability Practices Layers (Carlos Agostinho & Ricardo Jardim-Goncalves 2009).

In **Error! Reference source not found.** the levels of the Interoperability Efficiency Pyramid Model are illustrated: ***Slack interoperability*** (When there is no previous understanding between the sender and receiver, there are rudimentary communication methodologies with little support of advanced IS); ***Unregulated interoperability*** (Organizations work with peer-to-peer relationships and maintain their own data format and business rules mediated by as many mappings as the number of business partners); ***Standard-based interoperability*** (Several dedicated reference models covering many industrial areas are based on common collaboration models); ***Semantic interoperability*** (Athena (Athena IP, 2004) identified semantics as a cross-cutting issue along the four levels of interoperability within an enterprise since a system might be able to exchange data with another but still not be fully interoperable. This layer is therefore based on the previous and complements the reference model with reference knowledge (e.g. ontology), so that the content of the information exchanged is annotated (Missikoff et al., 2003; Sarraipa et al., 2007) and unambiguously defined: what is sent is the same as what is understood); and **Sus*tainability Interoperability*** (The simplest way to describe the term "sustainability" in this context is that it is related to the aim of improving the quality of service by contributing to a more robust interoperability, avoiding excessive consumption of resources (e.g. man-power and time), when the dynamicity of system and networks causes harmonization breaking. It is a novel concept whose objectives are to reconcile the economic interests ensuring that the network maximizes its efficiency by remaining interoperable at most times in any of the three previous levels (exception for slack interoperability)).

3.1.1. Maturity Schemes Comparative Analysis

This study is divided into three viewpoints as in ISO-11354. Each viewpoint can be a different choice, for example, the Interoperability Concerns can be of type Data, Service, Business, or Process, and this will depend on each maturity concern. These aspects can be identified in Figure 3-4, in which it is possible to see the difference and overlap between the different maturity models.



Comparative Analysis of Interoperabilty Maturity Models

Maturity Models	Interoperability Approaches	Interoperability Barriers	Interoperability Concerns
LISI	Integrated/ Unified/ Federated	Technological	Data and Service
EIMM	Integrated/ Unified/ Federated	Conceptual/ Technological	Business and Process
C4IF	Integrated/ Unified/ Federated	Conceptual / Technological	Data and Service
IPyM	Integrated/ Unified/ Federated	Conceptual/ Technological	Data

Figure 3-4: Interoperability Maturity Models comparison.

Each maturity level represents the interoperability in its own way. However, each one has the same aim, i.e. to allow the representation of the level of interoperability inside of an enterprise or network. This is made by positioning the system under evaluation in one of the levels. Then if the enterprise makes changes in their EIS it must be re-evaluated to see if it is possible to improve the level or not. This is the case of the C^{4}IF since it is a cumulative maturity model. In the other cases related to the rest of the maturities (e.g. LISI), they are not cumulative. In other words, it is not needed to fulfill a level to pass to the next one. To be given a layer it is needed to meet the requirements of each layer. To exemplify this table, let's follow the IPyM example, in the interoperability approach case, it has all the three options (Integrated, Unified and Federated) and it is represented by proposed levels that represent (in different ways) all the three options. In the interoperability barriers, there are two options (Conceptual and Technological), the conceptual since the IPyM consider the problems of syntactic, semantic and semiotic. The technological due to the fact that one of the main issues presented in the IPyM is how the enterprises communicate and how the models can be interoperable. Finally, it is the Interoperability Concerns, which has one option (Data). The IPyM is focused on the models and how the communication is done, is the reason for Data to be chosen.

A mapping between the different maturities schemes studied is proposed in Figure 3-5, by positioning all the levels between them and identifying the levels that are equal or similar to the ones of the other schemes. By doing this exercise, it is possible to see that LISI, EIMM, and IPyM are similar in their philosophy to interpret the level in the network. Still, each level represents the same idea, but focuses on different perspectives, as explained in Figure 3-5, for example, the IPyM is more for focused in a network, the others are more peer-to-peer connections. In the case of C^{4}IF classification is cumulative, thus it does not have a direct relationship with the others.

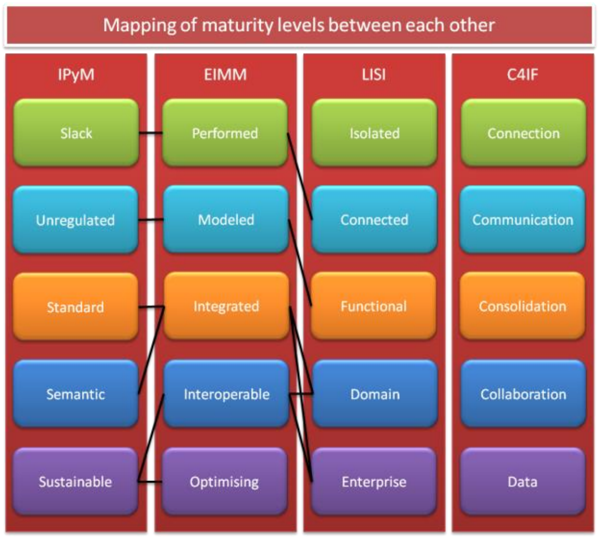


Figure 3-5: Mapping of maturity levels between each other.

3.2. Evaluation Strategies

In the last section, the importance of monitoring and evaluating systems was explained, which is different than monitoring, evaluation is an important point for developers since it will help them to be sure that their software is usable and meets the needs of users. Several definitions and techniques of evaluation exist, some of which involve users directly, while others call indirectly on an understanding of users' needs and psychology (Preece et al., 2002).

The evaluation varies depending on the themes in the cause, in this situation the focus is the system evaluation. To be an effective interaction, it is necessary to know how to evaluate different kinds of systems at different stages of development. Many techniques are available for supporting this evaluation, and in this chapter, some of these techniques are discussed (Preece et al., 2002).

The use of an evaluation methodology is needed since users want systems that are easy to learn and use as well as effective, efficient, safe, and satisfying. So, it is needed to identify what and when to evaluate, being important keys to the success of a system. It must be checked that users can use the system and like it, and at the same time, it meets their needs. For these reasons, developers use evaluation. However, another problem is related to the diversity of the systems in use, which increases the range of features to be evaluated (Preece et al., 2002).

Many evaluation approaches exist, but none of the approaches is the best for all solutions, as each approach has its own particular set of strengths and limitations. In some cases, it happens that two or more approaches are combined to obtain better results (Kahan & Kael Consulting, 2008). In this chapter, several evaluation methodologies and frameworks are analyzed.

3.2.1. Performance Measurement

The **Performance Indicators** (PIs) or **Key Performance Indicators** (KPIs) are quantifiable performance measurements used to define success factors and measure progress toward the achievement of business goals (iSURF Project, 2010). Each enterprise will analyze its mission, identify all its stakeholders, and define their goals, and it needs a way to measure progress toward those goals. Due to this, KPIs are important, since they are the measurements that will evaluate the status of the enterprises. However, the KPIs will differ depending on the enterprise (Reh, 2013), so, it is important to define the best KPIs for each case, depending on the goals that the enterprise wants to reach. Since KPIs vary from enterprise to enterprise, ServiceNow (2013) created a KPI library to facilitate the integration between enterprises. This library harmonizes the KPIs between each enterprise, supporting the idea of a sustainable interoperability environment.

The KPIs are one possible solution to measure the performance of each enterprise, and at the same time have the advantage that each enterprise defines their own KPIs depending on their own needs. This customization is largely due to the fact that each market is different and needs different ways to perform the evaluation. For this reason, the KPIs can be used in the interoperability maturity models to support the definition of the indicators to reach a specific level of maturity. By combining the KPIs with the ISO 11354, it will be possible to evaluate the enterprise with a considerable amount of customization, since it can be possible to adjust the criteria of ISO 11354 for each particular case, by choosing the best indicators to be used. These indicators will differ from enterprise to enterprise, depending on the system, standards, and models implemented. This measure will support them to identify their failures and what is needed to change in order to obtain good integration. However, KPIs can be used to evaluate and at the same time to monitor. This is made by creating metric KPIs for monitor purpose, where the system needs to fulfill these metrics to accomplish the desired goal.

Another approach to evaluating performance is the Goal-Question-Metric, which is a top-down approach to establish a goal-driven measurement system for software development, in which the team starts with organizational goals, defines measurement goals, poses questions to address the goals, and identifies metrics that provide answers to the equations. Goal-Question-Metric defines a measurement model on three levels.

3.2.2. Conformance Testing and Interoperability Checking

A good start for a stable system with high chances of being interoperable is to adhere to a well-specified standard or reference model. However, this simple approach turns into a huge issue when some details are neglected and can compromise the whole interoperability process. In an enterprise environment even when two or more enterprises adopt the same reference model to exchange data, it is not guaranteed that they can achieve an effective data mapping without syntactic or semantic errors. The ability to evaluate the adherence or non-adherence of a candidate implementation to a standard is called conformance testing (ISO, 1993a). To execute this kind of test there is a need for a special dedicated test system with full control, access, and observability connected to every single System Under Test (SUT). Some methodologies have been proposed and standardized to assist in the development of a conformance test platform, as described next.

The OSI Conformance Testing Methodology and Framework (ISO 9646) is a standard based on ISO/IEC 7498-1–OSI Reference Model for Open Systems (ISO, 1993b), which defines a common base to allow the intercommunication of open systems. This standard defines a methodology divided into three stages. In the first are defined the purposes of the tests, which allows the creation of the Abstract Test Cases (ATC) to be applied. The second stage is to select from among all the tests, which will be applied and generate a valid and executable test. The last stage is the test execution phase of the Implementation Under Tests (IUT), generating reports about the conformance status of the implementation (ISO & JTC, 1994).

The 30th part of STEP (STEP Conformance Testing Methodology and Framework) (ISO, 1993b) defines another methodology and a framework to apply conformance tests based on ISO 9646. The main stages are identical to the OSI 9946, except the execution that divides the tests into two types: **Pre-processed** (In these the reference model is inserted in the application in order to produce corresponding data and the output is compared with the expected results), and **Post-processed** (A data example is used as the input of IUT, and inferences are made about how this example is handled, in order to determine if it is being interpreted correctly).

ETSI also describes a methodology for application of conformity tests, i.e., the ETS 300 406 - Methods for Testing and Specification (MTS) (ETSI, 1995). The first part of this methodology is dedicated to identifying the test purpose and its structure (Test Suit Structure - TSS). Subsequently, based on the test purpose and TSS, the ATC is defined and described in TTCN. The tests applied by this methodology are similar to those implemented by ISO 9646. First, the TP (Test Purpose) and TSS are defined so that later the ATCs descried in TTCN-3 can be developed, a redesign of TTCN made by ETSI that will result in ATS.

Nevertheless, even having the above methodologies, how can we know if a system will be interoperable with another implementation of the same reference model, even when meeting all the requirements specified in the standard or reference model? Conformance testing can evaluate if the implementation is in conformity with all the requirements, but even that cannot guarantee the same semantic interpretation or the implementation of the same module (if the standard envisages multiple conformance options). Thus, there is a need to complement conformance testing with interoperability checking systems. The majority of the research to explore quantitative measures for describing interoperability relationships fails because it considers the system as black boxes and has no concerns about the details and semantics. In order to address this, Yahia et al. (2012) proposed a preventive approach based on formalizing the semantic relationships between systems by analyzing the detailed semantics of their conceptual model and relationships. In opposition to the ETSI approach which evaluates if working systems are interoperable, this approach evaluates if systems are goal candidates to become interoperable.

This work extends the well-known interoperability definition stating that two information systems (IS1 and IS2), in the context of cooperative enterprises, have to satisfy the following properties:

- IS1 needs to be able to communicate some information with IS2;
- IS2 needs to be able to understand at least partially the semantics of the information exchanged;
- IS2 needs to operate on that exchanged information.

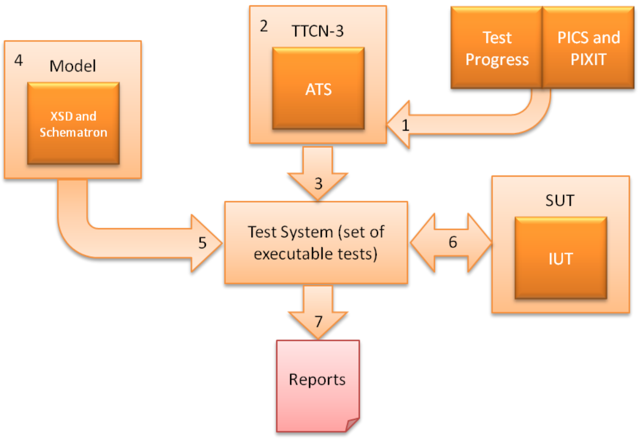


Figure 3-6: funSTEP conformance test methodology.

A new evaluation of a mixed method is also added by selecting a core of mandatory concepts, due to the fact that non-mandatory concepts are not necessary for the implementation to operate correctly.

One example of a mixed approach is the funStep approach (FunStep, 1999). In this case, the conformance test used relies on a methodology that provides two types of tests, the basic preliminary tests applied to check the conformance of the implementation under test, and the tests that assess the capability of the implementation in comparison with the PICS (Protocol Implementation Conformance Statement).

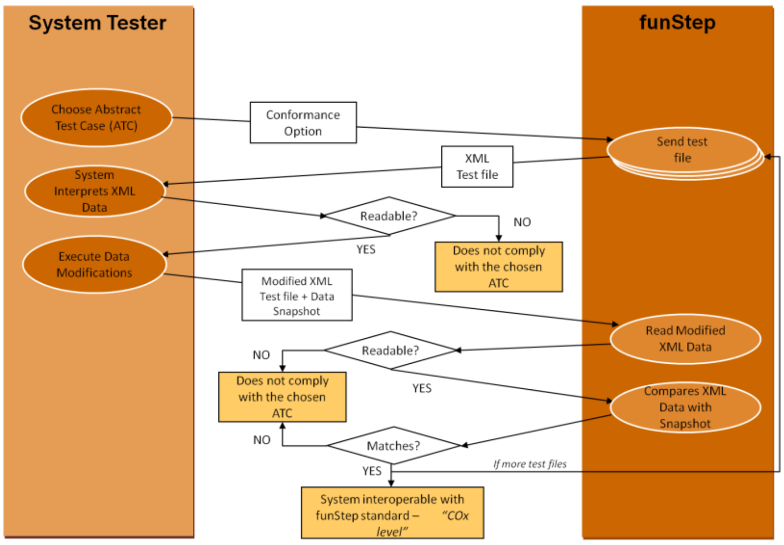

System Tester
funStep
Choose Abstract Test Case (ATC)
Conformance Option
Send test file
XML Test file
System Interprets XML Data
Readable?
NO
YES
Does not comply with the chosen ATC
Execute Data Modifications
Modified XML Test file + Data Snapshot
Read Modified XML Data
Readable?
NO
YES
Does not comply with the chosen ATC
Compares XML Data with Snapshot
NO
Matches?
YES
If more test files
System interoperable with funStep standard – "COx level"

Figure 3-7: funStep Interoperability Checking Methodology (FunStep, 1999).

These two types of tests meet the needs of two kinds of users, those that need only to check if the data are in conformance with the syntax and semantics of any STEP conceptual model, and the users that also need to evaluate the characteristics described in PICS. The methodology, illustrated in **Error! Reference source not found.**, comprises seven distinct phases (Onofre, 2007). Those reports that result from the execution of the defined tests have the information about the errors that were found.

The funStep interoperability checking is complementary to conformance testing and proves that end-to-end functionality between two or more systems conforms to what is required by the standard that rules them (ETSI, 2003). To execute this kind of testing there is a need of a qualified equipment, that comes from a different supplier of Equipment Under Test, and those tests are based on functionality as experienced by a user (ETSI, 2003). The main difference between this methodology and the ISO/IEC 9646 resides at System Under Test, which is now composed of an Equipment Under Test and one or more Qualified Equipment that works as the reference of an ideal implementation. Thus, the TP and TSS are also derived from standards but are now focused on testing specific functionalities of the EUT.

"Conformance and interoperability are both important and useful approaches to the testing of standardized protocol implementations although it is unlikely that one will ever fully replace the other" (ETSI, 2003). In the funStep project, it was developed an interoperability checking methodology to check if it is possible to exchange data information between different entities without incorrectness and misinterpretations. Since, the existing methodologies did not fill the requirements of funStep, due to their focus on specific systems that were designed, it was created this methodology, based on ATS (presented in **Error! Reference source not found.**). This is a semi-automatic methodology that allows the users to check if their systems are interoperable with other funStep compliant systems. It uses a set of generic files to validate the interoperability of a system, it is based on ISO10303-AP236 and uses XML technology (FunStep, 1999).

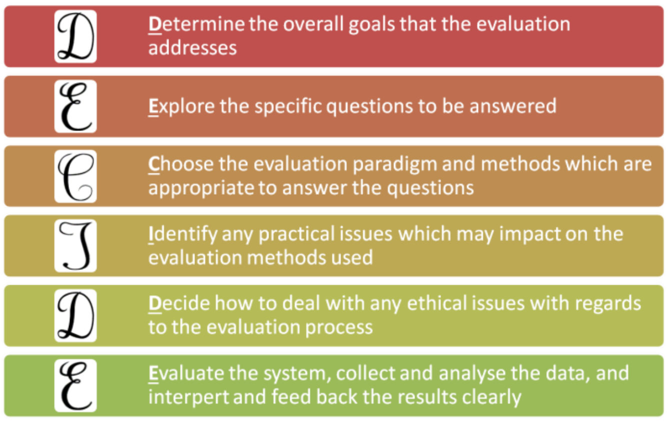

D
Determine the overall goals that the evaluation addresses
E
Explore the specific questions to be answered
C
Choose the evaluation paradigm and methods which are appropriate to answer the questions
I
Identify any practical issues which may impact on the evaluation methods used
D
Decide how to deal with any ethical issues with regards to the evaluation process
E
Evaluate the system, collect and analyse the data, and interpert and feed back the results clearly

Figure 3-8: DECIDE Evaluation Framework.

Finally, to help in the evaluation, some frameworks exist that seek to support in the planning of the goals that are needed for a good evaluation of the system. One of these frameworks is the DECIDE framework, which provides the following checklist to help evaluators. In **Error! Reference source not found.** the DECIDE Evaluation Framework is presented (Preece et al. 2002; iSURF

Project 2010).

SQuaRE (Software Product Quality Requirement and Evaluation) (ISO, 2005) as know ISO/IEC 25000:2005. It was developed to provide requirements and recommendations for the specification of software product quality requirements (British Standards, 2007). The SQuaRE is a conjunction with ISO/IEC 25010 and the other parts of the SQuaRE series (ISO/IEC 25000 - ISO/IEC 25099) of standards. The SQuaRE is divided into six divisions (ISO, 2012): **ISO/IEC 2500n: Quality Management Division** (This division defines all common models, terms, and definitions further referred to by other standards from the SQuaRE series); **ISO/IEC 2501n: Quality Model Division** (This details the quality models for computer systems and software products, quality in use, and data. Practical guidance on the use of the quality models is also provided); **ISO/IEC 2502n: Quality Measurement Division** (This part includes a system/software product quality measurement reference model, mathematical definitions of quality measures, and practical guidance for their application); **ISO/IEC 2503n: Quality Requirements Division** (This division helps to specify quality requirements based on quality models and quality measures); **ISO/IEC 2504n: Quality Evaluation Division** (This provides requirements, recommendations, and guidelines for software product evaluation, whether performed by evaluators, acquirers, or developers); **ISO/IEC 25050 - 25099: SQuaRE Extension Division** (These standards currently include Requirements for quality of Commercial Off-The-Shelf software and the Common Industry Formats for usability reports). The main objective of these standards is to provide requirements and recommendations for the specification of software product quality requirements (British Standards, 2007).

Following the work of Yalia et al. (Yahia et al., 2012) and the funStep, presented in Chapter 3.2.2, and to complement the explanation on how this work can support the measuring of the mappings, in here have an extended explanation. Having as an example, the mappings between two elements allow to identify the different existing relationships between the conceptual model entities, this relationship defines the correspondence between both elements. Going deep in the study of the relations arises three properties that characterize them:

- **Property 1: Non-symmetry** – Interoperability is not a symmetric relationship, that means that one element of a system can be interoperable with another, but the reverse does not occur. Especially in systems that only one interoperability direction is needed;
- **Property 2: Maximal potential interoperability** – When not only the core semantics, but also the non-mandatory is considered to define the interoperability relationships, and all the concepts are instantiated;
- **Property 3: Minimal effective interoperability** – Restricting the relationships to the core semantics provide a guarantee that they are effective, but the interoperability is classified as effective and minimal.

To accomplish the quantitative evaluation of the conceptual models there is a need for formalized measures and specific information about the mappings defined in a real state and an expected state:

- $\boldsymbol{R_c^2}$ – Represent the mappings defined from one model to another;

- $\boldsymbol{R_c^{2e}}$ – Represent the mappings containing only mandatory concepts and entities;
- $\boldsymbol{R_{c\,expected}^2}$ – Represent the mappings expected.

Taking into consideration the Property 2 and 3 and the formalized measures defined arises the Table 3-1.

Table 3-1: Measures for Interoperability Assessment.

Type of Evaluation	Interoperability	Measure	Value Conclusion
Maximal Potential Interoperability (MPI)	$V_{1-2} = \frac{\lvert R_c^2 \rvert}{\lvert R_{c\,expected}^2 \rvert}$ $\varepsilon_{1-2} = \frac{\lvert R_c^{2e} \rvert}{\lvert R_c^2 \rvert}$	= 0	S1 is not interoperable with S2
		< 100%	S1 is partially interoperable with S2, but only a percentage of the relationships with S2 (R_c^2) are effective. V_{1-2} can be reached if all concepts taking part into (R_c^2) are mandatory
		= 100%	S1 is potentially fully interoperable with S2, but only a percentage of the relationships with S2 (R_c^2) are effective. V_{1-2} can be reached if all concepts taking part into (R_c^2) are mandatory
Minimal Effective Interoperability (MEI)	$V_{1-2} = \frac{\lvert R_c^2 \rvert}{\lvert R_{c\,expected}^2 \rvert}$ $\varepsilon_{1-2} = 100\%$	= 0	S1 is not interoperable with S2
		< 100%	S1 is partially interoperable with S2 and this partial interoperability is effective
		= 100%	S1 is fully interoperable with S2 and this interoperability is effective

3.3. Monitoring Paradigm

One advantage of the self-sustainable interoperability paradigm is to provide the possibility for enterprises to do their own maintenance. In other words, the system is able to identify needs changes in the models and systems then identify if these changes can be a problem to the network or not. This will avoid problems in the future since it is identified on time, and at the same time, it will make it possible to predict incidents, providing an opportunity to solve them before the problem occurs. For this to be possible is important and complementary to evaluation.

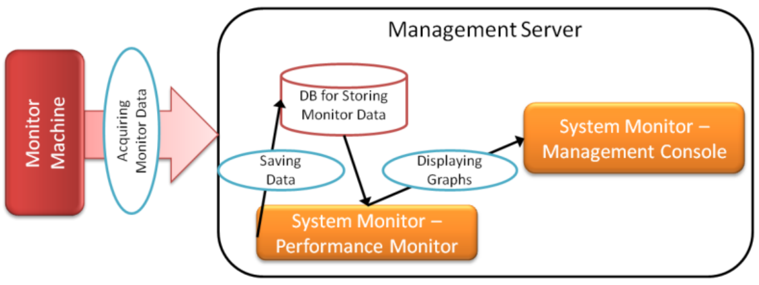

Monitor Machine
Acquiring Monitor Data
Management Server
DB for Storing Monitor Data
Saving Data
Displaying Graphs
System Monitor – Management Console
System Monitor – Performance Monitor

Figure 3-9: Example of basic SM (NEC Corporation, 2012).

Monitoring will systematically perform the collection and analysis of information as a project progresses. It helps to improve the efficiency and effectiveness of a project or organization. This will enable to determine if resources available are enough and are being used well and if what was planned is actually being done correctly. Implementing a monitoring system will not make problems disappear, but will bring some benefits such as the following (Shapiro, 2003): Help to identify problems and their causes; Suggest possible solutions to problems; Raise questions about assumptions and strategy; Provide the system with information and insight; and Encourage the system to act on the information and insight.

Looking to the advantages of monitoring and evaluation, it is clear that they will assist in the maintenance of the interoperability of the network, by identifying problems before they occur. However, it is necessary to identify the needs of the monitor and evaluation. What is involved in achieving the proposed objective relating to monitoring (Shapiro, 2003)? The following features are included: Establishing indicators of efficiency, effectiveness, and impact; Setting up systems to collect information relating to these indicators; Collecting and recording the information; Analyzing the information; and Using the information to inform day-to-day management.

A System Monitor (SM) is typically a software program that supports the monitoring of the performance status of systems. By using such software, it is easier to keep abreast of the performance status of the system by collecting data from the system and displaying these data graphically. The SM provides several functionalities to the user, monitoring the system with the greatest accuracy possible. These functionalities are (NEC Corporation, 2012): **Performance Monitoring Service** (This functionality acquires performance data of the system involved. It collects and store the data in a database); and **Management Console** (This is the user interface of the SM, and is where the results of the monitoring are shown using graphics data).

In **Error! Reference source not found.** a simple example of an SM is presented. We see there the machine that is being monitored to provide data to the SM, then the data is saved and displayed to users. With this system, it is possible to identify changes performed that will avoid problems in the future.

Since all systems are different, each system will be a different case, so it is necessary to personalize the monitor for each situation. To do so, the tone must choose performance indicators (KPIs) that will be used in the monitoring and will ascertain if the system is following them. These KPIs can be of different data types, depending on the needs of the users, and will be specified at the outset of the monitoring. For this purpose, data management is very important, since it will manage the data that are acquired in the monitoring phase, which will facilitate the processing of the KPIs.

Another form of the monitor is the Network Monitor (NM) is similar to the SM, but in this situation, instead of monitoring a system, it is monitoring a network. Using the same principles as an SM, the information collection function of a network management will allow monitoring network devices that are located in remote locations. Due to this fact, it is necessary to remotely monitor each node of the network, and since more people communicate using networks, new networks have appeared each year, making this task more complex. Maintaining a good network is important for good network monitoring, so the applications need to be effective in checking the status of their network, to guarantee good service (Wong, 1997).

An alternative way to monitor models is through tracking and analyzing the streams of data about the retrieved data of the sensors, for that, it is used a Complex Event Processing (CEP). CEP is an emerging network technology that creates actionable, situational knowledge from distributed message-based systems, databases and applications in real time or near real time. CEP can deliver the capability to define, manage and predict events, situations, exceptional conditions, opportunities and threads in complex networks. Implementation scenarios range from network management to business optimization, resulting in enhanced situational knowledge, increased business agility, and the ability to more accurately (and rapidly) sense, detect and respond to events and situations. By using a highly scalable and dynamic solution such CEP can contribute to extracting a higher level of knowledge from a situational information abstracted from processing business-sensory information (GemStone, 2005).

The CEP with the support of services can warn the user and monitor devices about pre-defined KPIs, for this reason, it was made a study of different CEP engines: **EsperTech CEP**[11] – is an open source for runtime architecture, platform administration, and event processing features. It is an engine embeddable in Java architecture, with strong CEP feature set and open source status make it a top candidate to be embedded in other vendor tools or applications. One of the features of this CEP is the Event Processing Language (EPL) that was designed for expressing situations and fast execution against both historical and currently arriving events, triggering actions when the incoming data satisfy the predefined rules (HTC Gllobal Services, 2016); **WSO2 CEP**[12] – is a lightweight, easy-to-use, open-source CEP server for real-time analytics of events. Detects events in real-time, actuating on the most significant events that are in the event cloud, being an high performing and massively scalable, supported by features like event partitions, mapping database values to events (WSO2, 2015); and **Fitman Dynamic CEP**[13] – is a CEP that supports complex real-time processing pipelines, resolving the issue of data-driven detection. It was created focusing the smart factories, and based on technology provided by Espertech, being extended to react more expediently to particular situations. Manufacturing depends on different data information sources that are very heterogeneous and dynamic, making this a need for the processing infrastructure being able to process. One of the advantages of this CEP is the possibility to change patterns inside the CEP improving the detection process to decrease the false positives (FITMAN, 2015).

Although several other CEPs exist, some are based on Esper event processor engines such as WSO2 Event Processor and Fitman DynamicCEP. Each CEP has their own Application programming interface (API), based in different programmable language (e.g. Java, C#, etc.) with a respectively integrated development environment (IDE) to be used in the customization of the events needed. The presented CEPs are in the same situation, being all of them developed in Eclipse, and in the case of the WSO2 and FITMAN use Esper's engine as the core, gaining with that more functionalities and in some cases better performance. However, in terms of support Esper appears to have a larger community. Being Esper a pure JAVA and .NET and the event queries are similar to

[11] http://www.espertech.com/
[12] http://wso2.com/products/complex-event-processor/
[13] http://catalogue.fitman.atosresearch.eu/enablers/dynamiccep

Structured Query Language (SQL) language opposed to the others where you may have to learn new languages for programming them.

In Table 3-2 a comparison between the different presented CEP is made. The comparison is made through eight different features, more can be added, yet these ones were selected considering their relevance for the CEP to be chosen for this work. The form of reading the table is equal to the description made into the Table 2-1, to check if appears a right or a wrong, or with a description. In the case of appearing a right represents an existing feature in that CEP. In the case of appearing a wrong means the opposite. The description means that this feature exists and gives a short detail about it. The table determines different characteristics to give a better understanding of the potential of each CEP. The first feature determines if the software is an open source or not, the second and third shown if the software has community support and technical support (since with the community or technical support facilitates the development of the tool). The next feature is to identify if the software has easy integration, in this situation is determined if the software have a free way to program. The Dashboard is a feature that facilitates de monitor of the integration of the software, allowing knowing the state of each process/goal. The Latency is the time interval between the events (measured in microseconds per event), and the Scalability is the capacity of the CEP to handle a growing number of events (measured in events per second). The Recovery option is the capacity for the system to recover from a forced shutdown and restores to an earlier point in time. For example, the Esper CEP is open source, have a community support and easy integration to their advantage. On the other hand, it has a lack of their technical support, it has a dashboard to follow the events running and allows to control them. It has a latency of 10 µs per event and a scalability of 10K to 200K events per second, it allows for recover the CEP every time that a bad interruption occurs in the system. This form of reading the table works for the rest of the CEPs.

Table 3-2: CEP Engines Comparison.

	Esper CEP	WSO2 CEP	FITMAN Dynamic CEP
Open Source	✓	✓	✓
Community Support	✓	✗	✗
Technical Support	✗	✓	✗
Easy Integration	✓	✗	✗
Dashboard	✓	✓	✓
Latency	10µs	17µs	+10µs
Scalability	10 to 200K	+100K	+10 to 200K
Recovery	✓	✓	✗

Every CEP engine that is listed is open-source, however, the support varies among the CEP engines. Esper CEP has large community support with very little technical support, though it's the easiest to integrate into projects. WSO2 CEP has very little community support but it's possible to get direct support from WSO2 to answer any questions. Integration is a bit more complex than the other engines because the WSO2 CEP is built to be easily integrated into other WSO2 products and no other general projects. FITMAN CEP has a largely detailed documentation explaining the use of their CEP but other than that it's difficult to find more support, either if it's direct technical or community support.

Another option to monitor is through the use of an agent, since it have the advantages of being: **Autonomy** (Agents make decisions without human intervention, these decisions are made with some kind of control over their actions and internal state), **Social ability** (Agents interact with other agents via an agent communication language), **Reactivity** (Agents react to changes in their environment), and **Pro-Activeness** (Agents have their own goals and besides reacting, they are capable of initiative) (Wooldridge et al., 1995). Taking these descriptions into consideration, especially the features of social ability and reactivity, it is common that the agent is capable of interacting with other agents, humans, or with the surrounding environment. This brings something new to the software technologies, i.e., communication and teamwork between software, in this case, between agents, and this is called a Multi-Agent System (MAS) (Bellifemine, Caire, Greenwood, et al., 2007).

The great advantage of using MAS for the implementation of this work, is that they are capable of cooperation, collaboration, negotiation, etc., and they understand each other via an agent communication language based on the act of speech (Bellifemine, Caire, Greenwood, et al., 2007), thus avoiding agents' interoperability issues. Therefore, due to these features, MAS is being used in different areas, from industrial applications to telecommunication and multi-robotic systems.

Based on the advantages of MAS, especially the capacity of working with different agents and collaborating with each other, MAS is being used to monitor systems and networks by collecting data and sending them to the monitoring station agent. The main agent will make requests from the monitor and other agents answer depending on the data collected. This makes the agent "lightweight" and able to access data for better monitoring, alerting, and reporting, and in some cases allowing a root-cause analysis and troubleshooting. Normally, this monitoring is made at predefined intervals, in which the monitor agent communicates with the station whenever agreed. The alerts are then generated depending on whether the pre-defined KPIs where exceeded (UptimeSoftware, 2013).

3.4. Analysis of State of the Art towards Hypothesis

Adaptive systems theory is taken as a basis for this work, considering that EIS collaborative networks are a complex macroscopic collection' of relatively similar and partially connected micro-structures, formed in order to adapt to the changing environment and increase its survivability as a macro-structure. It is particularly difficult to maintain the interoperability in such enterprise collaborative network. This is mainly because of how EIS systems are designed since each system usually has different information models and interfaces.

To assure the sustainability of interoperability in a network of EIS, it is necessary to create seamless relations between the heterogeneous systems, its standards, and models. In this chapter, it was made a study on several EIS modelling paradigms, to identify how they work and how they represent the enterprises and their own resources, enabling the reuse of proven methodologies and systems' models in a semi-automatic generation of software adapters to enable SI (in which it will be discussed in later chapters).

All the interoperability layers and maturity levels mentioned above have the purpose of evaluating the interoperability status of an organization or specific system or network (of enterprises or devices). It is important to know, in terms of interoperability, the position that an enterprise is in, to allow them to improve and become more efficient when exchanging information with the business partners. To this end, a study was carried out on several assessment methods that support the interoperability layers and maturity levels in the evaluation of the EIS and as a future work, it is needed to identify how these assessment methods will behave with a network of devices.

The interoperability layers and maturity levels will be used as the classification of the enterprise and the assessment methods will support them by giving an evaluation to them. By integrating these two systems it will become possible to know the interoperability state of each enterprise. Then, according to their evaluation, the system will react and respond to three possible solutions:

- **The network is not interoperable** – In this situation, the system must react and propose a solution to create an interoperability environment in the network. It will identify measures and execute them or propose them to the user;
- **The network is interoperable** – The system identifies a level of interoperability and whether or not it is possible to improve the interoperability in the network. Then it will take measures to create a self-sustainable interoperability network;
- **Self-sustainable interoperable network** – In this situation, the network is already self-sustainable interoperable, so the system needs only to continue the maintenance of the interoperability level, in order to avoid problems in the future by creating a self-adaptive system to allow the system to adapt over time.

To support this idea several technologies and methodologies are needed, one of which is to implement the CAS-SIF philosophy in the system, which will give guidelines to reach a self-sustainable interoperable network. Another thing is to use MAS and CEP to monitor and assess the network and its different systems independently.

The importance attributed to MAS in this work is due to the description made by Wooldridge (Wooldridge 2009), in which he says that decentralized multi-agents are considered to be an added value in the monitoring services implementation, assuring organizations' independence. This is an important advantage because the main point is to create a self-sustainable interoperability framework for the network. Also, MAS support interoperability by formalizing the communications. They communicate across multiple systems using a standard protocol created by FIPA (Foundation for Intelligent Physical Agents[14]) to assure that services and the ontology are common between the systems. Another advantage is that MAS can be implemented on a device level, so it will be possible to monitor a network of devices.

Another technology that can be used to monitor is the CEP. Comparing a CEP with a database, it can be said that is an inverted database, since database stores information and processes events in an ad-oc way, instead, a CEP stores queries and triggers events as information is

[14] http://www.fipa.org/

added. The queries are defined using dedicated query languages styles, which patterns are defined in order to form relations between events. This feature of CEP complements this work, by monitoring the KPIs to identify changes in the network and detect possible problems in the devices and systems, since each device or system has their own role in the network. The CEP with the support of services can warn the user and monitor the devices about occurrences that are happening, respond to the KPIs pre-defined (Weidlich et al., 2011).

This chapter supported to prepare the answer to four research questions presented in Chapter 1.3.2. The second question is studied in this chapter with the study of the interoperability maturity and evaluation strategies, these two topics studied to aim to select one or more options to support to identify and measure systems interoperability, being the study made to be used in the interoperability assessment module of the framework. The third question is studied in this chapter with the study of the monitoring paradigm and evaluation strategies. The monitoring parading is needed in this work to monitor the enterprise network to identify situations that can cause harmonization breakings, for that it is needing to identify the best or the best solutions to use, enabling to reach the sustainable interoperability in the network. In the case of the evaluation, strategies need to identify the interoperability issues in the enterprise, to allow the framework to know what it is needed to create the interoperability. The fourth question is related to the last question, to know how the system to react to poor integration, it needs to monitor and to know the level of interoperability of the enterprise.

4. Framework to Enable IoT in a Sustainable Interoperable Environment

With the market and the growth of the internet globalization, enterprises have found an opportunity to produce greater amounts of products, achieving better production times and more competitive prices. To achieve these objectives, enterprises create collaborative networks, in which each produces a part of the product, and be anywhere in the world. Yet, such networks are not static but evolve over time (meaning that the networks are dynamic) (Staudt et al., 2012).

This dynamism is due to several factors, as described in Chapter 2.3.3, affecting other members of the network to the point that they must react and adapt to changes, evolving and avoiding that a harmonization breaking occurs in the network. This work aims to facilitate a network of enterprises to maintain interoperability. This maintenance is done after the framework identifies interoperability in each enterprise, thus being able to know the level of interoperability at each enterprise and what can be interoperable in the network.

When assessing interoperability in every two enterprises, it is possible to identify the interoperability level of the network and thus enhance interoperability between the whole network (as described in Figure 1-2). Then it is intended that the network operates in harmony without experience more problems during cooperation. For that to happen, the network is going to be constantly monitored to detect changes and solve them (if required) in time to avoid problems in the harmony of the network. With this, a new step is attained, the Sustainable Interoperable Network, which is the goal of this work.

4.1. Towards a Sustainable Interoperability in IoT Systems

In Chapter 2.3.3 a study in sustainable interoperability was presented, defining the concept of sustainable interoperability and identifying problems that occur every time an enterprise wants to cooperate with another. It is considered that EIS, models, and devices are IoT systems, hence in an IoT environment, systems and devices must communicate between them without causing problems in interoperability.

The study made in Chapter 2.3.3, provides a better understanding of what can happen in an enterprise collaboration, supporting to the development of the framework, since it gives a vision of the requirements that need to be fulfilled, the framework was called Self-Sustainable Interoperability Framework (SSIF). For this reason, for the SSIF framework to support a sustainable interoperability environment it needs to consider that different legacy systems and devices exist, the enterprises are distributed around the world. Since, every time that one of these problems happens (e.g. a message structure changed, a device failed), it creates a harmonization breaking in the network, which forces the network to fail to collaborate.

For these reasons, the sustainable interoperability is ideal since it allows an enterprise to adapt their EIS and devices, and still maintain the interoperability along its operating life cycle. However, to reach this status is needed to identify requirements that will give the possibility to fulfill the desired goal. To have a sustainable interoperable environment, it is important to understand the modeling paradigm in use, the relationship among the business partners so that intelligent reconfiguration of components becomes possible. To create this maintenance system and manage such dynamics, it is necessary to monitor and adapt to the changes while learning over time. Nevertheless, in another perspective, it is needed to predict the transient that results from the dynamics of the individual systems since a network (and network of networks) will face changes that impact third parties in the global operative environment. Hence, an evolution of a system should only be decided in case it brings more benefits than damages. According to the reflectivity principle (Honour, 2008), changes can follow a cyclic loop that impacts the same system that motivated the initial evolution.

Another feature to consider is to have a conformance test and interoperability checking for systems interoperability assessment (already discussed in Chapter 3.1). This assessment is used to discover and notify every time that a new system node is integrated into the collaborative network, or it is updated. The conformance checking is required to check for conformance of data, models, knowledge, and behaviors of the systems and assure accuracy in the seamless communication. The interoperability checking will verify and assess the network to assure the maintenance of the network interoperability system (R. Jardim-Goncalves et al., 2011), (Vernadat, 2001).

The SSIF framework here proposed was inspired in the CAS-SIF (Agostinho & Jardim-Goncalves, 2009), which proposes the system a capacity for self-recognition in the face of environmental adversities, by detecting changes (of different types) on time, it can avoid problems in near future, see Figure 2-15. This work implements and improves all the cycle of the CAS-SIF, starting with the monitoring (detection of the harmonization breaking) and finishing in the communication with the network, as illustrated in Figure 4-1.

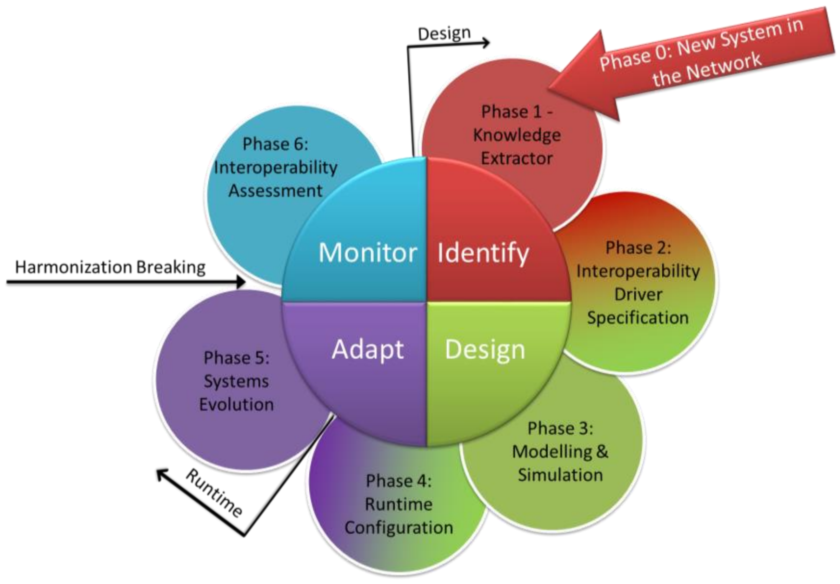

Design
Phase 0: New System in the Network
Phase 1 - Knowledge Extractor
Phase 6: Interoperability Assessment
Monitor
Identify
Harmonization Breaking
Phase 2: Interoperability Driver Specification
Adapt
Design
Phase 5: Systems Evolution
Phase 3: Modelling & Simulation
Phase 4: Runtime Configuration
Runtime

Figure 4-1: Self-Sustainable Interoperability Framework (SSIF).

It makes sense to start with the identification of the models in use by the enterprise, starting the cycle in the Knowledge Extractor Step (called Phase 1 as illustrated in the figure). When there is a New System in the Network there is a Phase 0 that is executed only once and is out of the cycle. Throughout the cycle, there are three milestones that influence state transitions. The first two are the modes of Design and Runtime, which are responsible for identifying the phases that make the specification of interoperability and the phase of monitoring and identifying problems that occur in the network. The other state is when a harmonization breaking occurs, that is where it is identified a problem that occurs in the network. Because of this study, the author proposes (in the center of Figure 4-1) a cycle divided into four Steps (the outside phases are explained in Chapter 5, Chapter 6 and Chapter 7):

- **New System in the Network** – This is the step responsible for the pre-configuration of an enterprise, and consequently, of the network, it is out of the cycle as a preparatory step. Being divided into two objectives, it allows the enterprise to register its own data, and at the same time their resources (to be used in the sustainable cycle). After insertion of all the data, it is possible for the enterprise to define its collaboration network. At the end of this part, the enterprise is ready to enter into the sustainable cycle;
- **Identify** – This is the phase of identification of the models and devices, in here it is possible for the SSIF framework to identify similarities between the different models and devices, to enable the creation of mappings. Since each model/device has their differences, it is needed to search for similar concepts between them to enable the identification of relations. This feature allows the identification of similar concepts to facilitate the discovery of patterns between the models and devices. After that, the interoperability driver specified based on the semantic mappings purposed by the system;
- **Design** – With the drivers specified and the enterprise network defined, the design step starts. This step allows the enterprise to design new services for new products, having as a base the defined resources. To support the validation of the processes a simulator exists, simulating device failures or verifying the impact that changes in the models have on the mappings;
- **Adapt** – This step is related to the beginning of the runtime mode, this happens when the drivers are connected and configured. Then, the system modifications are made, and to enable the sustainability of the network, a notification to the network is made, warning about potential problems that can have impact in other enterprises;
- **Monitor** – This step is responsible for monitoring the network, searching for problems, e.g. identify a device malfunctioning, service that is down, a possible problem in a mapping, etc. In a situation that a problem occurs, it is needed to analyze whether this situation is really a harmonization breaking to be able to signal as a problem and notify the identification phase. An interoperability evaluation is made by enabling to verify the interoperability level which enables the sustainability of the network.

As a result of the framework described in Figure 4-1, the methodology was developed to provide support, which is represented in Figure 4-2. Given the complexity of this methodology, each phase is described in the next chapter.

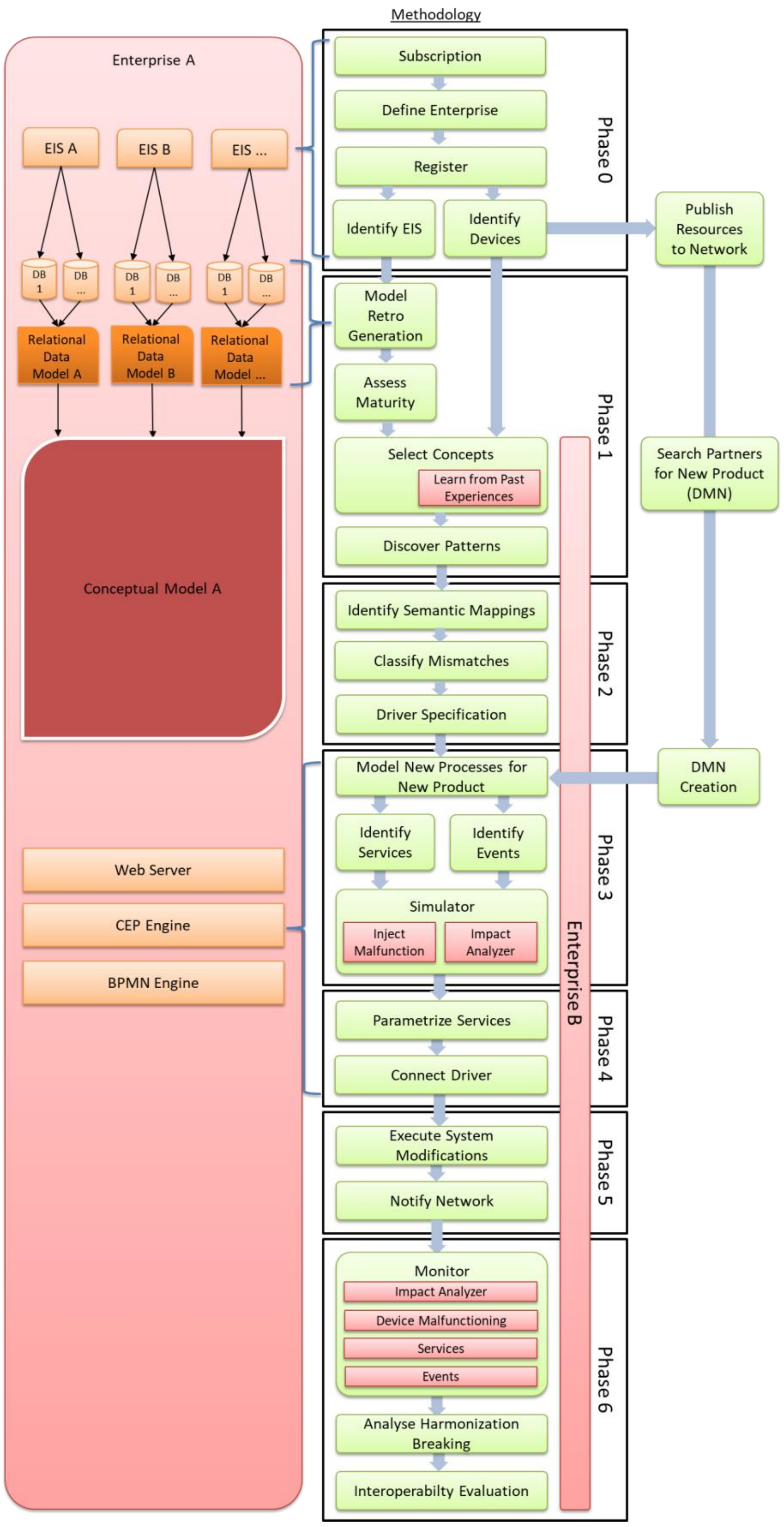


Figure 4-2: Methodology for a Sustainable IoT Interoperable Network.

4.1.1. EIS as Self-Adaptive Systems

Adaptive Systems are those that react to change in their environment and adapt to these changes by changing their behavior (Hveding 2008). This adaptation is needed because more and more software systems are used on, or access, a variety of handled networked devices or systems used by people around the world. This generates considerable and unpredictable dynamic variation in enterprise networks, which will lead to deadlock in the integration. For this reason, a dynamic adaptation is required in order to retain usability, usefulness, and reliability of the application under such circumstances (Hallsteinsen et al., 2004). As explained in this dissertation, large and rapid growth in the software systems and devices has brought great diversity to the market, complicating the integration between enterprises. For that, they need to adapt themselves to the difference between the systems and to their failures. Such systems increasingly tend to be long-lived, decentralized, heterogeneous, mobile, and ubiquitous. Their adaptation, therefore, cannot be effectively directed and controlled solely by a central source. Instead, these systems are expected to emerge, exhibiting some degree of self-awareness and self-adaptivity (Edwards et al., 2009). Indeed EIS self-adaptation will bring an enormous advantage to them (Metzger, 2009):

- **Ensuring the interoperability** – While standardization in web services makes interoperability easier, adaptation remains necessary. Adaptation is an important functionality that should be offered to enable open innovation and integration across enterprise boundaries;
- **Optimization** – The demand for quickly delivering new applications and adaptively managing them is a business imperative today. The quality of service offerings in an application may change, new service providers and business relationships may emerge, and existing ones may be modified or terminated;
- **Recovery** – Various faults can occur relatively often and unexpectedly in distributed systems. It is necessary to handle faults reported during execution of the component instance when monitoring business processes. In fact, web services compositions often implement business critical processes whose correct and uninterrupted operation is paramount;
- **Context change** – Services are made up of reusable software components. The adaptation's goal is to optimize the service function within their execution context. It searches and applies the viable solutions: component customization, insertion, extraction, or replacement.

An example of a self-adaptive system was presented in IBM's MAPE-K Autonomic Loop (Huebscher & Mccann, 2008), (IBM, 2003). In this model, a simple adaptive system was developed, in which every time a change occurs in the environment, the system detects it through "sensors" and creates events to warn the local sub-systems. This event activates an analysis block, looking for changes with the aim of knowing what kind of impact it has on the environment. When harmonization breaking occurs, planning services are called upon to find a solution to recover from it, by proposing a new Plan, which is to the executed after.

Other authors propose complementary models and strategies. One example is the adaptive system developed on the FAMOUS project (Hveding, 2008) for adaptive mobile services. Another one is the ADASIM Simulator (Wuttke et al., 2012) developed for evaluating and comparing automated traffic routing problem solutions. Also, CAS-SIF (already presented in Chapter 2.3.3.1) is a relevant framework proposed by (Agostinho & Jardim-Goncalves, 2009). This framework introduced the notion of sustainable interoperability and proposes a solution to enable it following the principles of complex adaptive systems. It seeks to adapt the integration of the different systems by creating a bridge between their inner elements, facilitating the communication and cooperation inside the network (seen as the macro system). CAS-SIF is adopted as a reference for this work.

4.1.2. Self-Adaptation of the Sensing Enterprise Devices

In Chapter 2.3.3.2 a discussion about Sensing Enterprise was made, in a case that an enterprise can implement the SE concept in their infrastructure, giving the capability of sensing through devices and reacting to stimuli. This is reached by integrating a decentralized intelligence, context awareness, dynamic configurability, and IoT sensorial technology into its decision-making process. The implementation of these features brings huge advantage, while on the other hand brings more complexity to enterprises. An enterprise is a complex system and SE makes it even more complex. From an SE perspective, a self-adaptation system is ideal, with benefits such as:

- **Context awareness –** To receive real-time information about its environment, is needed a considerable number of devices. Monitoring a considerable number of devices is not an easy task, involving several types of issues, as big data, the devices itself, interoperability of the data structure or semantics. In a situation where a device breaks down, the system needs to adapt to these new conditions. As an example, let’s consider the following: a device malfunction, and there is no new one for change, so it may have to use a different one. Since this new device may have a different data model from the previous one, it causes the system to have to adapt to these new conditions and to continue to work without problems;
- **Dynamic configurability –** A necessary requirement is to be adaptive, since for a dynamic configurability the systems requires a way to configure a new device and adapt the systems and devices network for the new change, without compromising the rest of the network;
- **Multi-identity enterprises and devices –** In this situation a new concept is created inside of an enterprise, the notion of physical, virtual and digital worlds. Although they are concepts that already exist in enterprises, are not well defined. It is necessary to define limits, causing information coming from systems and devices to change worlds. An adaptive system is an added value to the enterprise, recognizing the change of the world where the information is and adapt itself;
- **Relationships between humans and objects –** Situations where humans and objects must cooperate with each other, create an inconvenient situation to manage. Given that the human works and thinks in a way (each human is different from the other), the object must

know how to react to that stimuli. In these terms, the system needs to constantly adapt to the environment.

SE is a recent concept that was created in the advent of the Augmented Internet, as an attempt to reconcile traditional non-native “Internet-friendly” organizations with the tremendous possibilities offered by the cyber worlds (Danila et al., 2013). For this reason, some topics are still not very explored, turning it difficult to find in the literature others works in this area. Since the enterprise is a network of devices that interact with each other to give a context awareness of what is occurring in the network, then it can be said that a CPS is used within the enterprise. Looking at solutions in the context of CPS, in (Muccini et al., 2016) a study was made on adaptive systems for CPS. In this study, the authors say that the top three concerns related to the self-adaption in CPS area are efficiency/performance (66%), in second is the flexibility (48%), in third is the reliability (24%). Interoperability appears in the sixth position with 5%, demonstrating that the self-adaptive area in CPS is being developed, but concerning the interoperability is not yet a focus. Thus, this fact presents an opportunity for this work.

4.2. Architecture for a sustainable IoT Interoperable Network

This work seeks to contribute to the main research question, which is the identification of the EIS and models in use by the enterprises and a system to monitor and evaluate the models implemented in an enterprise. This will enable the creation of an interoperable environment inside of an EC through an SSIF framework that is able to identify the EIS, models, and standards in use by the enterprise, and to use that new knowledge to create a bridge that will allow a sustainable integration between other enterprises.

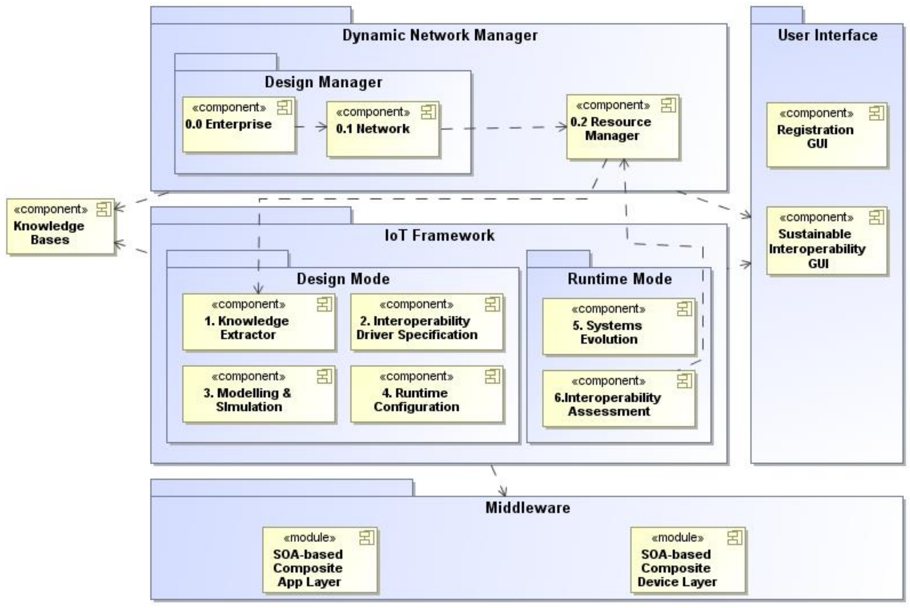

Dynamic Network Manager
Design Manager
«component» 0.0 Enterprise
«component» 0.1 Network
«component» 0.2 Resource Manager
User Interface
«component» Registration GUI
«component» Sustainable Interoperability GUI
«component» Knowledge Bases
IoT Framework
Design Mode
«component» 1. Knowledge Extractor
«component» 2. Interoperability Driver Specification
«component» 3. Modelling & Simulation
«component» 4. Runtime Configuration
Runtime Mode
«component» 5. Systems Evolution
«component» 6.Interoperability Assessment
Middleware
«module» SOA-based Composite App Layer
«module» SOA-based Composite Device Layer

Figure 4-3: Self-Sustainable Interoperability Architecture.

Based on the analysis of the state-of-the-art presented in the previous chapters, the SSIF framework will be composed of different modules that contribute with different targets to reach the final goal (as illustrated in Figure 4-3), establishing an interoperable environment capable of avoiding and reacting rapidly to harmonization breaking in the network. This SSIF framework will give freedom to enterprises, to work and evolve normally according to their business requirements enabling an automatic negotiation between their systems. This measure will allow for the seamless exchange of information between them via the bridge constructed.

In the complementary context of the physical sensing enterprise, if a user wants to use devices (e.g. sensors) of different providers, similar interoperability and sustainability problems are posed. Currently, each manufacturer produces devices using their own standards and rules, which can, therefore, be addressed by the framework proposed.

The evaluation and monitoring of systems will give the capability to identify changes that can cause problems to the enterprise or perhaps to the network. In case of being a harmonization breaking, the system will adapt to change and learn to avoid a repetition in the future.

With these measures, it will be possible to implement a self-sustainable interoperable environment on the enterprise’s network, which is the main point of this work. To keep the privacy of each enterprise, the system will be locally stored at each site and distributed through the network to emulate a centralized monitoring. The proposed architecture is divided into five main blocks:

- **Knowledge Bases –** During this work, it was identified the need of having knowledge bases (KB) to store the different information used. It stores the mappings that relate the different models and devices, to enable the maintenance of the interoperability of the enterprises. It also manages the registration of the resources of each enterprise and supports the process design. This block is detailed in Chapter 4.3;
- **Dynamic Network Manager –** This is responsible for making the registration of enterprises, each enterprise can register their data, as well as their resources (models, devices, events, services, and processes). After this registration, the component saves it in a KB to be processed and used by other components. This block is detailed in Chapter 5;
- **IoT Framework –** It is the block that performs the SSIF framework explained in Chapter 2.3.3.1, following the four steps described in the cycle (Identity, Design, Adapt and Monitor). This block consists of the phases (represented by the external circles of Figure 4-1) and which are responsible for maintaining the interoperability. To better describe this block it was divided into two chapters, one chapter to detail the cycle when it is in Design Mode (Chapter 6) and another one to detail the cycle when it is in Runtime Mode (Chapter 7);
- **Middleware –** This module is responsible for making the connection between the enterprise and the enterprise’ network and at the same time the connection with their own device networks. The devices and the applications need to be connected and enabled to communicate with other devices/applications/modules/enterprises;

- **User Interface –** This is the user interface that allows interaction with various components that the framework provides. It contains the GUI for the users insert the enterprise' information, design the processes, monitor the network, etc. Hence, enabling the user to follow the status of the network enterprises and being able to act in solving the problems that appear.

4.3. Knowledge Bases

This chapter describes the four knowledge bases (KB) used in this work, divided into two objectives, two of which represent enterprise information and the other two represent the mappings. Among them there is interaction, the KB of mappings is linked to the KB of information, to cross-data between the various enterprises. In resume, the Communicator Mediator for EIS interacts with the Enterprise Gateway Knowledge Base to relate the information between the different enterprises, and the Semantic Mapping Knowledge Base interacts with the Device Knowledge Base to relate the different devices inside the network.

4.3.1. Enterprise Gateway Knowledge Base

With the aim of integrating different enterprises in the SSIF framework, since each enterprise has its own nomenclature, for this was developed the Enterprise Gateway Knowledge Base represented in Figure 4-4. Such action intends to standardize the categories used in the platform, harmonize searching criteria to describe the business of each enterprise, e.g. market sector, material type, and to classify the type of material or product.

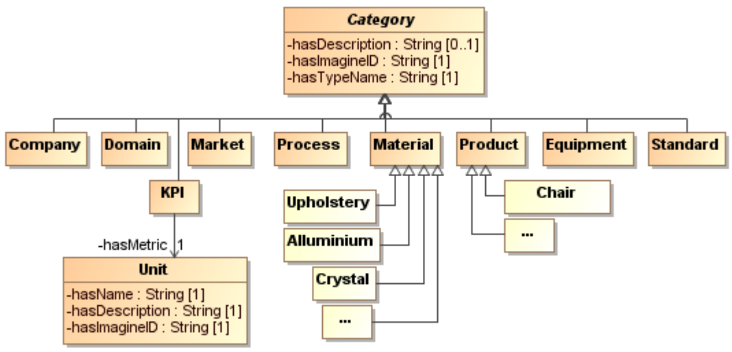


Figure 4-4: Excerpts of the Ontology Model for the Categorization of Enterprise Information.

One abstract class and some main classes compose this model. In more detail: Category is an abstract class, "parent" to all types of categories. It is used to manage the relationship with the platform BP knowledge base (through an "Imagine ID" attribute); Company is a sub-class of "Category" to identify the type of activity of each enterprise (e.g. manufacturer, retailer); Domain is a sub-class of "Category" to identify the working domain (e.g. furniture); Market is a sub-class of "Category" to describe the type of market that the enterprise can reach (e.g. National); Process is a sub-class of "Category" to represent the type of processes provided by enterprises (e.g. cutting); Material is a sub-class of "Category" to represent the material that each enterprise processes (e.g. wood, plastic); Equipment is a sub-class of "Category" to represent the equipment that each enterprise has in their facilities to execute processes (e.g. cutting machine); Standard is a sub-class of "Category" to indicate the standards in use; KPI is a sub-class of "Category" to maintain a list of standardized key performance indicators (KPI). These can have metrics to enable subsequent evaluation; Unit is a class to represent measuring units (e.g. meters).

4.3.2. Device Knowledge Base

For the SSIF framework to be able to monitor the device network, it needs to know which devices belong to the network. For that, a registration of the devices is needed and was developed a knowledge base that keeps track of them. In addition, it can identify the services and events that each device uses to support the monitoring, improving this feature of the SSIF framework. The implementation of the Device Model was based on IoT-A project's reference model, delivering a high level of abstraction. The choice of this solution was based on the study done in Chapter 2.2.

Whenever you want to add a device, the SSIF framework queries the KB with the main purpose to reuse information to be instantiated in the KB, therefore when adding new devices, the KB is consulted to get the device's relevant information (properties, used services, and events). The Device Model revolves around three key entities: Virtual, Service and Event. The Virtual entity is a virtual representation of the physical device, creating a virtualization of the physical device, relating the services and events that are specific to the characteristics of the device. A Service provides information about Virtual entities that can be used to control the device, creating impact in the real world. Finally, the Event is responsible for triggering actions in answer to the data retrieved from the devices. The model can be split into four sub-models as Figure 4-5 shows:

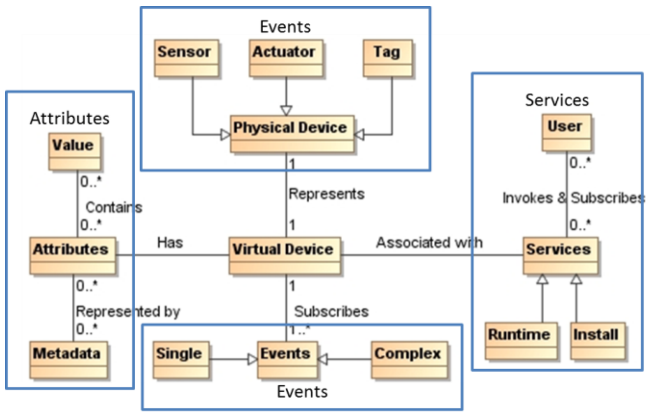


Figure 4-5: Structure of Device Model.

- **General Device Properties –** Each device connected is unique, however, there are some properties or characteristics that are common and can be mapped directly. Properties such as model number, type, name, location, etc.; these characteristics facilitate the insertion of a new device. As Figure 4-5 shows each device is mapped as a virtual entity no matter what type of device it may be. The virtual entity is a digital representation of the physical device itself as is shown through the Physical entity relationship. The physical entity can be classified according to the defined topology: Actuator, Tag or Sensor;

- **Service –** This entity is responsible for providing information relevant to service usage. Each device may contain a set of services that are available exposing information relevant to an operation. This includes a description, a service Uniform Resource Locator (URL) for access. In addition, when events are triggered, they may execute, depending on the rule creation inside the CEP, an action exposed by a service. It can be divided into two parts, the InstallService (services used to register a device in the model) and the RunTimeService (services used after the registration of the device is completed);
- **Attributes –** To be able to map non-general device properties and to store measurement value and meta-information such as quality of information or in what unit the measured value occurred, provides a way to map any kind of device property from a wide heterogeneous world of devices;
- **Event –** The basic representation of an event that is triggered, according to the pre-set rule conditions, that contains all the relevant information to describe the event and increase contextual awareness. When an event is triggered, the necessary consequential actions are executed through services, as Figure 4-5 shows. The triggered event can derive either from a simple event or a complex event.

4.3.3. Communication Mediator for EIS

One of the needs of this work, it is the need to represent the mappings between the different models of EIS, one way to achieve this situation is to save this mapping in a knowledge base. With this, every mapping between models or ontologies of business partners can be stored and accessed by their local systems. This allows communities to build systems with reasoning capabilities able to understand each other's representation format, without having to change their data and schema import or export processes (Sarraipa et al., 2010). This generates a novel approach to the network, by giving the capacity for each partner to manage their own mappings avoiding the centralization of the data. Yet, for that all the business partners in a collaborative network should have a KB in their local system, to act as a mediator for information exchange (Agostinho et al., 2011).

The proposed knowledge-based is called Communication Mediator (CM) and is defined by ontology in OWL format. The knowledge is stored according to the tuple format, proposed in Equation 4-1. Agostinho et al. (Agostinho et al., 2011) propose the usage of a 5-tuple mapping expression, with the objective to consolidate existent approaches to morphisms:

MapT = <*ID, MElems, KMType, MatchClass, Exp*> Equation 4-1

- **ID** is the unique identifier of the MapT;
- **MElems** is the pair (a, b) that indicates the mapped elements in the source and destination models;
- **KMType** stands for Knowledge Mapping Type and is used to identify the morphism as "Conceptual" if mapping concepts or terms; "Structural" if mapping model schemas; or "InstantiableData" if the mapping instantiable properties;

- **MatchClass** stands for Match/Mismatch Classification and is used to classify with reference data, knowledge about the mapping mismatches, i.e., inconsistencies of information that can appear when a mapping between two models is created, derived from the multiple conflicts between the entities, the different mismatches are illustrated in Table 4-1;
- **Exp** stands for the mapping expression that translates and further specifies the previous tuple components.

The structure of the CM is presented in Figure 4-6 and described as follows: the CM has two main classes: “Object” and “Morphism”. The “Object” represents any “InformationModel” (IM) which is the model/ontology itself and “ModelElements” (also belonging to the IM) that can either be classes, properties or instances. The “Morphism” associates a pair of “Objects” (related and relating) and classifies their relationship with a “MorphismType”, “KnowledgeMappingType” (if the morphism is a mapping), and “Match/Mismatch” class. The “Morphism” is also prepared to store transformation oriented “ExecutableCode” that will be written in the ATLAS Transformation Language (ATL)[15] and can be used by several organizations to automatically transform and exchange data with their business partners as envisaged before (Filipe Correia, 2010).

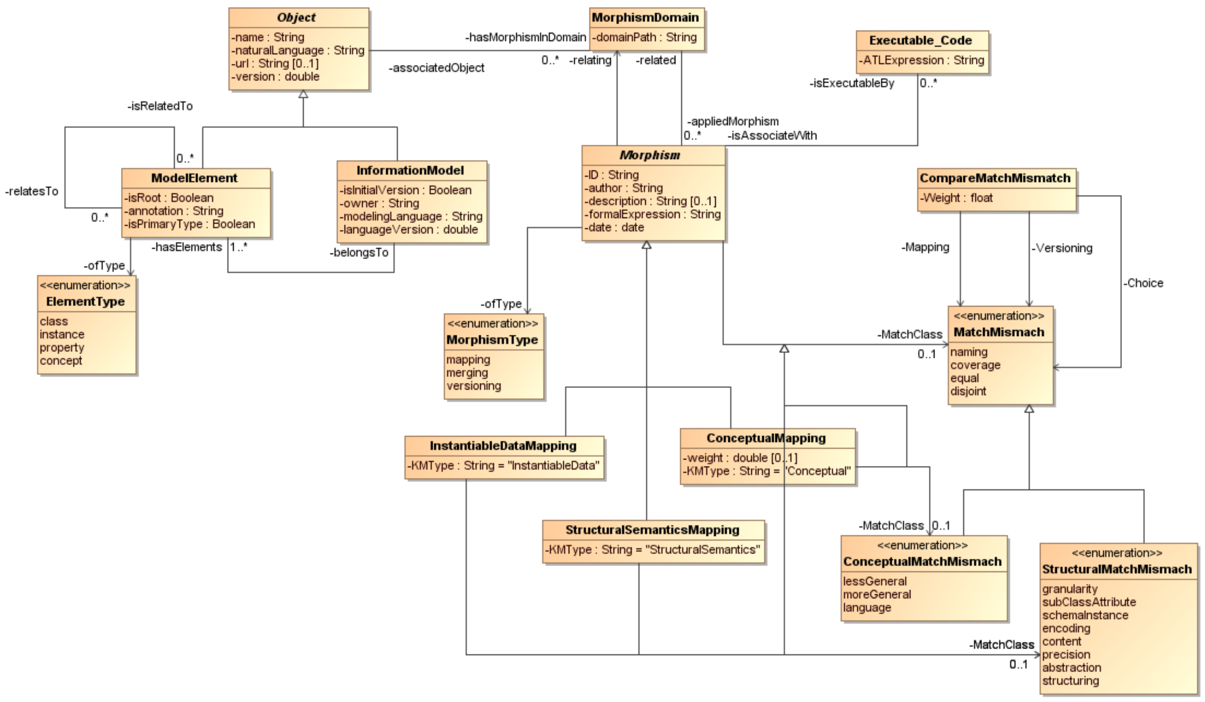
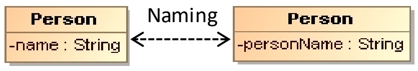
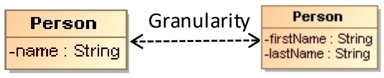
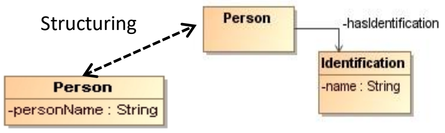
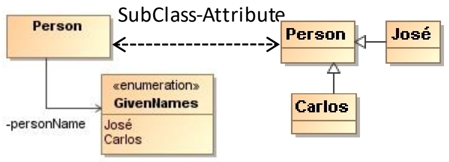
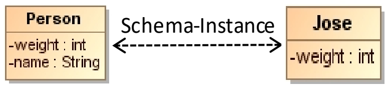
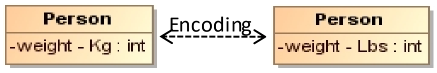
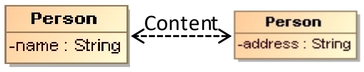
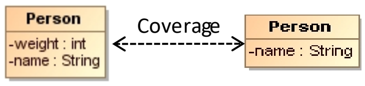
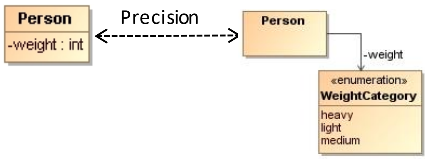
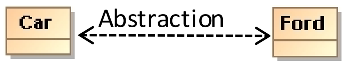


Figure 4-6: Structure of Communication Mediator (Sarraipa et al., 2010).

When a mapping is created between two models, sometimes some inconsistencies of information will appear derived from the multiple conflicts between entities. Those are called Semantic Mismatches and can be classified as either lossy or lossless, as shown in Table 4-1.

[15] https://eclipse.org/atl/

Table 4-1: Semantic Mismatches (based on (Agostinho et al., 2011)).

Mismatch		Description	Examples
Lossless	Naming	Different labels for the same concept of structure	 Person -name : String Naming Person -personName : String
	Granularity	Same information decomposed in or composed by (sub)attributes	 Person -name : String Granularity Person -firstName : String -lastName : String
	Structuring	Different design structures for the same information	 Structuring Person -hasIdentification Identification -name : String Person -personName : String
	SubClass-Attribute	An attribute, with a predefined value set represented by a subclass hierarchy (or vice-versa)	 Person SubClass-Attribute Person José «enumeration» GivenNames José Carlos -personName Carlos
	Schema-Instance	An attribute value in one model can be a part of the other's model schema (or vice-versa)	 Person -weight : int -name : String Schema-Instance Jose -weight : int
	Encoding	Different formats of data or units of measure	 Person -weight - Kg : int Encoding Person -weight - Lbs : int
Lossy	Content	Different content denoted by the same concept	 Person -name : String Content Person -address : String
	Coverage	Absence of information	 Person -weight : int -name : String Coverage Person -name : String
	Precision	Accuracy of information	 Person -weight : int Precision Person -weight «enumeration» WeightCategory heavy light medium
	Abstraction	Level of specialization	

4.3.4. Semantic Mapping for Devices

As in the case of data models, problems can also occur in devices, for example, a device is damaged by unknown reasons and provides unusable information (e.g., temperature readings out of predictable thresholds or an anomaly detected by comparison with nearby devices). To tackle this situation, human intervention is usually needed. In large industrial networks or in the IoT paradigm, there are numerous sensors, deployed for different uses, which may be suited to comply with operations related to the sensor that is malfunctioning. To measure that possibility, sensors can be analyzed regarding aspects like localization, type of measurement, role in the network and others. In this case, it is possible to find a different sensor that serves the same purpose and is suitable to replace the specific malfunctioning sensor in rules associated with it, as well as other similar sensors that measure temperature.

In these situations, it is possible for a human responsible for maintenance to change the faulty sensor to a new sensor. However, this may take a while, one solution is for the system to propose a sensor that is active and that can replace what does not, using a mechanism that provides autonomous dynamic adaptation. This is where semantic mapping comes to play because it represents and maps possible redundancies while providing room for developing the autonomous creation of possible maps regarding the aspects of the role of each sensor in the network. This autonomous creation is possible, but it is advisable to create the initial configuration of the existing devices and relevant possible core semantic maps by the developer, to provide consistency to the network.

For this reason, a mapping has been developed to specify possible sensors that can replace one that stops working. During this study, it was needed to understand what specifications the semantic maps need to distinguish in the sensors. These specifications are considered by relevant fields summarized in the tuple represented in Equation 4-2.

$$\text{Map} = \langle \mathit{ID}(\text{O}, \text{D[i]}), \mathit{Mismatch}, \mathit{Role}, \mathit{Correlation}, \mathit{Weight} \rangle \qquad \text{Equation 4-2}$$

The following aspects were considered to form the mapping tuple:

- **ID (Identifiers) –** Represents the individual unique attribute that differentiates each instance. The letter “O” defines the origin sensor and the letter “D” defines the destination sensor(s);
- **Mismatch –** Type of relation or association is a representation of the difference between each instance in a way that highlights the aspects that may have to be considered while mapping. This information is complemented in a mathematical way by the correlation filed. Refer to (Agostinho et al., 2011) for possible mismatch examples (Table 4-1);
- **Role –** The role of the original instance is very important, perhaps the main factor to be considered, because it represents each functionality that the origin sensor has in the network (e.g. measurement to infer the existence of a fire), i.e. every function in the network that needs to be mapped (one map for each). After the map is used, the destination sensor(s) add the new role to their device information;
- **Correlation –** Output data differences between sensors, is the relation between the output that each sensor may have, regarding its specifications (e.g. voltage output) and information to be considered when analyzing the data (e.g. average between measures);
- **Weight –** The importance weights in decision-making is a way to differentiate several maps for the same role of the same sensor. This value is dynamically adjusted during the operation of the system (e.g. a sensor with too many roles assigned gets its value decremented to avoid too much reliance by the network) and the map with a higher value is chosen.

The objective of this KB is to keep updated information about the current mappings, being the structure of the semantic mapping described in Figure 4-7. It stands as a dynamic record, updated every time a change is made to the network and is not particularly relevant for any other decision-making processes. The structure of this KB is better detailed and instantiates the structure of the mapping tuple described in Equation 4-2. Nevertheless, it is explicit that for each mapping there is only one origin sensor and may be one or more destination sensors.

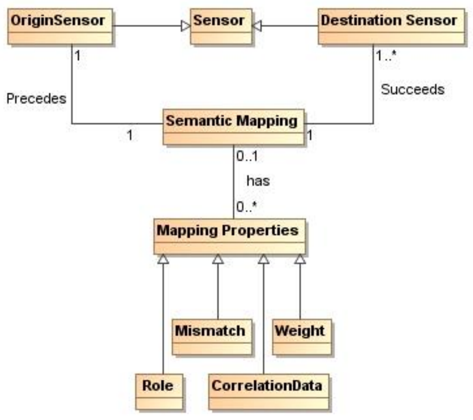


Figure 4-7: Structure of Semantic Mapping.

4.4. Analysis of Framework towards Hypothesis

With the design of the SSIF framework and architecture and the description of the three KBs throughout this chapter, a step was taken to be able to answer to the hypothesis presented, since the main objective is to create an interoperable and sustainable environment in the network. The 6-cycle diagram of Figure 4-1 details the concept to keep the network sustainable. It was necessary to introduce a phase 0 that serves as registration of enterprises and to design the network itself. Based on this concept, the architecture was divided into two major modules, the Dynamic Network Manager, and the IoT Framework, being able to identify the devices and enterprises, evaluate the interoperability between them, and propose and create the necessary mappings to create the interoperability inside the network. After the creation of the interoperability, it is prepared to monitor the network, to identify harmonization breakings and thus solve them so the network is not interrupted. The next three chapters detail these modules. Finally, having the SSIF framework been designed, at this moment it is missing the validation and the demonstration that will be done throughout Chapter 8.

5. DYNAMIC NETWORK MANAGER

The SSIF framework is divided into six phases as illustrated in Figure 4-1, in which this chapter describes the phase 0. This is the phase where is made the configuration of an enterprise, to prepare the system to operate for the collaboration within the network, being called the Dynamic Network Manager. This phase records each enterprise willing to collaborate, provides the necessary models for cooperation, e.g. devices in use, events that are triggered in its operation, services that are used by the enterprise, and the processes that are used in its execution, and establishes the enterprise network. After this phase, the system enters the normal cycle of interoperable sustainability. This chapter is divided into four sub-chapters. The first one does a brief description of the Dynamic Network Manager and their respective internal modules. The second and third subchapters describe use scenarios of the Design Manager and the Resource Manager, respectively. The last sub-chapters present do the analysis of this chapter towards the hypothesis.

5.1. Dynamic Network Manager Description

This module is responsible for the registration of enterprises (Figure 5-1 identifies main activities), at the moment that the registration of an enterprise is made, several parameters are asked to the user, which starts by introducing the basic data, such as name, fax number, contact (phone and email), etc. The system also prompts to register the models used by the enterprise, devices that are connected to the network, events that are related to the models/devices and processes used in the collaboration, and of course, the products each enterprise wants to produce in collaboration. This component is divided into two modules, the Resource Manager, and the Design Manager Modules.

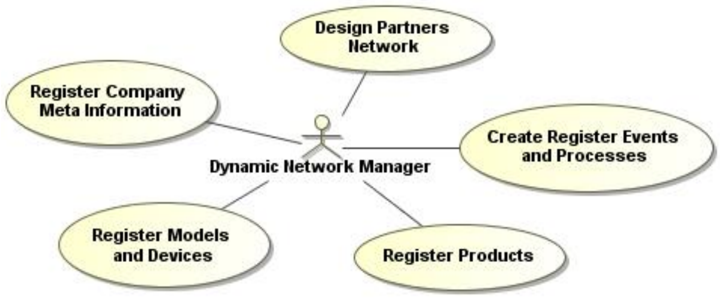

Design Partners Network
Register Company Meta Information
Create Register Events and Processes
Dynamic Network Manager
Register Models and Devices
Register Products

Figure 5-1: Use Case Diagram of DNM Module.

After the registration of the enterprise, the component provides an option for the enterprise set a collaboration network or introduce the enterprise in a network already set. At this point, it is possible to define the process of the network, where each enterprise defines its task on the network choreography of the production. This phase allows the system to identify which enterprises are on the network, what resources are being used by each one (models, devices, events and processes), allowing to evaluate the complementarity of the network with the purpose to check whether it is possible to interoperate with each other (this step is described in more detail in Chapter 6).

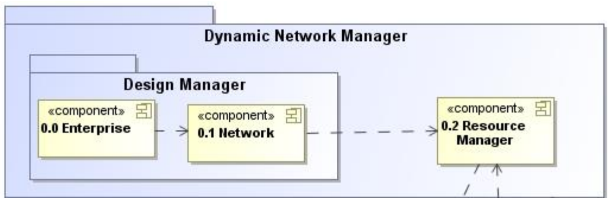


Figure 5-2: Dynamic Network Manager Architecture.

Because of the service requirements of Figure 5-1, an architecture was developed. This architecture is illustrated in Figure 5-2, is divided into three components:

- **Enterprise** – This is responsible for the registration of the meta-information of each enterprise, as the Enterprise name, VAT, Address, etc., and their products with the respective breakdown to support the description of the network. After fulfilling this step, the SSIF framework has the meta-information of the enterprise, enabling the SSIF framework to pass to the next phase. In Chapter 5.2 this topic is described;
- **Network** – To create an interoperable network, it is needed to know the partners that belong to the network. This component allows the users to describe the enterprise partners, by describing the enterprise itself (Step 0.0) and with whom each enterprise cooperates. Another feature of this module is to allow the design of the product to produce in collaboration, i.e. to identify who produce each part of the product and the timeline of the production. This measure is an enabler for the sustainable interoperable environment since it supports the initial part of the interoperability cycle by identifying where interoperability is missing and the measures to create it. In Chapter 5.2 this topic is described;
- **Resource Manager** – At this point, the enterprise resources are described, the component registers the devices in use, allowing the representation of the device network within the enterprise. In Chapter 5.3 this topic is described.

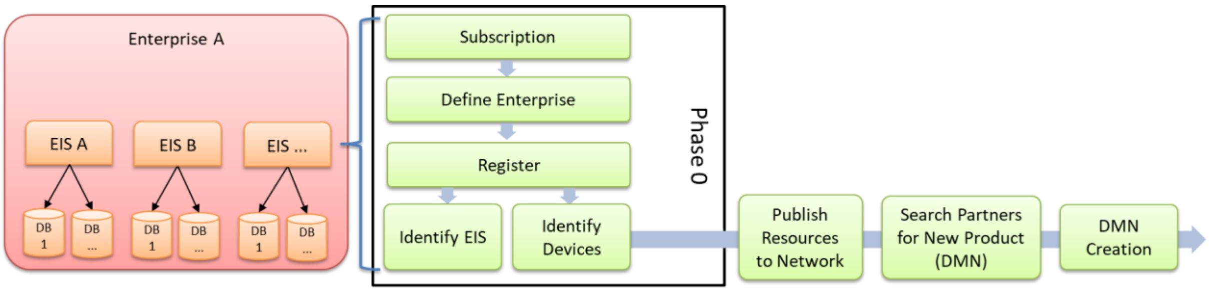


Figure 5-3: Dynamic Network Manager Methodology.

To better understand the operation of the architecture, a methodology was designed that describes the steps of Phase 0, this means the registration of an enterprise in the network (Figure 5-3). At this stage, only an enterprise is involved, because at this time (for this enterprise), the notion of a collaborative network has not yet been defined. The enterprise, when connecting to the SSIF framework, automatically starts Phase 0. At this moment, it is asked to the enterprise to provide their meta-information, the products which they want to produce in a collaborative network, and their resources, as illustrated in the figure. This information is used to improve the cooperation between the different partners, in a situation that a network wants to produce a specific product and knows how an enterprise works, enables a faster and interoperable network choreography design between the involved partners.

After the enterprise is registered in the system, the sustainable cycle can start. The collaborative network is achieved by describing the enterprise and with whom it cooperates. Hence the enterprise publishes its resources to the network, for the network to be able to know the enterprise production capacity and to identify their possible partners. At this point, the enterprise is exposed to the network, so it can look for partners with the best options available. This phase ends with the creation of the collaborative network and with the beginning of production of the product.

5.2. Design Manager

A platform was developed in the frame of the IMAGINE project ended in 2014, described in Chapter 1.4.1 To allow the platform to do an automatic search, it needs access to some meta-information of each enterprise, which to supports the choice of the best partner options depending on several criteria previously uploaded. Even though the upload of information can be done manually through a web portal, this work considers the automatic data exchange among enterprises legacy systems and the platform. Therefore, it proposes a service-based customization of the platform, capable of facilitating interoperability. It was tested with furniture sector enterprises.

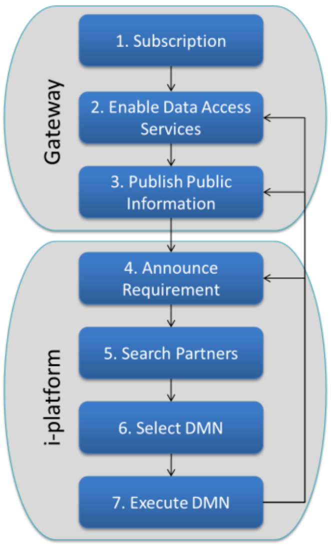


Figure 5-4: Methodology to support SMEs in the creation of an End-to-End manufacturing.

To reach the proposed idea, it was developed a methodology (Figure 5-4), which guides the DMN establishment through the platform development, including the extension made with the proposed adapter. It goes from the subscription of an enterprise until the DMN execution, triggering the collaborative manufacturing of products. The methodology identifies the steps that each enterprise needs to perform willingly to use the platform: (1) Subscription of the enterprise to the gateway portal; (2) Define services to enable access to public (business) information; (3) Publish that information in the platform, which requires internal mechanisms to transform data among legacy and platform formats; (4) Announce a production requirement; (5) Search for partners to support in the production; (6) Selects the adequate DMN; (7) Start collaborative production (platform enables monitoring of the DMN execution).

In consequence of this methodology, a gateway service for the furniture industry was developed, is illustrated in the central part of Figure 5-5 ("Furniture LL" gateway). In the left part, it is the IMAGINE platform (i-platform) and in the right the collaborating enterprises' legacy systems. At the top, there is the furniture platform portal, which is composed by a set of web-portlet's, some provided by the platform and others more customized for this industry (furniture) from the proposed gateway.

In order to create a bridge between the legacy systems with the i-platform standard data Blueprint (BP) (the Blueprints are described by Papazoglou et al. in (Papazoglou et al., 2015)) knowledge base, it was developed a service-oriented platform that for the implementation took in consideration the following points (illustrated in Figure 5-5):

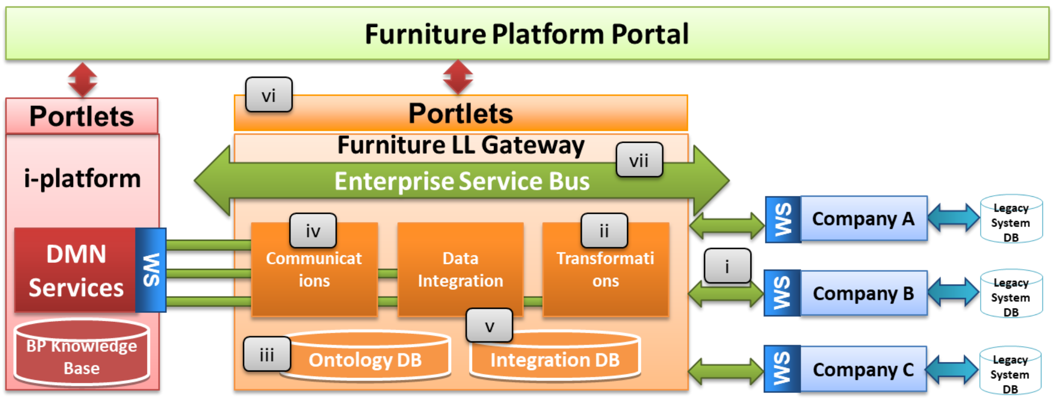


Figure 5-5: Architecture for end-to-end manufacturing interoperability.

i. The use of WS to access information from enterprises' legacy systems and databases to update the BP;
ii. The need to transform data to/from the BP structure;
iii. The use of an ontology to standardize domain-related information such as product categories, material types, manufacturing domain, etc. (since to create an ontology for the furniture sector is not an easy task, it was used the results of the funSTEP project[16], that developed an ontology for the furniture sector);
iv. Do the connection with the platform and enterprises;
v. The need to align/relate systems namely matching various nomenclatures or semantics, e.g. orders handled in both platform and legacy ERP;
vi. The need to have a customized view of the platform answering to each partner's user interfaces requirements;
vii. The Enterprise Service Bus (ESB) that does the management of the various WSs used to exchange the data between the different enterprises and the platform.

The gateway includes an ontology model (Ontology DB) to represent the reference categories

[16] http://cordis.europa.eu/project/rcn/33606_en.html

and concepts for each industrial domain. It is used to represent the semantics of the enterprises concerning their own manufacturing products and processes. Since the platform will work with different enterprises in a DMN, and each has its own nomenclature, there is a need to agree on the categories used in the platform to standardize search criteria to describe the business of each enterprise, e.g. market sector, material type, etc. Thus, enterprises can use that knowledge to support some important actions depending on the pursued goal, thus giving the platform a way to maintain and relate the categorization of different enterprises' profiles.

WS is responsible for the interconnection of the information between the different manufacturing partner's legacy systems and the platform, providing a common WS interface with the methods defined accordingly to the BP model implemented in SPARQL language, allowing reasoning of amalgamated datasets. A set of methods to retrieve the enterprise metadata from the databases of the referred ERPs of the IMAGINE Platform is made available at the gateway.

As already introduced, the furniture portal integrates various portlets. Portlets are web components, like servlets specifically designed to aggregate the context of a composite page. They extend the user interface of the portal by accessing distinct applications, systems, or data sources and generating markup fragments to present their content to portal users. In this case, it aggregates specific functions to support the DMN creation and management. These portlets are on the top of a Liferay[17] (or similar) portal, i.e. a free and open source enterprise portal, which allows the user to create custom web content in independent window container. In the platform, different enterprises could exist, each one with its own nomenclature, which requires semantic alignment with the platform. To accomplish this, it is used the Enterprise Gateway Knowledge Base (Figure 4-4), that represents points iii and v of the architecture depicted in Figure 5-5.

Table 5-1: Semantic Alignment Example.

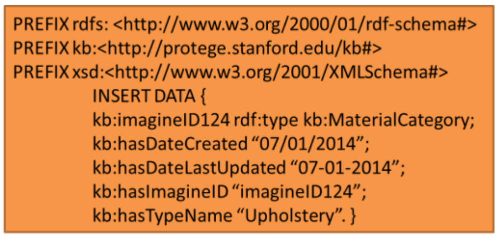
Product: Hotel Chair		
	ERP	Gateway KB
Material	Hotel Chair	Chair
Material	-	Upholstery

In Table 5-1 an example illustrating a possible semantic alignment using the Ontology DB represented. Every time that an enterprise is registered in the platform, the ontology provides the reference concepts to be used by enterprises to accomplish their needs/offers characterization process. Firstly, an enterprise sends information to be published in the platform (using the gateway WS defined). The content of the information could be, as an example, about a product that the enterprise wants to find partners to support the manufacturing process, specific WS running in the legacy systems, enabling the enterprise to upload certain metadata. However, ERP Managers have to manually categorize all the enterprise's assets uploaded. The representation of such categorizations is made aligning the enterprise's assets nomenclatures and the platform reference concepts, which are then stored temporarily in the Integration DB. Similar work on this type of semantic alignment can be found in ((Agostinho et al., 2011; Jardim-Goncalves et al., 2011)).

Elements can be manually classified with more than one concept, as in the "upholstery" case.

[17] https://www.liferay.com/pt

In Table 5-1 demonstrates the alignment related to "Hotel Chair" that was classified as a "Chair" and was associated with "Upholstering" material. Later, the gateway can transform the needs/offers of the enterprises to the BP following this alignment. Figure 5-6 presents an "insert" query in SPARQL that defines the product need to be classified as "Chair" and "Upholstery" in the material category.



```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX kb:<http://protege.stanford.edu/kb#>
PREFIX xsd:<http://www.w3.org/2001/XMLSchema#>
          INSERT DATA {
          kb:imagineID124 rdf:type kb:MaterialCategory;
          kb:hasDateCreated "07/01/2014";
          kb:hasDateLastUpdated "07-01-2014";
          kb:hasImagineID "imagineID124";
          kb:hasTypeName "Upholstery". }
```

Figure 5-6: SPARQL inserts query example.

In summary, this work developed within IMAGINE proposes a novel platform that is able to accomplish a plug-in component as a software gateway for an End-to-End manufacturing platform. This End-to-End manufacturing concept embraces the philosophy of directly connecting buyers and suppliers, eliminating middle steps as much as possible and improving their business processes' effectiveness. Thus, such proposed plug-in is responsible for establishing and managing the sustainability of seamless communications between each enterprise and the business front-end (platform) for the DMN creation and management. The gateway proved to be a crucial component because it works as an "active" gate that applies service interoperability, knowledge alignment, and model transformation techniques to facilitate an environment of SME's to adopt the DMN. In regard to the SSIF framework proposed in this dissertation, the gateway and the i-platform are a fundamental contribution to the network creation.

5.3. Resource Manager

To increase efficiency and productivity, device management is a must-have feature for a network, involving IoT-enabled enterprises. This work focuses on interoperability and sustainability of the device's network. An effective and intelligent management of the enormous heterogeneity of devices and the data exchange between them is a step forward to improve network efficiency. For instance, a device can be identified according to its properties/capabilities or by using a unique identifier like a serial number. A barometer can be identified using a serial number (xxx-xxx), and at the same time by its specific properties, such as model, measurement unit, etc., within a reasoning process.

This work proposes a solution for managing devices based on an architecture to support the registration of new devices on a network, automatically providing the services and events required related to that particular device. This method is supported by a model (to represent devices in the virtual world), a knowledge base (to reuse information and harmonization), web services (services used to access the data or to send information) and a CEP (used to manage events predefined by the system middleware). An automatic pre-configuration of a new device is done by an administrator, not only to provide a connection to the framework but also to provide services and events predefined by the system's middleware. The user is then free to decide if any other service/event needs to be added, providing a greater level of freedom in terms of configuration and setup. The system middleware responsible for installing a new device and orchestration is called DEVMAN. The DEVMAN framework provides an intelligent way to deliver additional services and events required for an effective, interoperable, and sustainable solution.

The DEVMAN module methodology is illustrated in Figure 5-7. It describes how DEVMAN reacts the moment a new device is detected on the network. The methodology is divided into three stages: the first is the detection of a new device in the network; The second step is to add additional information, services, and events to the device; The last phase represents the monitoring of a device, in a run-time usage. The first stage is called virtualization, which starts with the detection of a new device, using an installation web-service, DEVMAN identifies the type of device (e.g. temperature sensor, humidity sensor, etc.), supported by a device ontology. This ontology aims to standardize the concepts of the types of devices to avoid semantic interoperability problems. The need to identify the type of device with the brand and model is going to be supported by a KB of devices (described in Chapter 4.3.1). This step is possible because of the existence of several brands, each brand has several models, but the models have information always equal to that specific model, allowing reuse. In the second stage is the runtime preparation, using the information obtained from the KB containing the device properties, then instantiates a new device and associates its relevant services and events.

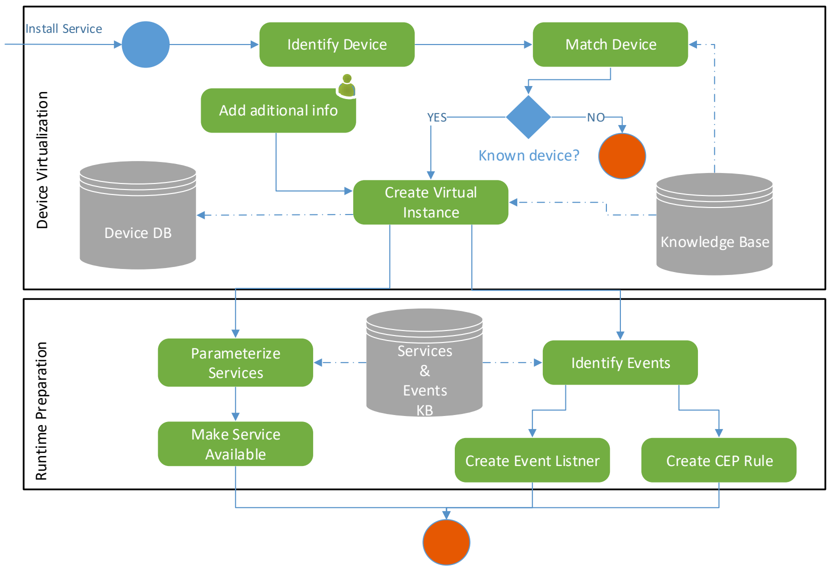


Figure 5-7: DEVMAN Methodology.

To support the understanding of the existence of the KB and its potentiality, an example with a detailed explanation about the matching device process, being is represented in Figure 5-8. It is divided into three main steps or questions “The device exists in the KB?”, "Exists Type in KB" and "Reuse sibling”. Staring with the first question, in case of a positive answer, the device already exists, the middleware creates the virtual entity of the device (in this example, the sensor Sharp SharpIRxyt, which is an IR sensor) and its corresponding attributes. In case the device does not exist in the KB, the middleware is going to do one of following options. If it cannot find a device of the same type, the process will be terminated which means there was not enough information to match the device and that the KB should be complimented. On the other hand, it will search in the KB for a similar type (in this example, there are two different similar devices), then ask the user if one of them can be used as a base device, in order to avoid a manual insertion of the relevant information. Of course, it can happen that there are no similar models of the same brand in the KB, in this situation, the methodology redirects the question to the user to choose or not a sibling. In case of positive answer, it will create the virtual device instance with additional information, specific to that device, provided by the user. On the opposite side, it will request the user information to add a new device type to the KB.

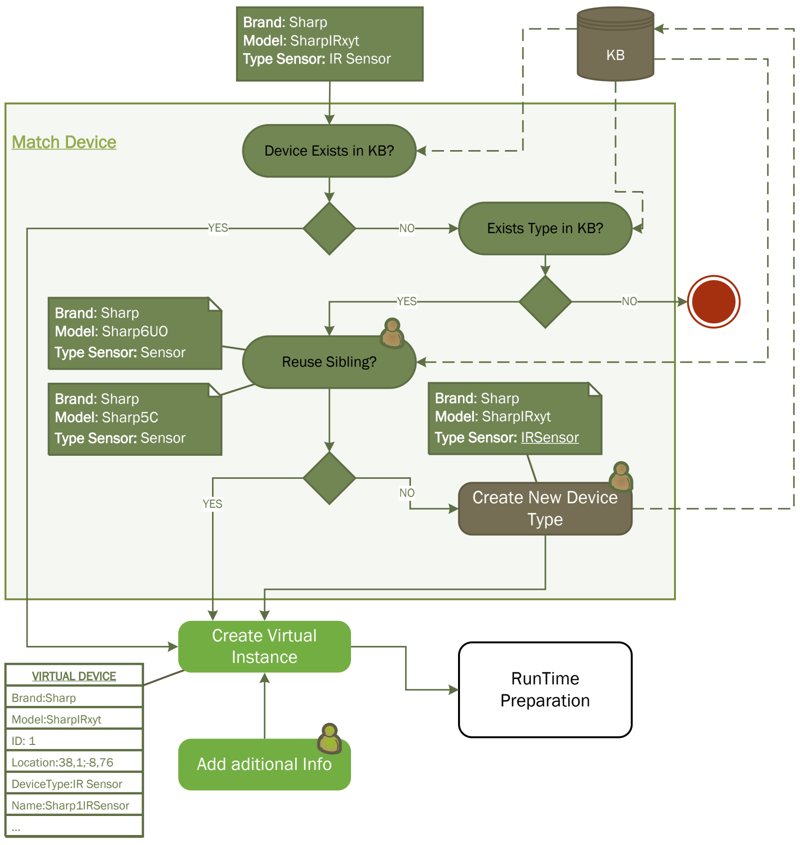


Figure 5-8: Guide Example for Adding a New Device.

With this methodology implemented in the middleware, the insertion of information of a device is facilitated, since the intelligent middleware searches the KB in a tree-like basis where it will look for a leaf node, branching out to find a proper match.

The following phase provides support to stage two (Runtime preparation) of the DEVMAN Methodology. At this moment the virtual representation of the device is created in the database, the next phase of the methodology is ready to start. This phase is responsible for attaching services and events to the device, for this task the information provided by the KB is reused since the system middleware already has a similar device stored. In this automatic process, the KB provides the relevant information for events and services that the device is going to use during its lifespan and to cooperate with the network. A list of generic services exists to make available to the devices, for each device like getting and setting data. Regarding events the procedure is different, according to the specifications and constraints provided by the KB a rule is instantiated inside a CEP, after which a listener is created that will trigger the necessary action (for example in the case of a temperature sensor, the KB provides a maximum temperature set-point that will trigger a warning). After this, the device is registered and ready to be used by the network.

To accomplish the methodology explained so far, a knowledge base was design with two objectives, as a concept taxonomy (this taxonomy is used to harmonize the concepts used by the device database, a type of sensors, actuators, unit, etc.), and to have a real-time taxonomy of the device characteristics to support the automatic insertion of a device. There many different devices, yet the same model can be used several times, therefore, this taxonomy stores this data and uses it when a similar device appears again, this will allow doing a semi-automatic insertion of the device since it will have the same characteristics. Earlier, a discussion in different models was made to select the appropriate model to be used in this module. It was selected the model described in the IoT-A project as the basis for this work, this KB is the one described in the Chapter 4.3.2.

In Figure 5-9 the architecture that implements the DEVMAN methodology is illustrated. It aims to manage the devices and connect them to the cloud platform. To reach this objective, the architecture is supported by a model to manage all the characteristics and features of the devices. Several components are used by the architecture to support the main object such as services, events, ontologies KB, etc. Using the project's description from the previous sections, services and events are mainly based on the IoT-A project while the ontologies and knowledge base are derived from the OSMOSE project (these projects are described in Chapter 2.2.1). The architecture is therefore divided into three main parts. The **Device Layer** represents the physical component of the architecture which is the devices themselves. The **core module** (DEVMAN) is where the orchestration occurs, and interoperability is achieved for device management. The final layer is represented by a **Graphical user interface** (GUI), where it is possible to view and manage the pertinent information.

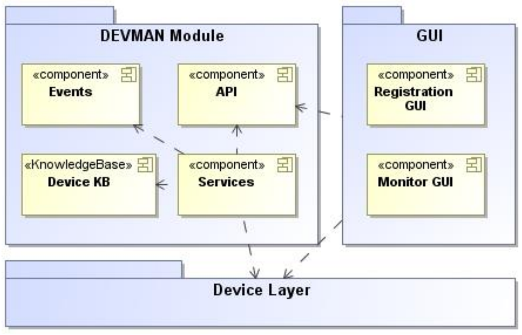


Figure 5-9: DEVMAN Architecture.

Implemented according to the methodology described in Figure 5-7, the DEVMAN module is responsible for identifying patterns during the device registration, and using these patterns to maximize the automation of registering other devices. This is made by proposing possible registration of each device, defining their characteristics, proposing the services and events that can be used by them. This feature is supported by a knowledge base, which is continuously being enriched overtime with the experience of other similar devices, and a database that provides storage for the initial setup, in terms of, which services and events need to be associated to a specific device. This module is implemented using java to process the information from the other components:

- **Knowledge Base** – The KB aims to do the registration of devices in the framework, by supporting the user, accompanied by each step of the device registration. This topic is addressed in the Chapter 4.3.2;
- **Services Component** – This component is a core entity in the architecture that provides and exposes model information not only to provide access but also to give events a way to report contextual awareness or notification to business stakeholders outside of the network. The Services component is divided into two parts. The InstallService (services used to register a device in the model), which is fundamental to phase 0 of the framework of Figure 4-1, handling device pre-configuration, after this registration the device is "visible" to the rest of the network. Depending on the features of the device, the services and events that can be consumed by the device are also added). The RunTimeService (services used after the registration of the device is completed), not part of phase 0, is used by each device during their time in the network. Since each device is different from the other, several services exist and can be used. For actuating services, the state of the entity attribute being controlled is also important. This post-condition state is modeled through a service effect. The same happens with a service pre-conditions that need to be met for the service to be used;
- **Events Component** – In a network, each device has a purpose, it is used to monitor or to send notifications to another device(s). The events are used to facilitate this purpose, and this module is the engine where the rules of the events are created (phase 0 of the framework of Figure 4-1) and runs over the lifetime of the device, triggering the necessary actions through the services module. Using the ontology defined by the OSMOSE project, an event can be categorized as, for example: Message Event; Timer Event; or Error Event; etc., other classifications are possible. The discussion made about CEP engines supported the selection of the appropriate CEP to use in this module. EsperTech CEP was selected;
- **API** – This component is the set web-services that expose the functionalities of the other different models including the GUI;
- **GUI** – Provides a visual representation of the entities as well as interfaces relevant to properly manage the devices;
- **Device Layer** – It represents the physical part in the architecture (e.g. Arduino, Raspberry Pi, etc.) that sends the necessary information depending on the nature of the device to the cloud via the services component (web-services).

In summary, this work focuses on describing a framework, called DEVMAN platform, for enabling a rapid interoperable solution for adding new devices to a network. Every time that a new registration is made, the framework follows the methodology described in this work, creating the necessary service association (from generic services) to access not only the mapping of the general properties but also the specific attributes that mainly characterize the device and corresponding associated values. It also aids in creating and associated the appropriate set of rules (from generic events), to a device, inside the CEP to implement the necessary actions or alerts pertinent to that event, this is used Phase 6 of the framework of Figure 4-1. The use of Services as a doorway contributes as a source of the CEP engine for providing the necessary event processing actions, giving a more scalable solution, to help provide a contextual aware environment. The DEVMAN methodology and DEVMAN architecture, described in this work are a step forward in contributing to the vision of IoT, in helping businesses and industries processes. In resume, this work focuses on describing a framework, called DEVMAN, for enabling a rapid intelligent solution for adding new devices to a network. Although there are a number of projects that try to deliver a solution for the problem at hand, this work attempts to combine some parts these projects to provide a faster and more reliable approach. DEVMAN shows that the incorporation of a CEP on a reference architecture supported by a KB model can scale even more the extensibility of the network for a rapid and interoperable implementation process. The large world of heterogeneous devices has a set of common properties such as brand or model. However, the main characterization of a device, either if it is a temperature sensor or a button, needs to be mapped differently from the main device properties. The DEVMAN methodology describes the key process of matching a device according to a KB that is accomplished by using a tree-like data analysis to accommodate the device being registered. Although this may require intervention by a human user, the process is still much more expeditious than doing a complete manual registration.

5.4. Analysis of the Dynamic Network Manager towards Hypothesis

DNM’s goal is to execute phase 0 of the SSIF framework, being responsible for registering the enterprises and their resources, and designing the network of collaborative enterprises, to be able to identify how they will cooperate with each other and check where interoperability issues can arise. Although this module seems to be of less importance, since it is used once for each enterprise, it becomes the module that has quite strong objectives to help answer the research questions, since it is the module that allows the identification of the enterprise’s models, giving rise to the desired interoperability. At the same time, has an impact on the identification of the network, as it is at this time that one gets to know the collaborative network and the resources of each enterprise, helping in this way to answer the hypothesis.

6. IoT Framework – Design Mode

To create an interoperability network, it is needed for enterprises to “understand” each other. To achieve this result, an identification of each model and devices are required, to create the bridge between each other, resulting in an agreement in that communication. This is the objective of the module described in this chapter it is the Design Mode of the IoT Framework. Getting four phases of the SSIF framework of Figure 4-1, the Phase 1 (Knowledge Extractor), Phase 2 (Interoperability Driver Specification), Phase 3 (Modeling & Simulation) and Phase 4 (Runtime Configuration), during this chapter these phases are described.

6.1. IoT Framework – Design Mode Description

This module is responsible for the design of the interoperability in the network, in other words, responsible for identifying solutions to put each enterprise communicating with the other and preparing them for the Runtime mode. In Figure 6-1, a study to identify the main activities of the Design Mode is illustrated. This mode aims to identify how to put two enterprises interoperable, by identifying their models and presenting solutions (identifying the mappings to enable the change of data between the different models). At the same time, identify the device's network and propose semantic mappings between the active devices, this solution is going to present different solutions for any time that a device stops answer, avoiding harmonization breakings in the device network.

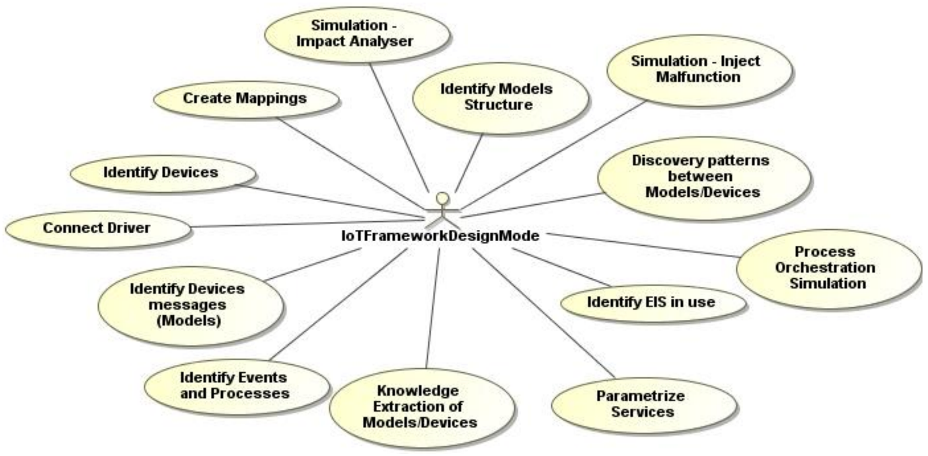

Simulation - Impact Analyser
Simulation - Inject Malfunction
Create Mappings
Identify Models Structure
Identify Devices
Discovery patterns between Models/Devices
Connect Driver
IoTFrameworkDesignMode
Process Orchestration Simulation
Identify Devices messages (Models)
Identify EIS in use
Identify Events and Processes
Knowledge Extraction of Models/Devices
Parametrize Services

Figure 6-1: IoT Framework (Design Mode) Use Case Diagram.

Then, the framework asks the user to design a process for a new product, where it is identified the collaboration between the enterprises and identified the resources that are used between them. At this moment the enterprises are ready to collaborate and to avoid possible harmonization breakings, it is made a simulation with the aim of identifying possible problems and solves them before it affects the network. After the user agrees with all the changes proposed by the framework, the driver is connected, being the end of the Design Mode.

In Figure 6-2 the IoT Framework Design Mode Architecture is represented, the Design Mode is divided into four phases: 1) Knowledge Extractor; 2) Interoperability Driver Specification; 3) Modeling & Simulation; and 4) Runtime Configuration. In the next sub-chapters are explained each of these phases.

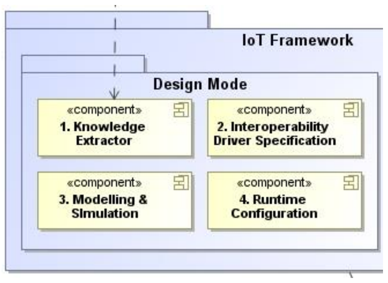


Figure 6-2: IoT Framework (Design Mode) Architecture.

6.2. Phase 1: Knowledge Extractor

The Knowledge Extractor Phase is responsible for identifying and implementing the necessary requirements to create interoperability between different components in a network of the enterprise. To achieve this goal this phase pass through two different paths (as illustrated in Figure 6-3), one path when an EIS is identified and another path when a Device is identified, in this situation to identify the devices is used the work described in Chapter 5.3, in which it registers and through this register it is possible to identify the type of device and its role in the network. These two ways end up being intercepted, but as in the case of the EIS, there are two more steps, for this reason, it was necessary to make this derivation.

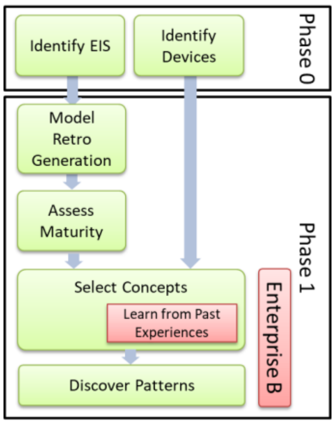


Figure 6-3: Phase 1 Knowledge Extractor Methodology

Following the path of the EIS appears two steps; the first one is the Model Retro Generator which has to check the existing EIS of an enterprise, to verify the models in use and to identify it, as discussed along Chapter 2.1, enterprise modeling can follow different paradigms, maintaining different types of models and information views. To do identification of the model, first it is needed to identify the model itself (how the structure is organized), this is made with a reverse engineering, by creating the model that represents the database structure. Then, it is made a knowledge extraction of each model, which collects the knowledge of the model, by derive to one or more relational models (see the work of Lezoche et al. (Lezoche et al., 2012)), which can then be abstracted to the conceptual level. The purpose of these initial steps is to identify the conceptual data models that need to be interoperable, in order to be able to compare with another model, at the same time it makes an assessment of the interoperability of the models to know what in this model is interoperable.

At this moment, the two paths come together again, in the Select Concepts step, and at the same time, they begin the integration with another enterprise. This step does the relationship between the models, by detecting the common or similar in use. The selection of concepts is used since, each EIS can be from a different country (that can be using different language), or some concepts can have different meanings in different domains (depending on the interpretation of each person/entity), etc., this can create semantic problems that will make harmonization breakings to occur in the environment. For this, it might be needed an ontology to standardize the concepts, to maintain the concepts harmonized in the different EIS. With the devices happens the same situation, it is needed to identify the type of device, their role, the type and structure of data to relate to being able to propose a semantic mapping.

As the framework does not contain all the concepts stored, sometimes there are situations that cannot be solved. Having to ask the user for support, for this reason, a sub-step has been inserted in the methodology, being responsible for learning with the past experiences to use in the future. Finally, it is the Discover Patterns step, with the concepts selected this step is able to search for patterns and correlate them with the aim of creating the mapping which will be the one to be used in the change of data in the Runtime mode.

6.3. Phase 2: Interoperability Driver Specification

The last phase was able to identify the patterns in each model or device, in Phase 2 the framework is responsible to specify the driver, in this situation it is going to define the semantic mappings between the different models and devices to be used in Runtime mode. To the framework define and propose the semantic mappings to the user, several steps are needed, so a methodology as design to explain it, this methodology is illustrated in Figure 6-4.

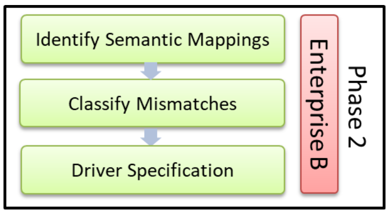


Figure 6-4: Phase 2 Interoperability Driver Specification Methodology.

Phase 2 is divided into three steps (following the figure), the first step is the Identify Semantic Mappings. The aim of this step, it is to define the mappings that correlate the different models and devices. In a situation that two models are being correlated to define the mappings, the framework suggests the mappings between the different fields of the models that are able to be mapped, then these mappings are stored to later use. In the case of a device, the mappings are related between different devices, this means that a device has mappings to one or more device that are similar or can do the same role that was assigned to the device. In a situation that a sensor is malfunctioning, the framework goes to the semantic mappings and identifies a solution (or more) to replace it, maintaining the network working smoothly without occurring problems. This allows the identification of semantic mappings between the different sensors, making these mappings to support the maintenance of the interoperability in the network.

After the identification of the mapping, it is needed to classify the Mismatch of each mapping, as explained in Chapter 4.3.3. At this moment, it is defined the mismatch for each mapping with the aim of identifying the inconsistencies of each mapping, following the specifications of Table 4-1. In the final of this step, the mappings are ready and it is made the driver specification for each enterprise, and pass to the next phase.

6.3.1. Morphisms to Support Sustainable Interoperability of Enterprise Systems

This work proposes the MIRAI (Monitoring morphIsms to suppoRt sustAinable Interoperability of enterprise systems) framework to monitor the systems' interoperability through the morphisms previously defined and stored on an enterprise CM. MIRAI detects changes in the interoperable environment, proposing the user morphism re-adaptations with the advent of harmonization breaking. The MIRAI is stored in each enterprise that comprises an enterprise network and every time that a change is detected in one MIRAI, it will trigger a warning to the others MIRAI's in the network to see if that change has an impact, avoiding with this problem in the future.

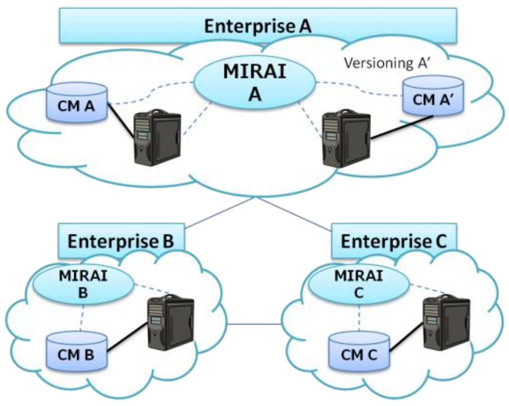


Figure 6-5: MIRAI Network.

As illustrated in Figure 6-5, MIRAI has the objective of monitoring the existing mappings and model versioning's stored in each enterprise's CM and timely detect the changes in the morphisms, proposing to the user a possible solution and preventing a significant transient period where interoperability in the network is not assured. The detection is carried as soon as CM changes, triggering an agent to search model differences. Indeed, when a versioning on one of the models is detected, MIRAI triggers a warning and automatically proposes a new mapping morphism to the user. This new suggestion is based on the 5-tupple mapping expression proposed by the authors in (Agostinho et al., 2011) and it is used to describe the relationship among the different models that are used in the enterprise, regardless of the language used. Within this framework, a mapping is created to respond to the evolution, where according to the tuple MapT. After that, since the user might accept the proposal or not, the authors decided to endow MIRAI a learning process based on weights to help in the choice of MatchClass for the new mapping, and increasing intelligence over time, and it will be explained later.

There are various types of mappings, which MIRAI could propose (e.g. Structuring, Semantics, Conceptual or Instantiable Data) to, relate such model elements and depending on the user choice, the proposal is accepted and is stored in the CM. The next time a similar situation occurs, MIRAI will provide a similar solution only if it (from the various possible mappings types) remains to be the most weighted one according to user's choice pattern. For reaching such objectives, MIRAI is directly associated to each CM. Moreover, within the collaborative network of enterprises there will be a kind of sub-network, i.e. the MIRAI network (as in Figure 6-5) that enables to keep all CMs synchronized (interoperable) and maximizes the learning process as the whole distributed framework contributes to knowledge concerning user's selections, which is shared among the mediators.

In resume, the role of MIRAI is to monitor all the mappings that exist among the several models used by business partners in the same collaborative network, controlling the changes, warning and proposing new mappings, preventing interoperability problems that could cause a destabilization of the network harmony. Enterprises' privacy is assured since each one has its own MIRAI associated to an internal CM that tracks the morphisms it maintains with their direct partners. In this chapter was described how MIRAI describe the mappings and the advantages of using it to map different models.

6.3.2. Semantic Maps for IoT Network

After the proper configuration processes, the IoT network should maintain an operable working state that corresponds to the high number of devices associated with various IoT applications. The failure of a single sensor should not compromise a whole network and, due to the high physical redundancy of having many devices, should be surpassed. To accomplish this, the error must be acknowledged, understood and should result in an unharmful and efficient network reorganization. This work aims to propose a contribution to the process using a concept definition usually regarded in areas of linguistics and learning, the semantic maps, which intends to provide meaning and knowledge to words in the same semantic field. In technological terms, the construction of semantic maps can be a contribution to achieve an intelligent and efficient network monitoring system, especially in challenging contexts of big data, where there are many heterogeneous sources that produce too much information to be manually reviewed (Knoblock & Szekely, 2013). The idea behind the use of this concept is to gather and organize meta-information, regarding the network and its devices, to allow redundancy by recovering from device failures. This recovery is possible by using the semantic maps as a source of information to this process.

To help explain the purpose of this work, a brief example is described in Figure 6-6, let's say sensor B is damaged by unknown reasons and provides unusable information (e.g., temperature readings out of predictable thresholds or an anomaly detected by comparison with nearby devices). Sensor B is now a liability to the event processing and may cause several rules to become obsolete for not having the necessary information to be triggered. To tackle this situation, human intervention is usually needed. In large industrial networks or in the IoT paradigm, there are numerous sensors, deployed for different uses, which may be suited to comply with operations related to Sensor B. To measure that possibility, sensors can be analyzed regarding aspects like localization, type of measurement, role in the network and others. In this case, Sensor A and C, serve the same purpose and are suitable to replace sensor B in rules associated with it, as well as other similar sensors that measure temperature.

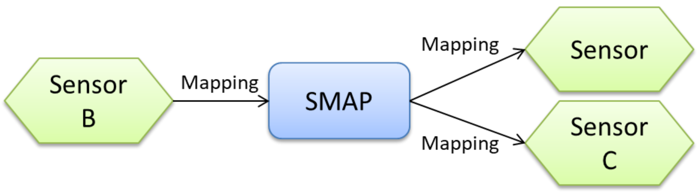


Figure 6-6: Sensor Malfunction Example.

This (Figure 6-6) is a simple situation, manageable by human intervention if it occurs sporadically. The idea behind this work is to present a mechanism that provides autonomous dynamic adaptation. This is where semantic mapping comes to play because it represents and maps possible redundancies while providing room for developing the autonomous creation of possible maps regarding the aspects of the role of each sensor in the network. This autonomous creation is possible, but it is advisable to create the initial configuration of the existing devices and relevant possible core semantic maps by the developer, to provide consistency to the network.

Thus, with the intention of creating this independence, it was developed the architecture represented in Figure 6-7. For this design, all the mentioned functionalities and processes were considered in addition to how contemporary systems of this kind are defined. It is important to mention that the fact that the SMAP module (SMAP derives from Semantic MAP), is defined separately from the rest of the system and somehow as autonomous as possible, making it not mandatory for a system to implement the SMAP module at an early stage, being possible to implement it on existing systems. The SMAP is divided into several modules as:

- **CEP Engine** – It is responsible for the runtime process and the CEP DB contains the required information for its functioning and the event rules that are compared with the situations of interest. The events processed in this module come from the devices. It is also important to mention that some event processing, regarding the verification of correct sensor operation, can be done by the semantic mapping module, allocating some working load from the CEP. This CEP is the same as that used in DEVMAN, having the same functionalities, being used to monitor the devices in use;
- **Devices** – Devices are the low-level part of the architecture that represents the technology that provides and generates events to be considered by the CEP and SMAP modules. These events are usually raw information about the monitored environment and make no judgment or evaluation about it. Devices are connected to the semantic mapping module

because of the need to keep an up-to-date device database that may also be updated, autonomously or manually, according to the changes that may occur;

- **Device Knowledge Base** – The main objective of this KB is to keep an updated registry of the sensor's information (e.g. the current roles, room, and position). The idea behind having a specific KB for this information is to specify and contextualize the sensors dynamically, during the functioning of the system and use of semantic maps. This KB is described in Chapter 4.3.2;
- **Mapping Knowledge Base** – Similarly, to the previous KB, the objective of this KB is to keep updated information about the current mappings. It stands as a dynamic record, updated every time a change is made to the network and is not particularly relevant for any other decision-making processes;
- **Semantic Mapping Module (SMAP)** – The SMAP module stands as the module that represents the concept and methodology of this work. It interacts with the devices, mapping knowledge base, device database and the CEP module, specifically the CEP engine.

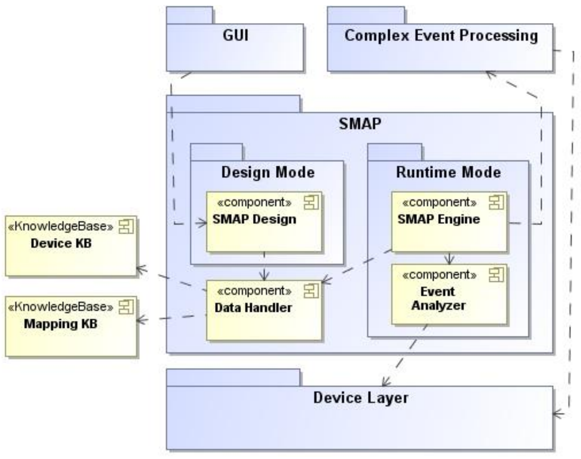


Figure 6-7: SMAP Architecture.

Since the SMAP is the central module of this architecture, it has a few main functions that are:

- **Specific event listening from CEP and Devices** – These events are analyzed according to situations of interest regarding the monitored environment, the potential of the developed semantic maps and the pre-determined mapping situations;
- **Verification of sensors** – The SMAP module, like the CEP, will process events that depict situations of interest. The main interest is to analyze the state of the sensor and to check if it is functioning properly. To accomplish this, it is started a loop of event listening and processing, until some situation triggers a semantic mapping procedure. Some pre-defined failure situations can be, e.g. measurements that exceed typical thresholds, constant measurements, noisy readings or non-concordant measurements between two or more sensors of the same type, in the same area (Munir & Stankovic, 2014), (Munir & Stankovic, 2014);
- **Search and update queries to device database about origin sensor** – In order to keep an updated record of the sensors and to search the correct mappings, the semantic

mapping module searches the available information about the sensor that expressed a failure, including the roles that is operating at the moment, and updates them to “none” and changes the state of operation of the device to Boolean zero);

- **Search and update queries to mapping knowledge base** – After acquiring the information about the origin sensor, the SMAP module elaborates the request for a semantic mapping solution. To accomplish this, it searches the role, or roles, of the origin sensor in the mapping knowledge base. If there are no semantic maps for that type of role, manual intervention is solicited or mechanisms of autonomous mapping are triggered. After finding the semantic maps for the needed role, the maps that have the mentioned sensor as the origin sensor are selected, by adding constraints to the previously mentioned query. The remaining maps, if more than one, have a weight component associated with them, as mentioned before, and the map that has a higher weight value is selected for the semantic mapping process;
- **Search and update queries to device database about destination sensor(s)** – Similarly to happen when using the device database for the origin sensor, the semantic mapping module queries the device database for information about the destination sensor:
    - It retrieves the current state to verify if it is indeed available (normally it is operating because the device database should have updated information. The roles may also be considered to avoid too much reliability on a solution, but again this is controlled by the weight value in sensor database and it is typically updated regarding that issue. If everything is according to the specifications for good functioning, the sensor is updated within the database with a new role and state, if that is the case);
    - Sensor output correlation (In this phase, within the semantic mapping module, the correlation between outputs from the origin and destination sensor is considered recurring to the output data and mismatch from the mapping information, retrieved from the mapping knowledge base. Any particular change or specification that the CEP has to deal with, in the event processing or event rules that use the destination sensor considered, are taken into account);
    - Event update in CEP (In this final phase, the information about the destination sensor, or sensors, is provided to the CEP to make the necessary changes to the events, previously using the origin sensor and replacing it. The information considered in the last point, sensor output correlation, is also provided to the CEP in order to integrate them in the mentioned events).

Considering the mentioned points, it is logical to define sub-modules according to the specifications and functions of the SMAP, as illustrated in Figure 6-7. The first two points, regarding the analysis of situations of interest, are the responsibility of the “Event Analyzer” module. Every change or query made to the device KB and mapping KB use the module named “Data Handler”. The core runtime process and, generally, all other module processes, are conducted by the “SMAP Engine”. In addition, the module “SMAP Design” is oriented to the design time of semantic mapping tuples by means of an interface or autonomous processes by a human user.

In resume, this work suggests a process for using semantic maps, recurring to network metadata and function-oriented recovery methodology, to accomplish autonomous error recovery and network reorganization, is presented. The objective of this process is to improve the IoT network sustainability, reliability, and trustworthiness regarding IoT devices, without compromising performance or significant structural changes to already implemented networks. These aspects were considered during the design of the SMAP module and its interactions with the rest of the system.

To contextualize and demonstrate the developed work, a semantic mapping tuple, and the use of the resulting map, was described to represent the functionality and to show the viability of the use of this solution. This implementation, regarding the validation of the objectives of this work, shows the capability of the network, when using the semantic mapping methodology, to detect common errors, trigger an error recovery process, analyzing the redundancies provided by the existing semantic maps, and to reorganize the network, to return it to a similar working state, as it was before the error occurred. Another important aspect is the potential to develop procedures for autonomous recognition of error patterns and the autonomous creation of new semantic maps, within the SMAP Design module.

6.4. Phase 3: Modeling & Simulation

This phase is responsible for the modeling and the simulation of the network processes, with the objective of modeling the processes of the enterprises and at the same time, it is possible to describe them. This gives the capability to the framework to know how the processes are, and allow to simulate and detect possible harmonization breakings before they occur. In Figure 6-8 the methodology of this phase is illustrated, it is possible to see that three main steps exit. It starts with the design of the processes of the enterprise, is provided a tool for the users design their processes. The advantage of using the process design is that the framework knows how the process works, facilitating the monitoring and the identification of errors. The design is to represent the process between the models and devices, identifying what is used and what is being transmitted to the different stakeholders.

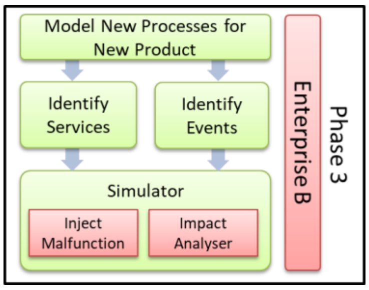


Figure 6-8: Phase 3 Modeling & Simulation Methodology.

In the moment the user confirms that the process is ready to use, it is asked to provide the services and events that this process needs to run. This is specified in the same design tool giving freedom to the users to describe all tasks, services, and events over the process lifecycle. The simulation has the aim to anticipate possible problems (i.e. Prediction of Future Behaviors) and addressing them before they occur. As actions in the present can create undesired problems in the future, this will use past knowledge to support decisions that may impact the efficiency of the network, to avoid it in the future. With these measures, the framework can do two types of simulations, Inject Malfunction and Impact Analyzer.

- **Inject Malfunction** – In this situation, the devices are simulated, to identify the impact that a device does in the moment that a malfunction is detected, which can be something as battery fault or failure, a message is lost, etc. This simulation aims to identify possible problems in the network, by creating conceptual maps of the network with the aim of creating subnets for use when a device fails, thus being able to use another device to replace and fill that gap;
- **Impact Analyzer** – To be able to predict possible problems with changes in the models, it is necessary to do a simulation at the mappings levels. By simulating different types of mappings and compare them with the existing ones, it is possible to identify situations in time to prevent problems that occur on the network. To support prevent problems in the network, in order to know which enterprises intervening on the network, and how this intervention is made, for example, specify the data sent between models, identify which data the devices are transmitting, etc. When having a notion of how the network operates it is possible to simulate the network and identify possible situations that can cause problems. Thus, it is possible to solve the problems in time, or (if it is not possible to solve it) to identify these situations to be the firsts possible causes of problems.

6.4.1. Process Modeling Approach for the Liquid-Sensing Enterprise

As it was detailed in Chapter 1.4.3, the osmosis processes are a special type of process used to moderate the information exchange among the real, digital, and virtual worlds. When instantiated, these processes will enable to seamlessly integrate the LSE, connecting events across the three worlds, and triggering services to provide the enterprise full knowledge about its inner systems and interactions.

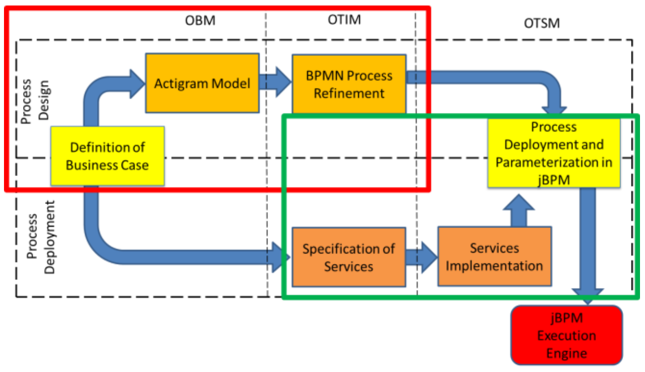
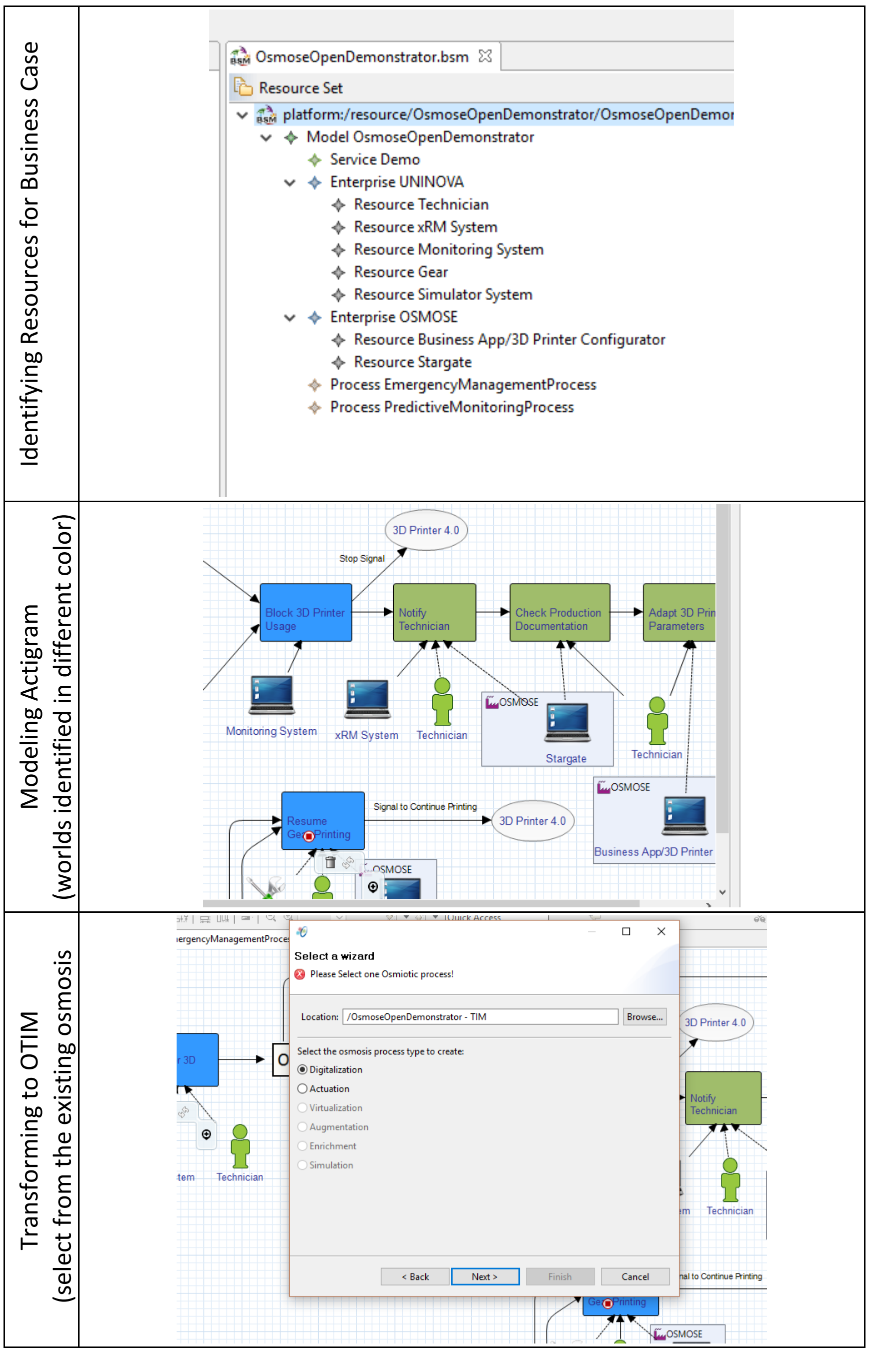


Figure 6-9: Overall Structure of Process Development.

The process design framework allows enterprises to take the most out of LSE and the

OSMOSE project, being able to carefully plan the new business strategies or specify the new services clearly differentiating activities and events in different worlds. Next section describes how the process design methodology is combined with the services specification and deployment, used at execution time (Figure 6-9). It is divided into two phases, the **Process Design** and the **Process Deployment** (the Process Deployment is out of the scope of this chapter, is described in Chapter 6.5.1). During this chapter is described the Process Design in the red square in Figure 6-9.


Identifying Resources for Business Case
Modeling Actigram (worlds identified in different color)
Transforming to OTIM (select from the existing osmosis)
3D Printer 4.0
Stop Signal
Block 3D Printer Usage
Notify Technician
Check Production Documentation
Adapt 3D Printer Parameters
Monitoring System
xRM System
Technician
OSMOSE
Stargate
Technician
OSMOSE
Business App/3D Printer
Resume Gear Printing
Signal to Continue Printing
3D Printer 4.0
OSMOSE
Browse...
< Back
Next >
Finish
Cancel

OsmoseOpenDemonstrator.bsm

Resource Set
- platform:/resource/OsmoseOpenDemonstrator/OsmoseOpenDemor
  - Model OsmoseOpenDemonstrator
    - Service Demo
    - Enterprise UNINOVA
      - Resource Technician
      - Resource xRM System
      - Resource Monitoring System
      - Resource Gear
      - Resource Simulator System
    - Enterprise OSMOSE
      - Resource Business App/3D Printer Configurator
      - Resource Stargate
    - Process EmergencyManagementProcess
    - Process PredictiveMonitoringProcess

Select a wizard

Please Select one Osmiotic process!

Location: /OsmoseOpenDemonstrator - TIM

Select the osmosis process type to create:
- [✓ Digitalization]
- Actuation
- Virtualization
- Augmentation
- Enrichment
- Simulation

Figure 6-10: OBM Modeling Activities.

The Process Design starts with the Definition of Business Case, which is a high-level description of the business case/service to be implemented. It can be made in the form of textual description of the user story, or a more formal definition following models. After that and illustrated in

Figure 6-10, the design of the Actigram Model is conducted (EA* (Extended Actigram Star) language is used (H. Bazoun, G. Zacharewicz, 2013)). It represents the initial part of the OBM level, starting with the specification of the enterprise, collecting meta-information about the organization and the resources (as illustrated in the upper figure of Figure 6-10).

Then, it is specified the business perspective of the process model (as illustrated in the middle figure of Figure 6-10), by identifying the innovation requirements and expected behavior. Using this model, the user visualizes in a simple form, which activities will go into processes between the different worlds. The world's identification procedure at the OBM initiates the model-driven paradigm, enabling the system to identify osmosis processes and ask the user the type of osmosis event that can occur (see the last figure of Figure 6-10). This gives the possibility to change from the OBM level into the OTIM level, through an automated model transformation that transforms the Actigram into a 3-parts BPMN model (BPMN 2.0 is used to instantiate OTIM in this work) representing the OSMOSE membrane and the respective worlds processes. The transformation used in this process is described in the next pages.

The next step is the BPMN Process Refinement; at this phase is possible to specify additional details for service integration and extended business logic. This represents the OTSM level of the methodology preparing the BPMN for execution, and it is out of the scope of this chapter, is described in more detail in Chapter 6.5.1.

To facilitate the identification of the osmosis processes between different OBM activities, the user should select to which world the activity belongs. This option changes the color of the activity in the diagram, hence facilitating the visualization of the worlds by the user, and providing the system the necessary information for it to compute the existing osmosis processes in a single business case (see the middle part of Figure 6-10). When more than one exists, the user should address them separately in different OTIM models.

To support this process identification, an algorithm has been specified and implemented. It detects the world transitions and asks the user which osmosis process he wants to work on. This new feature improves the transformation between the EA* and BPMN instances of OBM and OTIM, following the concept presented in Figure 6-10. For example, looking to Figure 8-16, the algorithm is going to ask the user to select between two transitions (which correspond the transitions between Blue-Green and Green-Blue), i.e., Digitalization and an Actuation processes.

Figure 6-11 describes the algorithm dividing it into two phases. The first phase is about the detection of the transitions between the worlds, and the second part about the detection of the Activities flow within the process. The algorithm reads the EA* diagram into a graph structure to search for world transitions. It does this by applying the following rules to the EA* model:

- A start and stop events needs to exist;
- At least one world transitions need to exist.

In the case of the model does not respect these rules, the algorithm is invalid and the framework does not allow the execution of the transformation.

After the user selects which osmosis process he wishes to further specify at OTIM, the second phase starts to iterate the graph and will identify all the Activity blocks that belong to the selected osmosis process. It starts detecting the Activities Blocks back and forth from the world transition point (e.g. RW->DW). It follows the graph until it detects the start/end of the diagram or a different world (RW backflow; DW forward flow). In the end, it merges the two flows into the osmosis EA* model to be transformed. As in the first phase, there are also some rules:

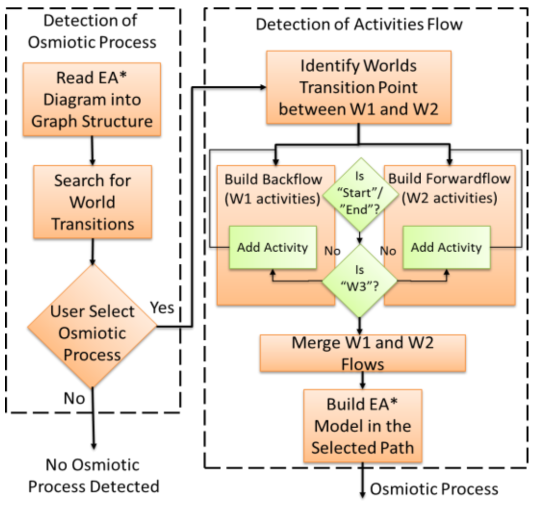

Detection of Osmiotic Process
Read EA* Diagram into Graph Structure
Search for World Transitions
User Select Osmiotic Process
Yes
No
No Osmiotic Process Detected
Detection of Activities Flow
Identify Worlds Transition Point between W1 and W2
Build Backflow (W1 activities)
Is "Start"/ "End"?
Build Forwardflow (W2 activities)
Add Activity
No
No
Add Activity
Is "W3"?
Merge W1 and W2 Flows
Build EA* Model in the Selected Path
Osmiotic Process

Figure 6-11: Algorithm for Osmosis Process Detection.

- Detect if an activity is already included in the flow and stop the iteration (it avoids to repeat it the case the graph iterates through the same activity more than once);
- "And" or an "Or" connections points respect the same rules as the Activities.

To support the modeling and model transformation process and it is being used behind the rules explained in Figure 6-11, this work continues the development of the MSEE's Toolbox for service modeling (H. Bazoun, G. Zacharewicz, 2013; Wiesner et al., 2014) which is using ATL[18] engine to automatically execute the predefined transformation rules between OBM and OTIM models. Since, the MSEE Toolbox did not have the OSMOSE processes concept implemented, (Marques-Lucena et al., 2016) started to make the update which is here continued with the new transformation rules identified in Table 6-1.

The changes made to the transformation are divided into two parts, the first one is related with the changes made in the resource, the first version of the Toolbox was made to do the transformation of the Human and IT resource, and at this moment the Material resource is also being contemplated. The second part is the big change since at this moment each activity is being transformed into the task and being allocated in the respective world. For example, in the case of being a Digitalization process, three different pools are created, one for the Real World, other for the Digital World, and the last for the Osmosis Membrane. Then in each pool is allocated the respective lanes (each lane represents the resource which is being used in that world), the tasks. In the case of the Start or End event is due to the rules of the BPMN, as each pool needs to have a start and an end event (missing in the previous

[18] https://eclipse.org/atl/

version).

Table 6-1: Summary of changes made in EA* to BPMN 2.0 Transformation.

Source Concept		**Target Concept**		
EA* (OBM)		BPMN2.0 (OTIM)		
Resource	**Material**	Data Object		
	Human	Lane		
	IT	Lane		
Osmosis Process (Digitalization, Virtualization, etc…)		Pool	*Source (Real, Digital, Virtual)* world	End Event
				Source Lanes
				Source Tasks
			Osmosis membrane	Predefined Pool, Events, and Tasks
			Target (Real, Digital, Virtual) world	Start Event
				Target Lanes
				Target Tasks

In resume, this work presents the Osmosis Processes concept and its associated modeling challenges for the liquid-sensing enterprise. The objective was achieved by following the three-layer paradigm based on MSDEA approach, which the supports potential coordination and cooperation between multi-disciplinary teams. It starts by defining the process application goals, to identify the activities for each world, in a business and technical language, ending with the osmosis process execution.

The modeling tool was adapted from MSEE's project results. In (Marques-Lucena et al., 2016) was presented a first version of the OSMOSE Toolbox explained the changes made in the MSEE's tool to follow the need to model the osmosis processes concept, namely the interactions between worlds, and the middleware membrane decision logic. In this new version, the objective was to improve the experience of the user, by giving the option to specify the worlds in the EA* model, facilitating with this approach the redesign in the BPMN2.0 model. For that, it was needed to change the transformation code, improving the transformation and accelerate the design time. These changes allowed to include in the business process model a more detailed information about the involved worlds. It facilitated, technical teams, with their knowledge about the osmosis worlds' concept and technical modeling skills, to enrich the business process model with the osmosis behaviors and constraints. This new version of the Toolbox improved user experience as well as the integration between the business level and the technical level. The notion of the worlds in the transformation rules improved the resulting BPMN process, causing the user to make fewer changes in it. At the same time, due to the fact that the Toolbox follows the MDA paradigm, it gives the capability (at the design phase of the processes models) to re-adapt over time, allowing to evolve when occurring a change in the process or service, which need to be changed.

6.4.2. Simulation of a Device Network

In Chapter 6.3.2 it was presented the Semantic Maps work to represent mapping to relate different devices inside of a device network, giving the capability to the network to recover interoperability whenever a device is malfunctioning. In here, it is presented a continuation of that work, presenting a new feature that is the simulation of the device's network. To achieve this, it is necessary to simulate an environment the closest possible to a real scenario, to provide a reliable idea of how this solution is viable. With that in mind, it was mentioned before that the sensors represent the lower layer of the architecture and do not depend on the monitoring system, therefore should be independent of the rest of the implementation. As (Theunis et al., 2017) define, in their work, the sensor is, only, the sensing element that transforms an external physical property into an electrical response. In the simulation, the sensors are represented by independent and parallel threads that output measurements, to which the monitoring system may consider or ignore, similarly to a real network. These sensor threads use the information provided by the database files and function based on that (active when the sensor is on, suspended when the sensor is standby, etc.). There is no significant logic or decision-making, like a regular sensor normally operates, except for the output measurement that varies depending on the input parameters (it can be randomly generated or it can be a selected value, being giving this decision to the user).

The Resource Manager do the registration of the devices and consequent description of the device network was made. Having thus a notion of how the network is represented in the real world, thus achieving the ability to do the virtualization of the network. With the support of a CEP, in this case, Esper was used (see discussion in Chapter 3.3), since it facilitates the setup of a fault detection methodology. First, the configurations for managing the complex event processing are set accordingly to the scenario. After that, the EPL statements are defined, using keywords to filter the type of event and time window that triggers the statement.

Depending on the device to be simulated, it is possible to change the values that the device is acquiring in order to see the behavior that will occur in the network. So, after defining which device it wants to trigger, it is made a definition of the statement and created an event listener for each specific statement. With this method, each fault detection instance is associated with an ID and, possibly, other data properties, similarly to the devices, and can be related by a class property with the devices. The beneficial points are to consider these methods inside the KB and DB, allowing direct correspondence to the sensors, allowing easy and automated manipulation of the conditions during the implementation (due to being present in the always updated knowledge bases and not in the initial programming code) and allowing the creation of typical fault detection profiles such as a "*fault detection instance for temperature sensors for fire alarm roles*" (which can be directly associated with a new sensor in the network, instead of manually creating a statement). Of course, this method has predictable disadvantages, and the most significant ones are the fact that demands the inclusion in the initial design of the KB model (which may be a time-consuming process, relative to the use of Esper) and the runtime processing time which is surely greater than using the optimized components of Esper.

The simulation of the devices, it is a good feature since it gives the testing aspect to the network, instead of using real physical devices. The use of virtual sensors provided a better manipulation over them, allowing a better testing process. The errors and failures, induced into them, were based on real-world situations and the sensors represented individual and independent threads. With this in mind, the author believes that the application of the semantic mapping process to physical sensors does not bring any advantage over the developed solution because the module only focuses on events.

6.5. Phase 4: Runtime Configuration

The last phase of the Design Mode is the Runtime Configuration, being the finalization and confirmation of the configuration made so far (see Figure 6-12). Basically, this phase is represented with parameterization of the services, which are needed to be used in the execution, and then connect the driver to the network. These services can be web services to interconnect the change of information, the services related to the data and devices mappings (already discussed), the definition of events to be used during the monitor phase, the transformation rules to be used in the exchange of data, etc. At the moment, the user closes the parameterization of the services, the framework connects the driver to the network, thus ending the design mode and the network is prepared to start the Runtime mode.

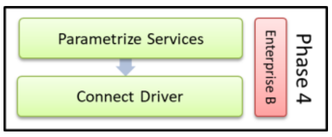


Figure 6-12: Phase 4 Runtime Configuration Methodology.

6.5.1. Osmosis Process Development Framework – Process Deployment

This chapter is a continuation of the work described in Chapter 6.4.1, where was presented the work developed in the OSMOSE project, being the focus in the process design. At this point, it is described the next phase of the framework, the Process Deployment. At this stage, the problems encountered were identified and solutions were presented to the user to give their endorsement. For example, a change was found in one of the models and the framework presented a solution for a mapping evolution, or else one of the devices broke down and a semantic mapping was presented for a device that can do the same work, although being different.

Looking at Figure 6-9 (green square) of the work done in the OSMOSE project, the next steps that are related to the work presented in this chapter are the Specification of Services and Service Implementation, being described in below and to help describe this phase:

- Specification of Services: From the BPMN process definitions service specifications are deduced which are needed to enact the processes resulting from the model transformation, working in parallel with the OTIM level. The input/output parameters are defined. On the other hand, additional requirements are specified for the input and output of the service together with non-functional aspects like for example expected execution time or costs etc.;

- Service Implementation: The services for the provided service specifications are implemented in this step. This might mean that services are implemented from scratch using some service development toolkit or, in case of legacy components, which wrapper services are implemented which map the input/output of the service specification to the input/output which is provided by the legacy component;

The Process Deployment is made to support process execution. In our implementation, the jBPM (Del Fabro et al., 2009) environment has been selected, since it is an open environment and is widely used by the community. A straightforward manner to start process execution is to use the jBPM Web console. This step represents the Code level in the model-driven paradigm. The processes from the Toolbox are transformed into BPMN processes that are uploaded into the jBPM repository, from where they are deployed in the jBPM process execution engine. In the jBPM execution engine, the processes can be further refined (OTSM) and executed when they are eventually completed.

During the preparation of the jBPM to execute the processes, it is necessary to specify and implement the services in order to get or set data used during the monitoring processes. The service part of the design framework can be handled in parallel with the OTIM and OTSM definition. These services are registered in an enterprise service bus being available to entities in the OSMOSE architecture to invoke process execution of these services. With this approach, the services are available anytime to be used in the processes, allowing the system to have two types of services: a) Services for invoking process execution; and b) Services for delivering messages to specific processes which are already in execution. Indeed, the specification of user and service tasks begins to be detailed in the BPMN model at the OTIM level (see Figure 6-9). Then using a standard IDE (Integrated Development Environment) is possible to generate the skeleton of the code to be applied on the service tasks, which then needs to be finalized using the usual programming rules and approaches.

These services have to be specified and implemented mapping the input/output of the service specification to the input/output of the process. To better understand how this works, follow the use scenario validation described in the Chapter 8.2.5.

In summary, this work represents the continuation of the work presented in Chapter 6.4.1, representing the deployment of the processes in the framework, and consequently the starting of the Runtime Mode. With the support of this tool, the process design and deployment is accelerated and more accurate, since this is divided for several steps that allow to the user to go back to do the changes needed and create again the processes. With this capacity, it is possible to improve the deployment and the running processes, since the user is able to identify problems, change it on time, and deploy a new version of the process, without stopping the processes and avoiding possible problems that were identified before it creates the harmonization breaking.

6.6. Analysis of the IoT Framework Design Mode towards Hypothesis

With the specification of the Design Mode of the IoT Framework and the description of the four phases that belong to it (as illustrated in Figure 4-1), a step was taken to be able to answer to the hypothesis presented in this dissertation, since the Design Mode contribute to reaching the interoperability inside the collaborative network. The design mode was developed to identify ways of creating interoperability in the collaborative network, both at the EIS level and in the devices. The first phase extracts knowledge from the data models, identifying their structure. This knowledge is used in the second phase that is responsible for specifying the mappings to be used in the data, starting to create the interoperability between the enterprises, thus achieving the ability to exchange data. At this point the mappings are defined and it is possible to do simulation to identify possible problems and be able to correct them before they occur and allowing the user to model the processes to increase the know-how by the platform, increasing the capability to react since the interoperability is increasing, this modeling and simulation correspond to the phase 3. The last phase is to configure the runtime, thus finishing the changes that are occurring in the collaborative network and thus responding to the hypothesis, in the part where it says that it is possible to recover the interoperability.

7. IoT Framework – Runtime Mode

After the framework has identified all the models and devices in the network and being ready for execution, then it goes to the next mode – the Runtime Mode. This mode is responsible for running communication between enterprises, monitoring them to identify harmonization breakings, and identifying the level of interoperability in the network. It starts by describing the Runtime mode, then as described in the SSIF framework of Figure 4-1, this mode is divided into two phases, the Phase 5 (System Evolution) and Phase 6 Interoperability Assessment, being the next two subchapters.

7.1. IoT Framework – Runtime Mode Description

This is the second mode of the IoT Framework, the Runtime Mode, being responsible for restoring and maintaining the interoperability harmonization. This goal is achieved through two different tasks, restoring the harmonization breaking and monitoring the network searching for changes. In Figure 7-1 a study to identify the needs of this module was made, identifying the features needed to reach the goals for this module.

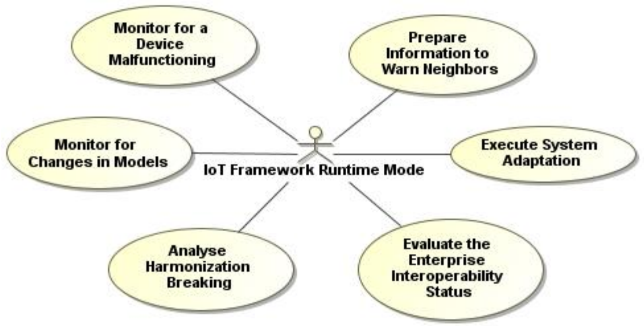

Monitor for a Device Malfunctioning
Prepare Information to Warn Neighbors
Monitor for Changes in Models
Execute System Adaptation
IoT Framework Runtime Mode
Analyse Harmonization Breaking
Evaluate the Enterprise Interoperability Status

Figure 7-1: IoT Framework Runtime Mode Use Case Diagram.

This means, at this moment the network of enterprises and devices is now interoperable and exchanging messages with each other. To achieve this status, two phases were identified to be developed for the framework (following the SSIF framework of Figure 4-1), as illustrated in Figure 7-2:

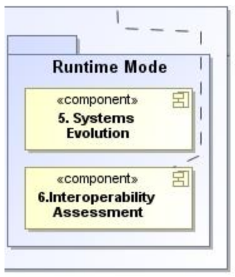


Figure 7-2: IoT Framework (Runtime Mode) Architecture.

- **System Evolution** – This module is responsible for two main tasks, the adaptation of the network and the network notifications, being able to adapt the enterprise's network in order to maintain the interoperability status. It is necessary to have a system that adapts to the diversities that it encounters and being able to learn over time. The self-adaptation module needs to have a proactive adaptation following the works presented in Chapter 2.3.3. Looking at the framework this module is placed in the Adapt phase, including the Self-Adaptation. In a network of enterprises and devices, if we consider each one a separate system, it can be seen the network as a MAS. With this method will be possible to have an enterprise as a set of intelligent agents (internal problems are detected and the agents will adapt, adequately reacting to them) with a holistic view on the network (where the individual may inform the partners about the changes so that they can be used by the others to avoid similar situations). If the framework transmits the problems that occur in each enterprise to the network, the others can simulate this problem and check if it has an impact on their system, preventing this problem from occurring in the near future;

- **Interoperability Assessment** – This module is responsible for monitoring the EIS and devices of each enterprise within the network searching for changes in the models or devices, which can have an impact in the network, creating a harmonization breaking. This work is in line with the one discussed in Chapter 3.3 being applied in here, the system monitors for a local monitoring of the EIS, and the network monitor for monitoring the enterprise's network, supported with the agents and events technology. Looking at the framework this module is placed in the Monitor phase, including the step Monitor. At the same time, it is responsible for identifying the level of interoperability of an enterprise, aiming to verify what level of interoperability of the enterprise, namely, it is possible to know what it interoperates with others, in order to support the system to maintain sustainability in the network In Literature, several interoperability layers and maturity levels exist with the purpose of assessing the interoperability status inside an organization or specific system or network. This module is responsible to categorize the desired interoperability levels and evaluate the EIS and devices of each enterprise within the network.

7.2. Phase 5 – System Evolution

This phase is responsible for restoring interoperability in the network, turning the network sustainable. To enable the system with this capability, it needs to be adaptive, in other words, it needs to adapt every time that a problem arises. This is possible by identifying what it is causing the harmonization breakings to wreak havoc during the minimum possible time, thus recovering the network interoperability. In the last phases, it was identified the harmonization breaking and how to solve the problem of restoring it. In this phase, these changes are applying to the network to re-adapt the network and recovering the harmonization. On these situations, shows that the intelligent module of the framework does not contain all the possible answers to the problems that may arise. In these situations, the system learns with the solution in case of similar situations happens again, can be reused.

Another feature is to give the notion of network, giving the capability to communicate with other enterprises in the network. This notion of the network is also used to be able to evaluate the interoperability of the network and to look for solutions to propose to the user (if possible) solutions to improve the interoperability on the network. The notification of the network follows the work described in Chapter 3.3, being used a MAS network to warn the others and recovering the harmony status in the network. In Figure 7-3 is possible to see the result of this study, being this phase constituted by two steps, the Execute System Modifications and the Notify Network.

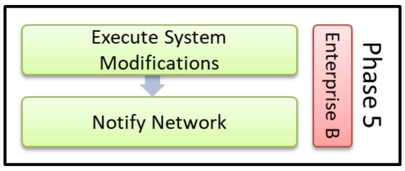


Figure 7-3: Phase 5 System Evolution Methodology.

7.3. Phase 6 – Interoperability Assessment

This is the last phase of the SSIF framework illustrated in Figure 4-1 it is responsible for three tasks, the monitoring of the enterprise and the enterprise's network, the identification of the harmonization breaking and the interoperability evaluation. In Figure 7-4 the methodology related with the Interoperability Assessment is presented based on the description made so far. As enterprises tend to protect the data, remaining “isolated” from the rest of the world, creating a kind of Island Syndrome. The symptoms of an Island Syndrome are an isolated population that influences their behavior, reproduction, demographics, and morphology (Adler & Levins, 1994). Comparing these symptoms with an enterprise, an enterprise is an island in the sense that works isolated from other enterprises, without sharing details or any genre of knowledge, influencing their behavior when they are together, with the culture, language or different models, taking thus a consequence of the data to send, resulting in problems in the collaboration network.

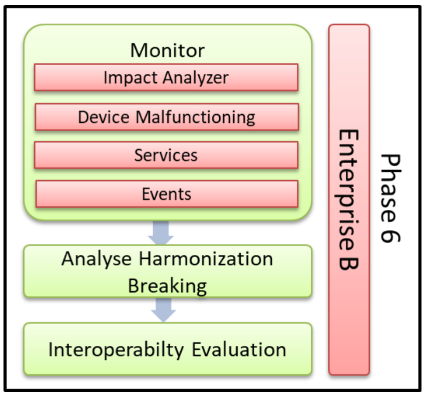


Figure 7-4: Phase 6 Interoperability Assessment Methodology.

Therefore, as an enterprise suffers Island Syndrome, for not wanting to share data, or at least share the minimum possible for the collaboration to works. This work was careful to take into account that each enterprise is isolated, so monitoring is done internally. Each enterprise has their own system that makes the monitoring of models, devices, events, and processes whenever it detects a situation that may cause problems for the network guard this occurrence to be assessed internally.

When detecting a problem with a model or a device and solve it internally, avoids an internal interoperability problem. However, this solution alone does not mean that nothing occurs on the network. On the contrary, a change which entails the exchange of information between enterprises break (creating a harmonization breaking), since this information at this time may have a different structure. To resolve this situation, it is needed the network to become social and cooperative with each other, being able to identify their problems and warn the neighbors aiming to break interoperability in the network, to reach this point the network adapts, becoming sustainable. For this reason, in Chapter 7.2 the network communication was presented, in which it is once again demonstrated its importance. It is necessary to communicate among the various enterprises in order to identify situations that can generate harmonization breaking.

As the system must monitor various aspects, some of which are not similar, it was necessary to identify which situations and how to solve these problems. In the case of models, at this stage the system already knows the models, making this module to be responsible for monitoring possible changes in the models, namely whenever a change will make the data exchanged between the enterprises to stop working, creating communication problems. In summary, it is necessary to identify a way to get monitor models from an enterprise in order to find changes to avoid network problems.

As this work is focused on models, in the case of the services monitor is done in situations where services fail due to changes in the models, which have arisen and can create problems. In part, this monitoring is related to the monitor of models, since whenever there is a change may have consequences in services, whenever a problem is identified in a service, it is sent to the SSIF framework and Phase 5 is called, an example of how this recovering of the services are made is the work presented in Chapter 7.3.2. The events in this context act as KPIs to monitor the interoperability of the network, these KPIs support to track and analyze the streams of information and monitoring whether the data will impact the system creating a harmonization breaking. An event to activate may detect a possible problem and give a real-time response to detect a harmonization breaking and to identify their type. In this way, it is possible in real-time to detect problems and have an answer at the same time, in the case of knowing a solution for the problem.

At the level of system devices, the monitor is done by looking at the data, identifying communication failures and failures in the devices. At the Data level, the system monitors the structure of the data and verifies that it is all being sent correctly (looking at the data model and verifying that there were no changes) as a data packet received with changes causes there are failures. At the communication level, the system checks if the device is sending the data, identifying if there was a miscommunication, this is done with the support of the processes described above. Finally, it is a device level, detecting if a device fails, in a case a device fails is possible to identify it and warns the user about it, allowing repairing it.

So far, it was described how the monitor of the network is made, being missing how it evaluates the enterprise network. This evaluation is made in different steps, first it is made an evaluation at semantic level of the models (to the system identify patterns relating to the ontologies, and thus facilitate the integration between models), then the interoperability is evaluated in the models (at this point is given a metric to models to be known the level of interoperability in the models, at this point is possible to know if a model is able to interoperate with another model or not), after finishing the evaluation to the model and verified if the model is interoperable, it is made an evaluation of the model structure (to identify its structure in order to be compared with other model and thus verify the connectable between each model), the final part is to display a map with the evaluations for the user to validate.

7.3.1. Monitoring Morphisms to Support Sustainable Interoperability of Enterprise Systems

In Chapter 6.3.1, MIRAI was described as a solution to represent the mappings between different models, in order to facilitate the exchange of data between them. In this chapter, MIRAI is again addressed, but to demonstrate how MIRAI maintain the interoperability throughout the life cycle of the enterprise network, to monitor the model mappings. Here, it is explained how MIRAI monitor the mappings to identify possible harmonization breakings.

As already explained, MIRAI is divided into several blocks, in which one is responsible for a different task. For the monitor task, the block in charge is the MIRAI Intelligent Supervisor Block, since this block only purposes the users for solutions (Ferreira et al., 2011), needing for a confirmation of the user, for this reason, the MIRAI Administration Block is also described, which is the block that interacts with the user. As a result, it was created an architecture represented in Figure 7-5, divided into four important blocks that define the framework:

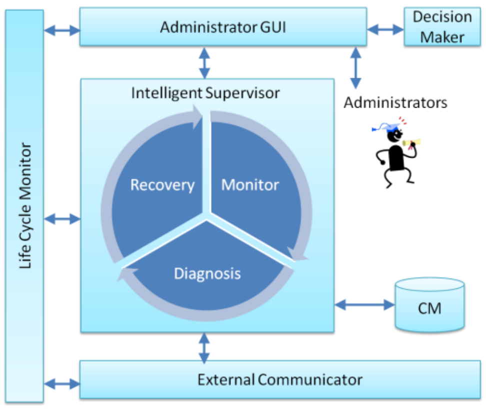


Figure 7-5: MIRAI Architecture.

<u>MIRAI Intelligent Supervisor Block</u>

This block is responsible for the detection of the harmonization breaks by performing a scan in the organization’s CM in search for new morphisms or morphisms evolutions. For the system reacts to a recovery from an evolution of a morphism have two agents involved, the Agent Monitor Mediator and the Agent MoMo. They are responsible to search for changes (Monitor) and then to check if the changes occurred to have an impact in the system’s interoperability status (Diagnosis), and if it does, it proposes a possible solution (Recovery). To achieve this state, it is crucial to understand how the two agents work:

- **Agent Monitor Mediator** – This agent detects changes in the CM, being responsible for the detection of the breaks in the harmonization. Basically, it does a scanning in the CM to search for changes in the mappings, this is made by finding an evolution of the same (versioning's). Every time it finds a new mapping it warns the Agent User and the Agent MoMo. This agent needs to do two types of communications with other agents, one type is to inform about what happens, and the other is to ask the Agent MoMo to present a solution to the founded changes in the CM, these communications follows the FIPA protocol;
- **Agent MoMo** – This agent acts when requested by the Agent Monitor Mediator. It is this agent that makes the decision in the MIRAI. The main objective is to check if the changes occurred to have an impact in the system interoperable status, and if it does, it warns the Agent User and proposes a possible solution (i.e. a new morphism or a correction to the existing one). This new mapping is done by using MapT, comparing the two morphisms in use, and using rules in the decision of the matchclass. The communication used between Agent MoMo and Agent User is again FIPA protocol.

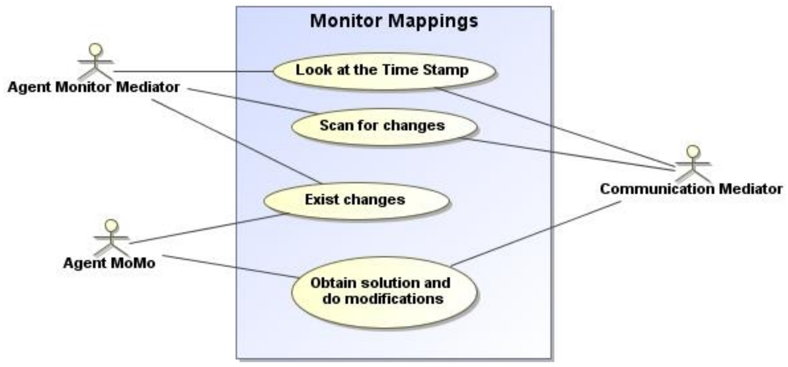


Figure 7-6: MIRAI High-level Interaction.

In Figure 7-6 is illustrated the workflow of this block, it is represented at a high level and demonstrates how the agents interact with them. In this figure is possible to see that the agents are autonomous of each other, but need to cooperate between them to aim for the main objective, and how the Intelligent Supervisor reacts to changes in the CM, it begins in the Agent Monitor Mediator, which look at the time stamp until it finds changes. Then it will scan the CM to search for new mappings, every time that finds one it will call for Agent MoMo, this is the routine of the Agent Monitor Mediator, repeating this process once a day. When the Agent MoMo is called, it will evaluate the received changes and propose a solution to the user. Then the user will accept or not that suggestion.

Since the user can refuse the proposal presented, the system will react and ask the user to change the proposal to be like he wants, then Agent MoMo can do the update of the CM and ask the Agent Communicator to send a message informing about the updates to the Network. Later, it will be described how the Agent MoMo reacts whenever it is invoked.

Until now it was explained how MIRAI system detects changes within the morphisms a network of enterprises, while here it will be explained how the Agent MoMo generates a new morphism to respond to a harmonization breaking situation. Thus, every time that a versioning is detected by MIRAI, it is required to evaluate the morphism and generate a new mapping to recover the interoperability status. Figure 7-7 illustrates the process of how the proposing this new morphism is calculated, is described using a tuple (MapT) represented by Equation 4-1, and presented to the user.

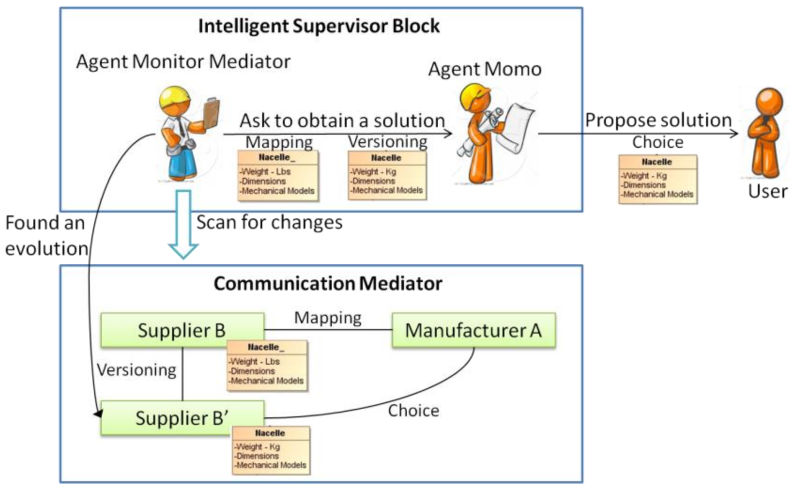


Figure 7-7: Detection and propose of morphisms.

To purpose a new mapping, it is needed to choose the best solution for the new mapping, since the mismatch can change depending on the type of map. Enabling the system to adapt to the environment and to be able to recover the interoperability status, to achieve better solutions and a more accurate system, therefore it was decided to use a learning capability. This learning capability has the aim to analyze new changes, and propose a solution to those changes, based on the past decisions, while progressively updating the CM. With this is possible to provide a more educated mapping adaptation that will fit better the user's needs, facilitating its decision and minimizing the transient periods. This learning ability is based on weights that help in the choice of MatchClass for the new mapping and increasing intelligence over time.

This implementation was made to help the MIRAI in the proposal of the new morphism, with the aim of learning with the choices of the user and in time to avoid poor decisions by MIRAI. Every time that an evolution appears the MIRAI will propose a possible solution to the user, but it could happen that the user does not agree with that, and changes the proposal. The intention in the use of the learning ability is so that MIRAI learn the decisions of the user and use it in the future. This is made using a "weight", in the CM a list with all the possible cases exists, and every time the user changes the proposal, the weight of the user choice is incremented. So the next time the MIRAI will choose the one with the higher weight, gaining a more reliable decision.

MIRAI Administration Block

The Administration Block is where the user does all the decisions in relation to the mappings, and it is possible to see how the MIRAI is working. This is achieved with the support of one agent, the Agent User. This agent is the interface between the MIRAI and the human user. Through this agent, the human is informed of the solution proposed by the Agent MoMo to decide whether if the user accepts it, or propose a new one. This agent communicates with all the other agents since the only proposition is to be informed except when the user has to choose a solution, which was decided to use the FIPA protocols. Being almost all of the messages informative, for example the Agent Monitor Mediator send messages informing about what it is doing, creating a log of all the steps, the Agent Persistor is sending messages every time that it finds a dead agent or a reborn one, the Agent Persistor Police is the same but only in the specific case of Agent Persistor.

MIRAI Network Block

This block is responsible to warn the other members of the network about a possible problem. It is said 'possible', because it can be an internal problem to the enterprise, but does not cause problems in the rest of the enterprises, so the network is advised so that they can check if it has an impact or not. In the case of creating impact, the problem is identified and solved, otherwise, the message is ignored. To create the notion of network enabling the communication between the different enterprises, it was used a Multi-Agent System (MAS).

Inside the scientific community exist several definitions to describe an agent, Wooldridge, and Jennings in (Wooldridge & Jennings, 1995) distinguish two general usages of the term agent: the first is weak and the second is stronger. In this work it will focus on the weak notion since this notion is a part that is incontestable within the community:

- **Autonomy** - agents make decisions without human intervention, these decisions are made with some kind of control over their actions and internal state;
- **Social ability** - agents interact with other agents via an agent communication language;
- **Reactivity** - agents react to changes in their environment;
- **Pro-Activeness** - agents have their own goals and besides reacting, they are capable of initiative.

Taking into consideration these descriptions, in particular, the features of social and reactivity, it is common that the agent is capable of interacting with other agents, humans, or with the surrounding environment. This brings something new to the software technologies, i.e., communication and teamwork between software, in this case, between agents, and this is called Multi-Agent System (Bellifemine, Caire & Greenwood, 2007).

The great advantage on using MAS for the implementation of this framework is that they are capable of cooperation, collaboration, negotiation, etc., and they understand each other via an agent communication language based on speech act (Bellifemine, Caire & Greenwood, 2007) thus avoiding agent's interoperability issues. Another feature of the MAS is the capability of restoring an agent in the case of that agent goes down, creating a disruption in the communication, since one of the enterprises is not accessible. In this case, by restoring the agent it turns again the enterprise visible to the network. Therefore, due to these features, MAS is being used in different areas, from industrial applications to telecommunication and multi-robotic systems.

In this work, three agents were developed to validate the described concept these agents are divided into two blocks. The first block is the External Communicator Block, which it is responsible for the communication between the MIRAI's, it is composed of one agent, the Agent Communicator. This agent is responsible for the communication between different MIRAI's of a collaborative network. This agent has some tasks to do, one of them is to inform the Network every time that exist changes in the CM, thus enabling them to react as well. Other is to have the notion of the network, for that it has a type of yellow pages where it can find his neighbors, allowing to communicate with them, every time that it needs it.

Since this block only takes care of the communications between the MIRAI's, his only purpose is to send or receive messages from another MIRAI's. These communications are made using web services provided by JADE[19] and the web service is known by the name of Web Service Integration Gateway (WSIG). The WSIG is an agent that sends and receives messages from other agents, do the subscription of the agents in the Director Facilitator and publishing the service in the UDDI (Universal Description, Discovery, and Integration) registry. The agent receives from the Network news about changes that occur in another MIRAI and then will send that change to the Agent Monitor Mediator to see if the received information will have an impact on the system. Then if this information creates some changes, the Agent MoMo will ask the Agent Communicator to warn the Network about that, all this is made with web services.

The second block is the MIRAI Life Cycle Monitor Block, which it is responsible for the maintenance of the MIRAI interoperable, if one of the agents goes down, the harmonization of the systems breaks. For example, if Agent MoMo goes down, it is impossible to have a diagnosis and a recovery of the system, so if a new mapping occurs the MIRAI will identify the problem, but will not propose a solution, so in this case, it is not doing his job. Because of this, it was added an agent to monitor the MIRAI for the case that an agent goes down it will recover him, avoiding a harmonization break. The Agent Persistor Police exist to assure that the Agent Persistor does not goes down and continues to do the monitor of the system. The two agents of this block are described here:

- **Agent Persistor –** This agent controls if any agents go down, to recover them if required. Since it will be responsible for the restoration of the down agent, something to help in that manner is needed, so the JADE provides an agent that does that work, and that agent is the Agent Management System (AMS). This agent receives the registration of all agents while managing their life cycle;
- **Agent Persistor Police** – This agent has a similar role with the Agent Persistor, but only controls the Agent Persistor, thus adding some redundancy to the MIRAI.

MIRAI is a solution to monitor all the mappings that exist among the several models used by business partners in the same collaborative network, controlling the changes, warning and proposing new mappings. It was presented the communication and recovering part of this work, explaining the advantages this implementation, which brings several benefits to the network, by sharing the problems that each enterprise find over the life cycle time, enabling the improving the response to the harmonization breaking. At the same time, allows identifying situations that an enterprise goes down of the network, enabling the identification of that enterprise and recovering it in the moment.

[19] http://jade.tilab.com/

The role of MIRAI is to monitor all the mappings that exist among the several models used by business partners in the same collaborative network, controlling the changes, warning and proposing new mappings. Thus, preventing interoperability problems that could cause a destabilization of the network harmony. Enterprises' privacy is assured since each one has its own MIRAI associated to an internal CM that tracks the morphisms it maintains with their direct partners.

Such solution facilitates the creation of a network sustainable interoperability. This means that the systems are self-organized and capable of responding to environment changes, and network evidence system of systems behavior. Systems' communication with the environment allows all necessary exchanges through its own boundaries, even though transformation mechanisms are influenced by internal rules, values, beliefs, constraints, culture, and internal models. This dynamicity is essential to support and sustain the interoperability applied to global business networks.

7.3.2. Methodological Framework for Detection of Harmonization Breaking in Service Environments

This works aims to further explore the occurrence of harmonization breakings in interoperable environments as well as different types of approaches in order to monitor and detect such events, by presenting a proposal for a methodological framework for the detection and possible system recuperation of harmonization breaking events in service environments. To better exemplify the problem a scenario is presented in Figure 7-8. The discussion is centered on a scenario where a consumer (C) wishes to exchange information with a provider (P). Before communication can be accomplished and the exchange of data takes place, the process that describes the interaction between the two cooperating entities must be described. This consists in specifying every single interaction and procedure that occurs between both and structures them accordingly. As mentioned before, this is achieved by means of service modeling and orchestration and results in a working business process that, after being deployed in the proper server engine, performs the described tasks and enables the interaction between both enterprises. Naturally, because it is C who wishes to access and consume the services made available by P, the business process must be deployed within its control and management, assuming that a centralized entity responsible for such tasks does not exist. Thus during this time, and supposing that no other issues occurred, the business process operates normally in a stable environment allowing the exchange of information between the related entities in an interoperable way, as illustrated in Figure 7-8.

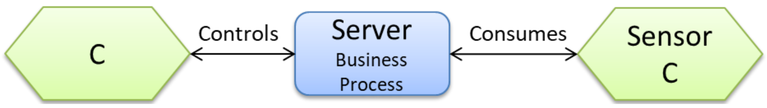


Figure 7-8: Example of a Consumer (C) interacting with a Provider (P).

However, if for some given reason the communication between C and P were to fail, an exception would raise within the business process in consequence of its inability to reach its destination. Generally, assuming that it is possible to catch and handle this exception, then it is also possible to identify the disrupting issue and take the proper steps to notify C about the inability to reach P while alerting for the eventual need of human intervention. Nevertheless, in large networks such as a dynamic managing network, where several consumers and providers communicate simultaneously and depend on each other to achieve a common goal, this type of solution could temporarily hinder the system's production capability, as part of the system would be stalled until a proper solution were to be implemented and the connection to P restored. Therefore, during this transitory phase in which interoperability does not exist and needs to be reinstated, the system would unavoidably suffer setbacks.

While the inability to communicate with P could be due to some unsolvable situation, from C's point of view, such as a physically broken cable or other similar condition, it could also be as simple as an incorrect mapping during the invocation of a given WS due to an unexpected change in the service's interface. Thus, in order to properly evaluate the cause of interoperability disruption, this work suggests that by incorporating into the system an external intelligent component, which were to be invoked once an exception was to be raised within the business process itself, it would be possible to assess and analyze the situation from an exterior standpoint while also enabling the capability to act and alter the faulty business process. In this sense, a methodological approach was conceived in order to properly tackle harmonization breaking events that may arise as a consequence of similar scenarios, and in this manner contribute to reducing the transitory phase that naturally occurs between the state of network instability and interoperability.

Following the scenario introduced in Figure 7-8, in here it is introduced a methodology that seeks to approach the issue of harmonization breaking events in interoperable service-based networks. The main objective is to design a methodological approach that can effectively reduce the time required to establish interoperability after a harmony disruption has occurred, by resorting to automatic interventions that directly act on the faulty procedures and in this way avoid the need for human intervention. The process is centered on the capability to properly handle exceptions that occur during normal business process operation and subsequently invoke an external intelligent component that enables to directly approach the cause of the error. Thus, it was considered that if such an implementation could be achieved then three different outcomes would be possible, as illustrated in Figure 7-9.

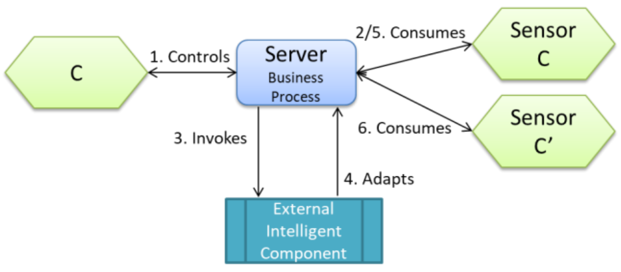


Figure 7-9: Methodological steps to approach harmonization breaking.

As already described, the consumer (C) controls the server where the business process that runs the interaction between C and the services provider (P) is deployed (1). If a harmonization breaking event occurs when consuming the services provided by P (2), then an exception within the business process is raised and handled by its own error handling capabilities, after which it proceeds to invoke an external intelligent component (3) whose objective is to evaluate the cause of the issue and correct it by adapting the business process according to the newly introduced requirements (4).

In a primary intervention, the external intelligent component would proceed to analyze the situation and attempt to restore the connection to P by directly modifying and automatically re-deploying the business process in the respective server engine (5). If there were no success and a process adaptation could not be implemented, then a secondary intervention would be attempted by searching the network and trying to connect to a similar provider (P’) with the same characteristics and services as P (6). If an alternative provider were to be found, this solution would also require the business process to be adapted in order to properly reflect the necessary changes and start interacting with P’ and disregard the previous service provider P. It should be noted, that this particular step can only be achieved in very specific situations. To be successful, the network must be prepared for such modifications since its initial construction and design. Furthermore, not all domains can be considered. A service provider that supplies consumers with weather information, for instance, represents a service somewhat simpler to replicate and seek for in case there is a need to find an alternative since the information it provides is rather generic.

Finally, if none of the two interventions mentioned could be accomplished successfully, then no automatic solutions would be possible and a request for human intervention would inevitably be required. With this in mind, a methodology was devised to reflect the necessary steps to be implemented in dynamic service-based networks when seeking to sustain interoperability, as depicted in Figure 7-10.

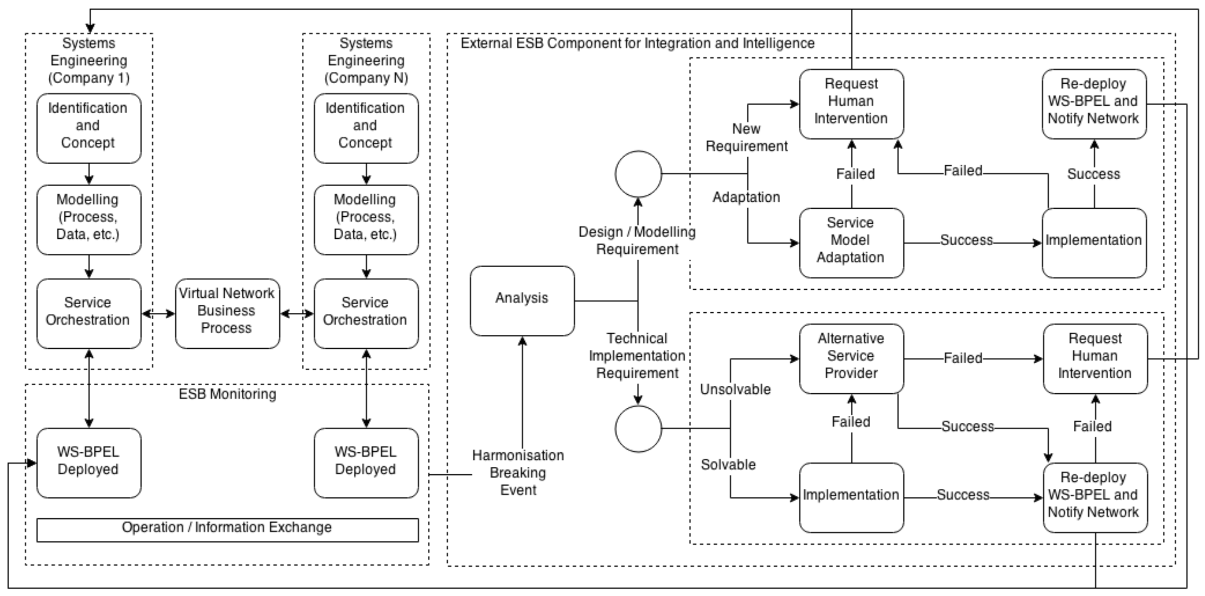

Systems Engineering (Company 1)
Identification and Concept
Modelling (Process, Data, etc.)
Service Orchestration
Virtual Network Business Process
Systems Engineering (Company N)
Identification and Concept
Modelling (Process, Data, etc.)
Service Orchestration
ESB Monitoring
WS-BPEL Deployed
WS-BPEL Deployed
Operation / Information Exchange
External ESB Component for Integration and Intelligence
Analysis
Harmonisation Breaking Event
Design / Modelling Requirement
Technical Implementation Requirement
New Requirement
Adaptation
Request Human Intervention
Failed
Service Model Adaptation
Success
Implementation
Failed
Success
Re-deploy WS-BPEL and Notify Network
Unsolvable
Solvable
Alternative Service Provider
Failed
Request Human Intervention
Failed
Success
Implementation
Success
Failed
Re-deploy WS-BPEL and Notify Network

Figure 7-10: Methodology for the sustainability of interoperability in service-based networks.

The methodology starts by delineating a procedure of SE in which the concept of the process to be developed must be properly identified and modeled which, in respect to the subject of this research, implies the modeling of data, services, and others. Service orchestration is what allows taking these modeled concepts and structuring them in such a way that it enables the establishment of interoperability between different enterprises. It is achieved through the formulation of a business process that regulates the tasks and interactions that occur during the exchange of information between collaborating entities. The relation between the business processes across different enterprises constitutes a virtual business process to the network that represents the interactions that take place from enterprise to enterprise. Although required for the methodological approach here proposed, these SE concerned steps are not obligatory for every enterprise. However, only the enterprises that choose to implement them will be able to sustain interoperability according to the proposal here described, which means that if an enterprise that has implemented these methodological steps is communicating with a partner that has not, the former will still be able to activate the recovery process as described. Although still valid, this can increase the difficulty to restore interoperability as one of the enterprises cannot adapt to the new requirements, and thus result in a possible setback to the overall recovery process or even the automatic methodological exclusion of the rogue entity.

Once the SE procedure has taken place, the developed business processes are then deployed in the ESB, which contains the server engine that allows operating them. It is during this time that process monitoring takes place and is of greater relevance to aid in the detection of unexpected events. This can be achieved by the ESB own monitoring capabilities as well as the implementation of monitoring techniques. An effective monitoring is most relevant when working towards the sustainability of interoperability, as explained before since it allows for the detection of occurring changes and enables the possibility to quickly react to them and avoid harmony disruption.

For the purpose of this methodology no third-party monitoring software was used, besides the one already incorporated with the ESB. The focus is on the capability to handle exceptions as they occur along system operation, and consequently launch an external intelligent component to assess the situation. Thus, as the system operates and process execution takes place, eventual harmonization breaking events lead to the emergence of exceptions that are handled by the business process itself. When an exception is raised, if it pertains to an approachable issue, then the process proceeds to invoke the external component before terminating execution and warning the user that an unexpected error has occurred and that the issue is being automatically evaluated. Thus, once a harmonization breaking event takes place, the external ESB Component for Integration and Intelligence (ESB-CII), as described in the methodology, is invoked in order to assess the situation and attempt the automatic recuperation of the affected process. At this point, once the ESB-CII has analyzed the cause of the issue, three different scenarios are possible:

1. When the failure concerns a non-existing concept then no adaptation is possible, since the concept must first be created. In this case, human intervention is necessary so that the proper SE steps can be fulfilled in order for the new concept to be implemented;
2. In case the failure is due to a design or modeling issue that concerns a problem capable of being automatically approached, then an adaption of the business process may be attempted in order to integrate the new requirements into the system. However, if the process adaptation fails or the change in question cannot be taken automatically, then no automatic recovery is possible and human intervention is required;

3. In a third possibility, if the cause of the error is due to a technical issue instead, which usually does not imply an adaptation to the process's own code and occurs in the result of an internal implementation or an external factor, then it must be determined if it can be automatically corrected or not. From here, three distinct developments are possible: if the issue is solvable, then the process adaption is performed and implemented before automatic re-deployment; if the issue is unsolvable or if the former solution fails to succeed, then the ESB-CII attempts to find an alternative provider, associated with the same partner, capable of providing the same services as the original, after which the required adaption takes place and the process is re-deployed; if none of these solutions is possible or if they fail to succeed. However, then once again human intervention must take place.

If process adaptation according to the required changes is indeed possible, then the ESB-CII proceeds to performing the necessary implementations itself and re-deploying the process in the server, accompanied with a notification to the user and system manager detailing the events that took place and the alterations that were performed. After these, the user can once again invoke the business process and return to normal operation. However, in case no process adaptation was possible and the issue could not be automatically solved, then a notification to both the user and the system manager has performed warning about the disrupting error and the inability to automatically solve it.

Thus, the proposed methodology suggests that by means of monitoring and an external aiding component, it may be possible to automatically resolve unexpected issues that occur during the normal execution of a business process, and in this way reduce the transitory phase that takes place during the state of quasi-equilibrium.

To demonstrate the methodology is used the project IMAGINE, the i-platform requires gateways to make the bridge between the enterprises and itself (seen as another enterprise). Hence, the methodology above has been applied to the model and orchestrates the furniture living lab case, involving twenty different enterprises with different EIS (only selected functions of the EIS were considered for the living lab demo). These gateways are responsible for uploading and synchronizing the information provided by the enterprises through the use of WS. Each enterprise possesses its own gateway, customized according to its own needs. However, despite these automatisms, there are still some issues that need to be addressed, such as possible interoperability issues due to the usage of different data structures.

The implementation is based on an existing business process that describes the interaction between the IMAGINE platform and an enterprise that has recently connected to the network, consisting of a simple interaction where the i-platform serves as a consumer and wants to request information about the referred enterprise. The WS-BPEL, which describes the process, defines the invocation of the WS by the i-platform by mapping and sending the required parameters to the WS, which in turn, provided that the parameters are correct, returns the requested information. The data is then received by the i-platform and the process terminates.

To properly test this implementation, the process was deployed in real-time during project development, after which the enterprise entity proceeded to alter the interface of its WS in order to simulate a design adaptation of the system. The introduction of a new requirement led to a failure when consuming the WS and consequently, with an exception being raised within the deployed WS-BPEL. At this time the ESB sees an issue arising from a communication failure, a harmonization breaking event, and proceeds to properly handle the resulting exception and launching an external ESB JAVA component, responsible for analyzing and attempting to automatically adapt the WS-BPEL according to the new requirements. Thus, once the exception was handled, code analysis and process evaluation took place, which could be achieved due to the Extensible Markup Language (XML) nature of the WS-BPEL process itself, allowing to easily detect and correct previously defined inconsistencies and errors. After analyzing the process's XML, the external component detected an erroneous mapping of variables on a WS invocation and determined that the possible cause of the error was due to a modification in the WS's interface, which occurred when a previously non-existent parameter was introduced as a new requirement by the respective enterprise. The original and the new parameter mapping between the business process and the WS are illustrated in Figure 7-11.

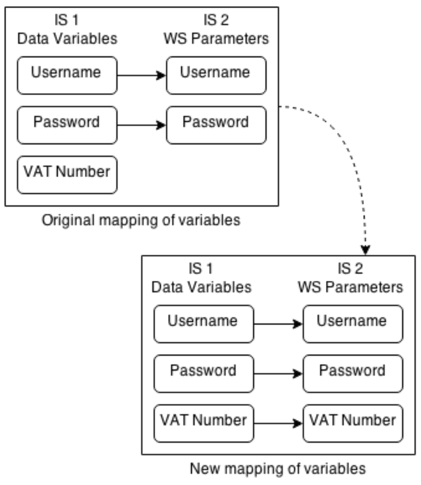


Figure 7-11: Mapping between the WS-BPEL and the WS.

Because the external component was able to relate the newly introduced parameter to an existing concept already present, a solution was automatically suggested and implemented, proceeding thereafter to immediately re-compiling and re-deploying the newly modified WS-BPEL process. On a second request by the i-platform to the given WS, the mappings were correctly performed and the process proceeded without faults, having no need to attempt a secondary solution and seek for an alternate service provider. In this manner, the external component was able to automatically recover from a break in harmonization, resulting from a need to adapt to new market requirements, effectively avoiding a disruption in interoperability that would otherwise occur due to the i-platform being unable to exchange information with the enterprise. Because the required change was considered and previously defined in the original process's XML code, a solution could be rapidly achieved and implemented by the external component. However, if further unaccounted parameter requirements were to be introduced without proper code-prepared solutions, the external component would not be able to supply automatically the WS and human intervention would be necessary, in which case a disruption in the system's interoperability would be unavoidable for a longer period of time.

In summary, this work proposes a methodological framework for the detection of harmonization breaking in service-based networks that build upon the concept of a good SE process in order to be able to adapt the system's business processes on the event of emerging requirements. Based on a simple procedure of error handling, the methodology focused on the ability to handle exceptions as they occur and consequently invoke an external component to address the cause of the error and, if possible, to adapt the erroneous business process accordingly. Its implementation is based on a centralized solution, the ESB, which offers greater control and reliability, allowing managing and orchestrating all the interactions that occur between different partners and their respective WSs while keeping an eye on what is being exchanged and handling faults as they happen. As mentioned, by individually handling each fault, each business process is thus able to invoke an external component, the ESB-CII, which consists on an application located somewhere in the system that acts as a recovering agent capable of analyzing and, if possible, automatically recovering from the issue in question.

7.3.3. Framework and Methodology for Enterprise Interoperability Assessment

In this context, an EI evaluation enables to identify the status quo of an organization in terms of readiness for cooperation and also in terms helping understand what to improve. However, seeking to evaluate any interoperability issue should be considered as a part of a thorough and methodological process rather than an ad-hoc activity. Therefore, to improve the interoperability testing and validation, an approach that views the interoperability process as the flow of information from one enterprise to another was adopted. That flow is path dependent, so one enterprise can be interoperable with another, but the reverse is not guaranteed at the same level or at all. Moreover, each step of the path refers to specific testing methods according to the barriers defined in Chapter 0 (see the Interoperability Barriers Viewpoint), defining three assessment layers.

The detailed evaluation framework realizing the envisaged interoperability assessment process, which targets the evaluation of systems according to any of the scientific areas (concerns), to their maturity and barriers addressed in a sequential flow. Each barrier has an eliminatory role, thus if the systems do not match the minimum requirements to interoperate while testing each barrier, the whole integration effort is useless. The proposed framework follows the same axes as detailed in the EI framework (ISO, 2011a), i.e. three different axes of evaluation measurement: a) The EI scientific areas that will enable to categorize the type of scientific problems, each evaluated system is experiencing; b) The maturity level to categorize the severity of the problem by means of a qualitative measurement (the lower the level, the worst is the problem); and c) The EI barriers that relate directly to the assessment layers envisaged.

Resulting in three different axis of evaluation measurement which are:

- **Axis 1: EI Scientific Areas** – The EI scientific areas adopted for the evaluation framework was a simplified version of the scientific areas listed in Chapter 0 (it is described the EI

Scientific Areas). Restricting the scientific areas to the lower level of **Error! Reference source not found.** allows decreasing the complexity of the evaluation process. As long as the levels are increasing, the complexity also gets larger making the test definition too extensive. This approach focuses the tests on low-level areas reducing the complexity and the number of tests to be defined. Since those areas are defined in a tree diagram, it will be possible to test higher-level areas by running a set of tests from a group of lower-level ones, e.g. a knowledge test results from a composition of cultural, rules, process and data testing;

- **Axis 2: EI Maturity Levels** – This axis arises as a measurement of the interoperability assessment process. For each assessment, the layer must be defined specific tests because human-related issues cannot be evaluated the same way as the technological ones. Thus, for each scientific area must be applied different maturity levels that are more appropriate to classify that area. In Chapter 3.1 are presented several categorizations of interoperability using maturity models and levels. However, none of them fits perfectly to all three assessment layers or scientific areas. In order to categorize the results of the tests executed at each layer, those methodologies must be adapted according to each case. Therefore it was decided to generalize the maturity classification to a five-level scale of values (0-4);
- **Axis 3: EI Barriers and Assessment Layers** – The EI Barriers for the evaluation framework have been defined in conformance with the framework for enterprise interoperability proposed in (ISO, 2011a) as follows: **Organizational** – It is caused by human interaction, their capability, and interest to cooperate, especially in hierarchized enterprises where the goals are heterogeneous; **Conceptual** – It is caused by different methods to represent information and knowledge, especially in complex systems with a lot of entities containing crucial information; **Technological** – It is caused by different interfaces to exchange data or a huge technological gap between the parts interested in cooperates. This axis represents the interoperability assessment flow, envisaging the advancement of the three assessment layers. This flow is instantiated by the evaluation methodology described hereafter.

To evaluate interoperability between at least two systems, there is a need of specific methodologies to test each kind of problem that derives from each EI Barrier, because human-related issues cannot be tested the same way as technological issues. During the rest of this subchapter, the tests are presented. For all the situations, in the case of the system do not have at least level 2 in both ways, it is not advisable to continue the interoperability process because then a half of the questions designed to this test have distinct results.

<u>Assessment Layer 1: Organizational Testing</u>

The human decisions and objectives are a strong inhibitor to the interoperability process because humans run enterprises and decide their goals. This way is impossible for two enterprises with distinct market views, objectives or hierarchical structure to interoperate and cooperate for a reasonable period.

To evaluate the organizational influence in the interoperability process there is need to know and compare the motivation and goals of a sample of workers from every hierarchical group from both enterprises. To do this shall be applied questionnaires designed specifically for each kind of worker to get viewpoints from all of them, and some questions directly to the organization. For example, in (ISO, 2011a) are proposed the following questions:

- **Persons** – are authorities/responsibilities clearly defined on both sides?
- **Organization** – are the organization structures compatible?

A reasonable number of questions are needed to enhance the credibility of the survey, but a number of right answers are not enough to determine the origin of the issue. To clarify the cause of the problem, specific sets of questions must be defined for the scientific areas that can affect this Assessment Layer. Analyzing Table 7-1 in terms applicable scientific areas, all can be assessed because, in a certain point, all of them are related to organizations and humans.

Table 7-1: Assessment Layer 1 - Applicable Scientific Areas.

	Data Interoperability	Process Interoperability	Rules Interoperability	Objects Interoperability	Software Interoperability	Cultural Interoperability
Organizational Testing						

The Maturity levels are defined depending on the percentage of equivalent responses as follows:

- Level 0 – ≥ 0%
- Level 1 – ≥ 25%
- Level 2 – ≥ 50%
- Level 3 – ≥ 75%
- Level 4 – = 100%

Assessment Layer 2: Conceptual Testing

Due to different enterprise market opportunities and own interests, there is a need to specify the area that both want to cooperate and later share information. In this process, both parts need to define a set of data and process, to be tested, which belongs to the previous agreement of cooperation.

Having the specific information selected, an interoperability expert shall define the mappings from one conceptual model to the other, and the reverse, taking into consideration the semantics of each model. With all the mappings defined on both directions, shall be applied the equations defined by (Yahia et al., 2012), explained in Chapter 3.2.2 and summarized in Table 3-1.

The scientific area that is being assessed, see Table 7-2, depends on the type of model in the test. In fact, it is impossible to evaluate all kinds of interoperability recurring to model analysis. If cultural behaviors and objects were represented in an information model to be assessed, that evaluation was at the data level, and the other scientific areas were going to remain un-assessed. In fact, only four kinds of interoperability can be evaluated at this layer:

- The assessment of models that represent databases or information repository will test data interoperability;
- The assessment of models that represent business processes and methodologies will test the process interoperability;
- Rules from enterprises represented in information models will be assessed by rules interoperability;
- The assessment of models that represent the concept of the software implementation will test the software interoperability.

Table 7-2: Assessment Layer 2 - Applicable Scientific Areas.

	Data Interoperability	Process Interoperability	Rules Interoperability	Objects Interoperability	Software Interoperability	Cultural Interoperability
Conceptual Testing						

After the calculation of MPI and MEI, there is a need to get the maturity level that characterizes the system. Since MEI only considers Mandatory concepts to the calculation of the interoperability ratio, it is more important to the goal of the system, and so it has more importance to define the maturity level on this barrier. Those maturity levels are defined as follows:

- Level 0 –MPI < 100%, MEI < 25%
- Level 1 – MPI < 100%, 25% < MEI < 50%
- Level 2 – MPI < 100%, 50% < MEI < 75%
- Level 3 – MPI <100%, 75% < MEI < 100%
- Level 4 – Both MEI and MPI = 100%

Assessment Layer 3: Technological Testing

This part of the testing methodology will be a validation of the conceptual testing because now there is a well-defined model that both parts agreed to follow and each implementation has to be in conformance with that agreement.

Before any kind of interoperability testing, there is a need to check the conformance of both implementations to the standard or agreed conceptual model. If any of the implementations do not pass this conformance test it will be totally not recommended to continue this process before review the first and second step because probably those applications will not be minimally interoperable or at all. To implement a conformance test platform shall be used one of the methodologies listed in Chapter 3.2.

If the system passes the conformance testing, will begin the interoperability checking itself that needs at least one qualified equipment to run the tests. To test the system an interoperability expert shall define a set of abstract test cases that form an ATS, which will be executed by the test driver that can be a human or an automatic application like TTCN-3 to test communication protocols (ETSI, 2003). Those tests shall include data sharing and service invocation from and to EUT.

Since this Assessment layer only concerns about technological tests, it will only be able to apply to a restrict set of scientific areas, see Table 7-3. It is impossible to evaluate all kinds of interoperability recurring to technological tests, only because there are a lot of inhibitors to interoperability that is not present in the implementations such as human and organizational related issues. In fact, only it is only possible to assess data and software areas because both of them are the only directly related to the implementation of enterprise systems.

Table 7-3: Assessment Layer 3 - Applicable Scientific Areas.

	Data Interoperability	Process Interoperability	Rules Interoperability	Objects Interoperability	Software Interoperability	Cultural Interoperability
Technological Testing						

The maturity levels of technological testing are defined as follows:

- Level 0 – 0% successful tests
- Level 1 – 25% successful tests
- Level 2 – 50% successful tests
- Level 3 – 75% successful tests
- Level 4 – 100% successful tests

The result of this testing phase cannot be interpreted in isolation because it is a validation of the conceptual testing, and so the maturity level shall not be lower than the one achieved on that test. If the level is higher, shall be defined another ATS that identifies the missing concepts that are detected by conceptual testing. If the maturity level is lower than expected, the whole process shall stop and be reviewed.

In resume, most of the enterprise systems are heterogeneous and distributed, and so, the main concern about those systems is the capability for them to interoperate correctly. It was proposed in this work, a methodology that allows evaluating, quantifying and qualifying the interoperability process from the initial phase to the working implementation. To achieve this purpose were defined specific tests to all three barriers that inhibit the process, such as organizational, conceptual and technological. Despite these barriers are defined separately, the test phases are dependent on the validation of the results from the previous phases. Since a group of enterprises is not able to cooperate due to their own structures, it is not necessary to spend money and time to test the models and implementations. Otherwise, in a situation that a system passes all the tests, the conceptual and technological gaps are identified, if they exist, making easier the process to find the solution path. A generic advancement scale of maturity levels from 0 to 4 classifies each test. Thus, the significance of those maturity levels depends on each interoperability barrier, because organizational issues cannot be evaluated the same way as conceptual or technical.

7.4. Analysis of the IoT Framework Runtime Mode towards Hypothesis

With the specification of the Runtime Mode of the IoT Framework and the description of the two phases that belong to it (as illustrated in Figure 4-1), a step was taken to be able to answer to the hypothesis presented in this dissertation, since the Runtime Mode contribute to reaching the sustainability of the EC, enabling the framework to maintain the interoperability during the life-cycle of EC. The phase five is responsible for implementing the changes that were outlined in previous phases, and at the same time communicates with the other enterprises that are in the network to see if those changes have an impact on them. This ability to communicate with the entire network helps in sustainability since it enables companies to check whether it is a problem or not, helping to answer the hypothesis. The last phase is responsible for the monitoring and detection of the harmonization breakings, being this phase one of the main responsible for the sustainability of interoperability in the network, since in identifying problems it is possible to react and recover the harmony again, thus responding to the hypothesis presented in this dissertation.

8. Implementation Testing and Hypothesis Validation

In this chapter, the validation of the hypothesis presented in this dissertation is validated in two principles, through scientific validation (with the acceptance of scientific peers) and through industrial validation (validation through European projects). This chapter begins with the scientific validation by presenting the list of published articles, and then by explaining the work done in the industrial validation. Finally, the validation of the hypothesis of this work is done, is the objective of this work.

8.1. Acceptance by Scientific Community

During the research and developments here presented, one scientific publication was submitted to a journal and seventeen scientific publications in proceedings, below is the list of papers:

1. Ferreira J., Agostinho C., Sarraipa J. and Jardim-Goncalves R., "Monitoring Morphisms to Support Sustainable Interoperability of Enterprise Systems", Accepted in: OnTheMove 6th International Workshop on Enterprise Integration, Interoperability, and Networking (EI2N 2011). Oct 21-23, Crete, Greece, 2011, (Ferreira et al., 2011).
2. Ferreira J., Agostinho C., Sarraipa J. and Jardim-Goncalves R., "Monitoring Morphisms to Support Sustainability of Interoperability in the Manufacturing Domain", Accepted in: 14th IFAC Symposium on Information COntrol Problems in Manufacturing (INCOM 2012). May 23-25, Bucharest, Romania, 2012, (Ferreira, Agostinho, et al., 2012).
3. Ferreira, J., Beca, M., Agostinho, C., and Jardim-Goncalves, R., “Monitoring Morphisms for Advanced Interoperability in Factories of the Future (FoF) Platforms”, Accepted in: 14th International Conference on Modern Information Technology in the Innovation Processes of Industrial Enterprises (MITIP 2012). January 24-26, Budapest, Hungary, 2012, (Ferreira, Beca, et al., 2012).
4. Ferreira, J., Beca, M., Nunez, M., and Jardim-Goncalves, R., “Management of Dynamic Furniture Manufacturing Networks”, Accepted in: Enterprise Interoperability: I-ESA’12 Proceedings. Wiley, pp. 209-217. DOI: 10.1002/9781118561942.ch31. August 20-23, Valencia, Spain, 2012.
5. Ferreira J, Agostinho C, Ilie-Zudor E, Jardim-Goncalves R., “Monitor for Information Alignment and Sustainability in Logistics Networks”. ASME International Mechanical Engineering Congress and Exposition, Volume 3: Design, Materials and Manufacturing, Parts A, B, and C. pp. 423-432. DOI: 10.1115/IMECE2012-89478. November 9-15, Houston, Texas, USA, 2012.
6. Ferreira, J., Agostinho, C., Sarraipa, J., and Jardim-Goncalves, R., “IMAGINE Blueprints for the Furniture Industry: Instantiation with the ISO 10303-236”.Accepted in: ICE'13 IEEE International Technology Management Conference. June 24-26, the Hague, Netherlands, 2013.

7. Ferreira, J., Beca, M., and Agostinho, C., Nunez, M., and Jardim-Gonçalves, R., “Standard Blueprints for Interoperability in Factories of the Future (FoF)”. Accepted in: 7th IFAC Conference on Manufacturing Modelling, Management, and Control. DOI: 10.3182/20130619-3-RU-3018.00427. June 19-21, St. Petersburg, Russia. 2013.
8. Ferreira, J., Gigante, F., Sarraipa, J., Nunez, M., and Agostinho, and Jardim-Gonçalves, R., “Collaborative production using dynamic manufacturing networks for SME's”. Accepted in: ICE'14 IEEE International Technology Management Conference. DOI: 10.1109/ICE.2014.6871620. June 23-25, Bergamo, Italy, 2014.
9. Raposo, C., Agostinho, C., Ferreira, J., and Jardim-Goncalves, R., “Automatic Detection of Harmonization Breaking in SOA-based Enterprise Networks”. Accepted in: 21st ISPE International Conference on Concurrent Engineering (CE2014). DOI: 10.3233/978-1-61499-440-4-726, pp. 726-735. Beijing, China, 2014.
10. Ferreira, J., Agostinho, C., and Jardim-Goncalves, R., “Dynamic Software Adapters as Enablers for Sustainable Interoperability Networks”. Accepted in: 5th International Conference on Information Society and Technology (ICIST 2015). March 8-11, Kopaonik, Serbia, 2015.
11. Ferreira, J., Agostinho, C., and Jardim-Goncalves, R., “Information Realignment in Pursuit of Self-Sustainable Interoperability at the Digital and Sensing Enterprise”. Accepted in: 15th IFAC/IEEE/IFIP/IFORS Symposium Information Control Problems in Manufacturing. DOI: 10.1016/j.ifacol.2015.06.055. May 11-13, Ottawa, Canada, 2015.
12. Ghimire, S., Melo, R., Ferreira, J., Agostinho, C., and Jardim-Goncalves, R., “Continuous Data Collection Framework for Manufacturing Industries”. Accepted in: On the Move to Meaningful Internet Systems: OTM 2015 Workshops. DOI: 10.1007/978-3-319-26138-6_5, pp. 29-40. October 26-30, Rhodes, Greece, 2015.
13. Marques-Lucena, C., Ferreira, J., Sesana, M., Fischer, K., and Agostinho C., “Process modelling approach for the liquid-sensing enterprise”. In I-ESA 2016 - Interoperability for Enterprise Systems and Applications. DOI: 10.1007/978-3-319-30957-6_17, pp. 211-223. March 31 – April 1. Guimarães, Portugal, 2016.
14. Ferreira, J., Sarraipa, J., Ferro-Beca, M., Agostinho, C., Costa, R., and Jardim-Goncalves, R., “End-to-end manufacturing in factories of the future”. Accepted in: International Journal of Computer Integrated Manufacturing, Volume 30, Issue 1: Factories of the Future: Challenges and Leading Innovations in Intelligent Manufacturing. 2017.
15. Agostinho, C., Ferreira, J., Pereira, J., Marques-Lucena, C., and Fischer, K., “Process Development for the Liquid-sensing Enterprise”. Accepted in: 5th International Conference on Model-Driven Engineering and Software Development (Modelsward 2017). DOI: 10.5220/0006331602390249. January 19-21, Porto, Portugal, 2017.
16. Ferreira, J., Soares, J., Jardim-Goncalves R., and Agostinho, C., “Management of IoT Devices in a Physical Network”. Accepted in: 21st International Conference on Control Systems and Computer Science (CSCS 2017). DOI: 10.1109/CSCS.2017.75. May 29-31, Bucharest, Romania, 2017.

17. Ferreira, J., Lopes, R., Chatzikokolakis, K., Zissis, D., Vidal, M., Mouzakitis, S., Agostinho, C., “Maritime Data Technology Landscape and Value Chain Exploiting Oceans of Data for Maritime Applications”. Accepted in: ICE/IEEE'17 - International Conference on Engineering, Technology and Innovation 2017. June 27-29, Funchal, Madeira, 2017.
18. Lopes, F., Ferreira, J., Jardim-Goncalves, R., and Agostinho, C., “Semantic Maps for IoT Network Reorganization in face of Sensor Malfunctioning”. Accepted in: IEEE SMC 2017 - IEEE International Conference on Systems, Man, and Cybernetics. October 5-8, Banff, Canada, 2017.

8.2. Industrial Validation

To validate the viability of the SSIF framework proposed during this work, it was necessary to implement it, having been divided by several validations through different projects, already described in Chapter 1.4. This dissertation was developed integrated into the Group for Research in Interoperability of Systems (GRIS) at UNINOVA and in the ENSEMBLE, CRESCENDO, IMAGINE, OSMOSE and C2NET Projects. During this chapter is presented the various works developed for the validation of this work.

8.2.1. Scenario for Enterprise Network Creation in frame of IMAGINE Project

In the furniture business, it is usual customers make customized requests that easily needs a DMN establishment, which a system like the one proposed is essential. One example of such customized requests is presented in the following to demonstrate how the proposed architecture executes it (this work follows the IMAGINE project described in Chapter 1.4.1).

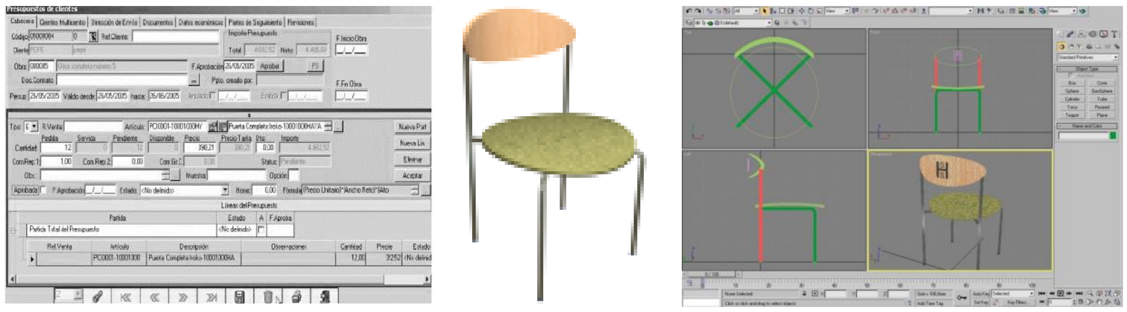
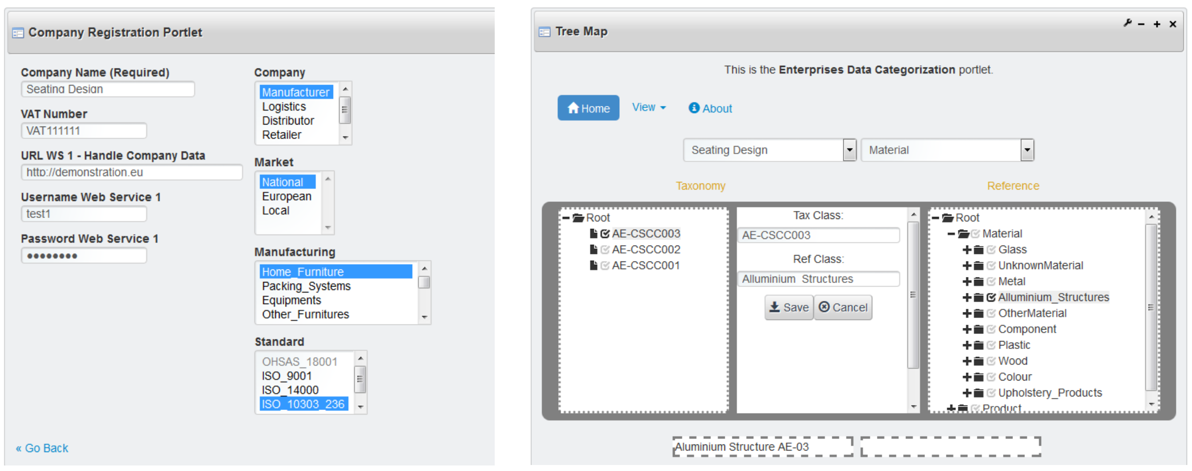


Figure 8-1: Representation of the Chair in their ERP (left), Customized Chair model 145 (middle) and the CAD design of the chair (right).

A furniture manufacturer receives an order to produce 1000 chairs for a specific hotel (Figure 8-1). This requested chair is a customized model defined from a particular chair model existing in the manufacturer catalogue, the Chair Model 145. The order has to fulfill the following requirements: a) the back of the chair should be in wood oak; b) the seat should be upholstered using green fabric; c) the structure should be made of aluminum; d) the back component should have the hotel logo in black glass. This order also contains a very tight delivery time, i.e. only three months since the acceptance of the request, and the cost of one single complete chair may not exceed one hundred euro.

This manufacturer is not able to produce all the parts of the chair, thus it needs to search for production partners to fulfill the order. The only part that this enterprise is able to produce is the wooden back. The other parts (the upholstering, the crystal logo, and the aluminum legs) should be requested to external production enterprises. For this reason, the furniture manufacturer subscribed his/her enterprise to the i-Platform manager, which in return received details and instructions about to establish or develop the required WS accordingly to the specific mentioned API able to establish the connection between the EIS and the i-Platform.

Afterwards, a second stage registration in the i-Platform is made in which the enterprise inserts the Company Name, Vat Number, URL and respective Username and Password, as illustrated in left part of Figure 8-2. The URL represents the address of the WS developed to transfer the data, which provides the overall required enterprise business information such as resources, contact details, etc. Once this information has been provided, the gateway then proceeds to transform it according to the platform's BP prior to storing the resulting data. In this way, an automatic insertion of data can be performed, avoiding further data interoperability problems.


Company Registration Portlet

Company Name (Required): Seating Design
VAT Number: VAT111111
URL WS 1 - Handle Company Data: http://demonstration.eu
Username Web Service 1: test1
Password Web Service 1: ●●●●●●●●●

Company: Manufacturer, Logistics, Distributor, Retailer
Market: National, European, Local
Manufacturing: Home_Furniture, Packing_Systems, Equipments, Other_Furnitures
Standard: OHSAS_18001, ISO_9001, ISO_14000, ISO_10303_236

« Go Back

Tree Map

This is the **Enterprises Data Categorization** portlet.

Home | View | About

Seating Design | Material

Taxonomy: Root, AE-CSCC003, AE-CSCC002, AE-CSCC001

Tax Class: AE-CSCC003
Ref Class: Alluminium_Structures
Save | Cancel

Reference: Root, Material, Glass, UnknownMaterial, Metal, Alluminium_Structures, OtherMaterial, Component, Plastic, Wood, Colour, Upholstery_Products, Product

Aluminium Structure AE-03

Figure 8-2: Company Registration Portal (left), Mapping Resources Portal (right).

The next phase is about the categorization of the different resources (as illustrated at the right side of Figure 8-2). Each enterprise has different concepts to categorize their resources. This is due to different characteristics, such as language, culture, etc. For this reason, a solution must be implemented to avoid enterprises of inserting their own concepts, which would create anarchy of concepts within the platform itself. Since for a simple concept could exist several meanings, if accepted any in the platform, it could create difficulties every time there is a need of searching for a specific concept or product. Thus, to avoid this situation, a reference ontology was created with several concepts, to be the referential semantic allowing each enterprise to select the ones that are suitable for its own purposes. The reference ontology integrates the structure of the BP, which represent resources, the categories to support the description of the enterprise, and the units and metrics. In Figure 4-4, an excerpt of the reference ontology is illustrated. Each enterprise has to categorize each resource, using the reference categories or concepts available in the reference ontology. In this step, the gateway creates mappings to relate each resource of each enterprise with their respective categories, after which they are stored in the repository (Integrated DB). With this method, it is possible to search for a partner that is able to provide a specific resource related to the manufacturing of a product. The existence of semantic referential representation facilitates the standardization of categories or concepts used in the i-Platform avoiding its repetition and consequently facilitating the searching process.

An example of a semantic alignment process using the developed user interface is illustrated in Figure 8-2 (right figure). It presents the "AE-CSCC003" product, which is a specific aluminum structure of the "Seating Design" enterprise. Thus, in order to be harmonized with the existent semantic referential, such product was mapped to the reference category "aluminum structures". Consequently, if someone would like to search for an aluminum structure supplier will find "Seating Design". Through this, many products with particular names or just IDs will be harmonized, in such way to be searchable. Since each enterprise can keep unchanged their nomenclatures internally, while having the concepts harmonized in the proposed platform, it will facilitate smooth business collaborations that avoid semantic interoperability problems. After the correct insertion of data, Phase 0 ends and the can start the 6-cycle SSIF framework.

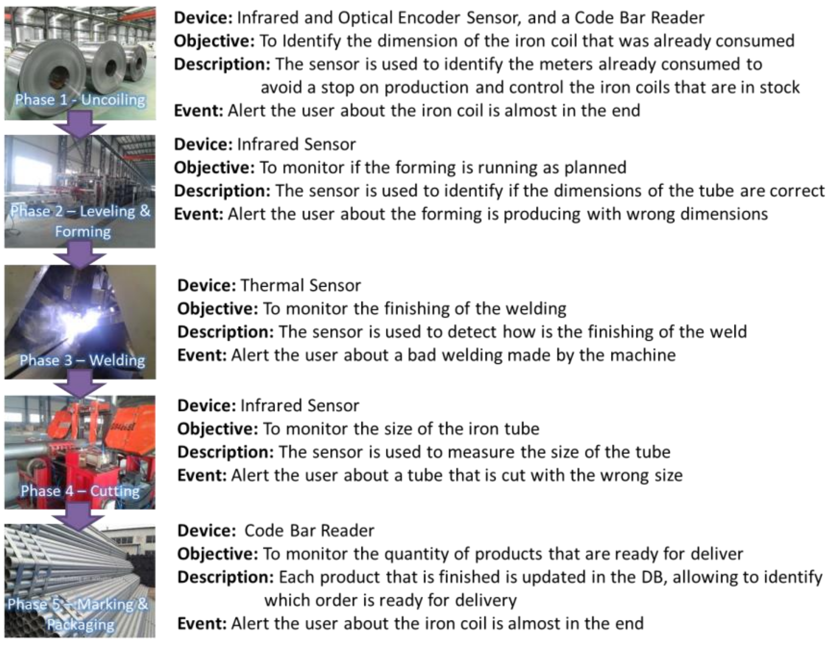
In resume, the presented scenario in the furniture industry demonstrated the proposed framework implementation feasibility in the FoF, the methodology and architecture were described in Chapter 1.4.1. It shows how possible is for any manufacturing enterprise to interoperate within a manufacturing network management system such as the platform. Consequently, it addresses the many concerns related to the interoperability, such as common data structures, semantics and methods for data extraction and input. Additionally, due to its generic principles, which embody the overall and present research challenges, the framework can be adapted to other industrial scenarios or application domains. With all these advantages, the platform provides to the enterprises a faster, simplified and more efficient way to create a DMN, reducing time to market and improving the quality of their products. This scenario is a validation of the architecture presented in Chapter 4.2, it validates the Dynamic Network Manager Module presented in the Self-Sustainable Interoperability Framework of Figure 4-1, by demonstrating how an enterprise can do the registration of their information and resources, representing the Phase 0. This work was validated through the IMAGINE project described in Chapter 1.4.1 and with the publications 2, 3, 4, 5, 6, 7, 8, 10, 11 and 14 represented in Chapter 8.1.

8.2.2. Scenario Resource Management in Metalworking in frame of C2NET Project

This scenario is related to a metalworking enterprise, which produces iron tubes of the metalworking SME’s case of C2NET Project, as described in Chapter 1.4.2. The enterprise produces three types of tubes (square, round and flat), that can have different measurements, for example, the round tube can have 5, 10 or more centimeters of radius (as illustrated in Figure 8-3). Production is done according to demand, meaning that they produce depending on the orders they receive, implying that there is no stock of material or product. This situation forces the enterprise to have a more rigorous control of their stock since it needs to buy more raw materials every time an order is received. Due to these facts, the enterprise wants to improve their production by having more control over the production, to know in real time the consumption of the raw material, allowing the order of new raw materials to be done on time, avoiding delays in the production.

Figure 8-3: Iron Tube Types.

One way to improve the production is through the use of sensors to monitor the production since it enables the identification of the raw material used in the moment of production, the tube that is being produced, verify the quantity of the order and check if the status of the production is ready for delivery. In Figure 8-4 the production of an iron tube is illustrated, divided into five phases.



Device: Infrared and Optical Encoder Sensor, and a Code Bar Reader
Objective: To Identify the dimension of the iron coil that was already consumed
Description: The sensor is used to identify the meters already consumed to avoid a stop on production and control the iron coils that are in stock
Event: Alert the user about the iron coil is almost in the end

Device: Infrared Sensor
Objective: To monitor if the forming is running as planned
Description: The sensor is used to identify if the dimensions of the tube are correct
Event: Alert the user about the forming is producing with wrong dimensions

Device: Thermal Sensor
Objective: To monitor the finishing of the welding
Description: The sensor is used to detect how is the finishing of the weld
Event: Alert the user about a bad welding made by the machine

Device: Infrared Sensor
Objective: To monitor the size of the iron tube
Description: The sensor is used to measure the size of the tube
Event: Alert the user about a tube that is cut with the wrong size

Device: Code Bar Reader
Objective: To monitor the quantity of products that are ready for deliver
Description: Each product that is finished is updated in the DB, allowing to identify which order is ready for delivery
Event: Alert the user about the iron coil is almost in the end

Figure 8-4: Production Phases Description.

The first phase is the uncoiling of the iron coil, at this point the production starts with the preparation of the iron, enabling the ability to control the raw material being used and the standpoint of the iron coil, with the support of sensors (these phases are represented in Figure 8-5). For the sensors to give an advantage in this phase, it was decided to use three different sensors, one Code Bar Reader to read the code attached to the iron coil to identify the coil in use. The Infra-red Sensor is used to identify the beginning and end of the iron coil, measuring the distance between the sensor and the coil. Every time the coil is missing, the sensor is going to measure the distance to the floor, identifying that the coil is missing. The last one is the Optical Encoder Sensor, which is attached to the belt of the production line, it measures the length consumed by the machine, allowing the system middleware to know (with the support of the Infra-red sensor) the length that is missing of the coil. With these three sensors working together, it is possible to have events to warn the user about the stock of the raw material (since sensors identify the coil that is being consumed). This situation occurs, every time the production stop or not (due to the sensor that is identifying if coil exists in the production line), or about the missing length of the coil to complete the ordered quantity (to avoid a stop in the production to add a new coil).

The second phase is the leveling and forming of the iron, which is going to transform the iron plate into a tube. This phase passes through several stages until it reaches the final stage. Depending on the final form of the tube, the sensor is able to measure the dimensions of the tube and verify if the tube has the required dimensions. The system middleware needs to know which order is being executed to verify that all the settings are according to the production requirements. For example, if it's producing around the iron tube with 5 centimeters of radius, the sensor is going to measure the radius to verify if it's correct or not. In the case of error, the user can be warned through an event.

The next phase is to do the welding of the iron tube, using the Infra-red sensor to control the weld. The sensor follows the weld to detect if mistakes occur, warning the user through an event. The fourth phase relates to the production of the final product, which is to cut the tube with the respective length, usually, these type of tubes have 6 meters in length. In order to identify if the correct tube length is correct or not, an Infra-red sensor is used, in case the measure is incorrect an event is triggered to warn the user. This event brings a big benefit since it also allows to identify if the machine is not well calibrated avoiding the waste of more raw materials and time.

The final phase of this process is the packaging of the iron tubes, in this phase, the machine packages the iron tubes, ready to be shipped for delivery. The use of a Code Bar Reader in this phase benefits the tracking of the orders, therefore identifying if an order is fulfilled and ready for distribution. Every time that an order is not fulfilled or delayed, an event is triggered, warning the user. Since this occurred in real-time, it is possible to take measures to avoid more delays.

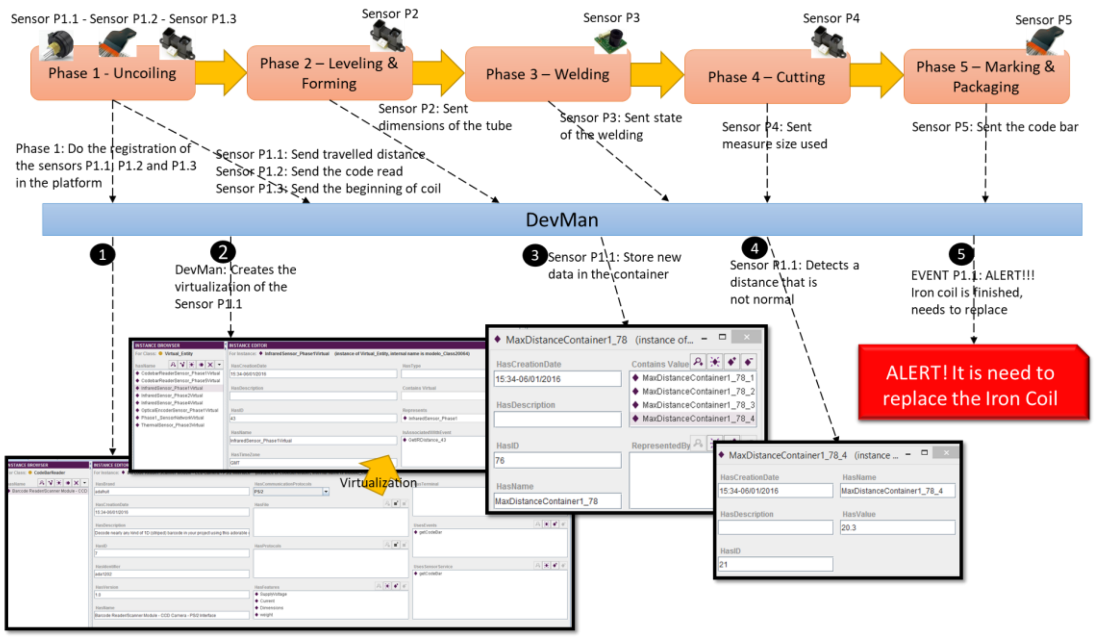

Sensor P1.1 - Sensor P1.2 - Sensor P1.3
Sensor P2
Sensor P3
Sensor P4
Sensor P5
Phase 1 - Uncoiling
Phase 2 – Leveling & Forming
Phase 3 – Welding
Phase 4 – Cutting
Phase 5 – Marking & Packaging
Sensor P2: Sent dimensions of the tube
Sensor P3: Sent state of the welding
Sensor P4: Sent measure size used
Sensor P5: Sent the code bar
Phase 1: Do the registration of the sensors P1.1, P1.2 and P1.3 in the platform
Sensor P1.1: Send travelled distance
Sensor P1.2: Send the code read
Sensor P1.3: Send the beginning of coil
DevMan
1
2
DevMan: Creates the virtualization of the Sensor P1.1
3
Sensor P1.1: Store new data in the container
4
Sensor P1.1: Detects a distance that is not normal
5
EVENT P1.1: ALERT!!! Iron coil is finished, needs to replace
ALERT! It is need to replace the Iron Coil
Virtualization
MaxDistanceContainer1_78 (instance of...
HasCreationDate
15:34-06/01/2016
Contains Value
MaxDistanceContainer1_78_1
MaxDistanceContainer1_78_2
MaxDistanceContainer1_78_3
MaxDistanceContainer1_78_4
HasDescription
HasID
76
RepresentedBy
HasName
MaxDistanceContainer1_78
MaxDistanceContainer1_78_4 (instance ...
HasCreationDate
15:34-06/01/2016
HasName
MaxDistanceContainer1_78_4
HasDescription
HasValue
20.3
HasID
21

Figure 8-5: Process for Adding Devices.

In Figure 8-5 an example of how the systems work is illustrated. First (Black Circle 1) the sensors network is created in the cloud platform. This is made by creating the physical entity representation of each sensor (assuming the sensors already exist in the KB, the platform cloud is going to purpose the sensor to avoid the user to add it manually). Then, it is added to the sensor their characteristics (as type of sensor, name, reference, protocols, etc.), the conceptual events and services that the platform cloud provides to this type of sensor (for example a temperature sensor by default, has a service to get the temperature and another to store values in the DB, and an event to detect a maximum or minimum temperature and warn the user). After the insertion of the physical entity, the framework detects a new entry and virtualizes the sensor (Black Circle 2). The virtualization is made by creating the virtual entity and associates the specific events and services as Chapter 4.3.2 (based on the conceptual events and services provided by the KB).

After these steps, a sensor is registered in the cloud platform, being able to provide the sensed data to the platform cloud, storing it to be later used for events or, for example, in a simulation scenario. This ends the installation phase and starts the runtime phase. In this phase, the monitor starts with the support of events to detect the performance indicators defined by the user and the consequential actions (warnings, alerts, etc.). This represents the black circles 3 and 4, wherein 3 a service is called to get the values of the sensor, this happens every time a new value is read. Then in 4, an event is triggered due to the measured length, read by the sensor, indicates that the coil is missing, or in this case the coil finished. After which an event is triggered, warning the user, to change the iron coil. Only the process of one sensor is described with a practical example. All the steps are equal to the rest of the sensors, the only difference is which services and events are provided by the platform cloud depending on the type of the sensor.

In resume, the framework was able to virtualize the devices and associated the necessary services and events, as well as, monitor sensed data to trigger actions, according to the dynamic CEP rule creation. However, further development is required, to increase reliability and CEP rule creation dynamics. Also, the matching device process can be improved in order to accommodate more branches and nodes in the tree-like lookup, to decrease human user intervention. Therefore, it can be realized, that the development the DEVMAN framework is on the right track for achieving the objectives of providing real-time data collection and intelligent device management. The knowledge enriched network, devices can be related and interconnected to each other ensuring interoperability in IoT. This enables manufacturing enterprises to focus on their core competencies, taking the most of IoT devices and increase efficiency in business and manufacturing processes. This scenario is a validation of the architecture presented in Chapter 4.2, it represents the register of the physical devices that demonstrate Phase 0, Phase 1 and Phase 2. Phase 3 to identify the services and events that each device needs to use and Phase 6 of the device monitoring. This work was validated through the C2NET project described in Chapter 1.4.2 and with the publications 12 and 16 represented in Chapter 8.1.

8.2.3. Scenario for Detection of a Device Malfunction in Frame of C2NET Project

This scenario represents two different sensor demonstrations divided between two rooms, implemented for the validation of the C2NET project (Chapter 1.4.2) in a factory of an enterprise called

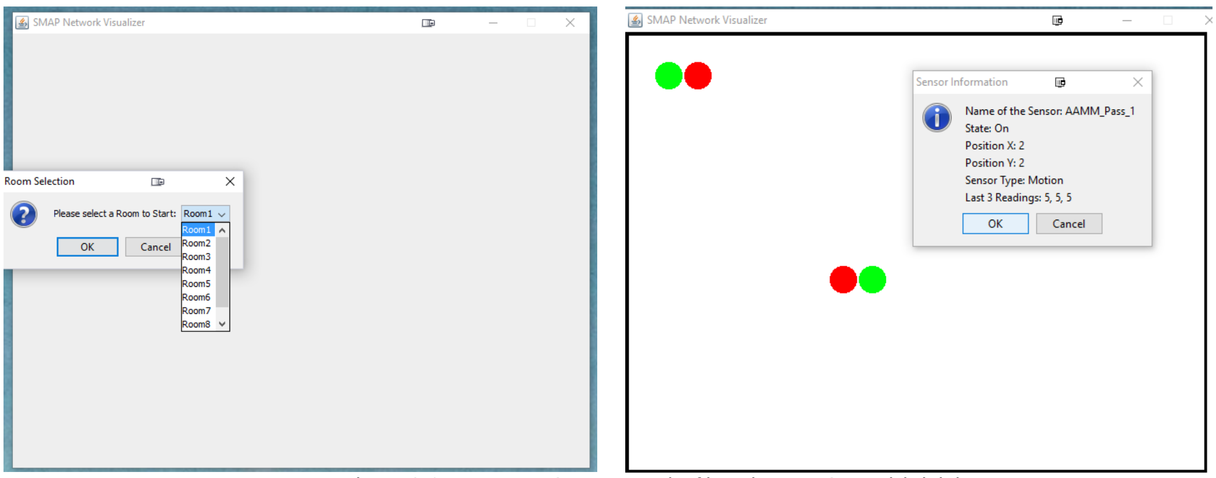
"*António Abreu Metalomecânica, Lda*" located in Vila Nova de Famalicão, in the north of Portugal".

Room 1: SMAP Initialization and Use of Semantic Maps

As can be seen in Figure 5-3, this room has four motion sensors:

- **AAMM_Pass_1** - Ultrasonic sensor, active sensor on the top left corner of the SMAP Network Visualizer;
- **AAMM_Pass_2 -** Passive Infrared sensor, inactive sensor on the top left corner of the SMAP Network Visualizer, redundant to sensor AAMM_Pass_1;
- **AAMM_Pass_3** - Ultrasonic sensor, active sensor on the bottom right corner of the SMAP Network Visualizer;
- **AAMM_Pass_4 -** Infrared sensor, the inactive sensor on the bottom right corner of the SMAP Network Visualizer; redundant to sensor AAMM_Pass_3.

These sensors represent the four motion sensors implemented in a station, of the mentioned factory, responsible for detecting the passage of manufactured pieces. The sensors are placed in pairs, in the top left corner and the bottom right corner (as appears in the SMAP Network Visualizer of the following figures), and, initially, only one sensor of each pair is active, the other one is only for redundancy purposes. This redundancy is achieved with the use of semantic maps, represented by a blue line connecting the involved sensors, as can be seen in the left image of Figure 8-6.

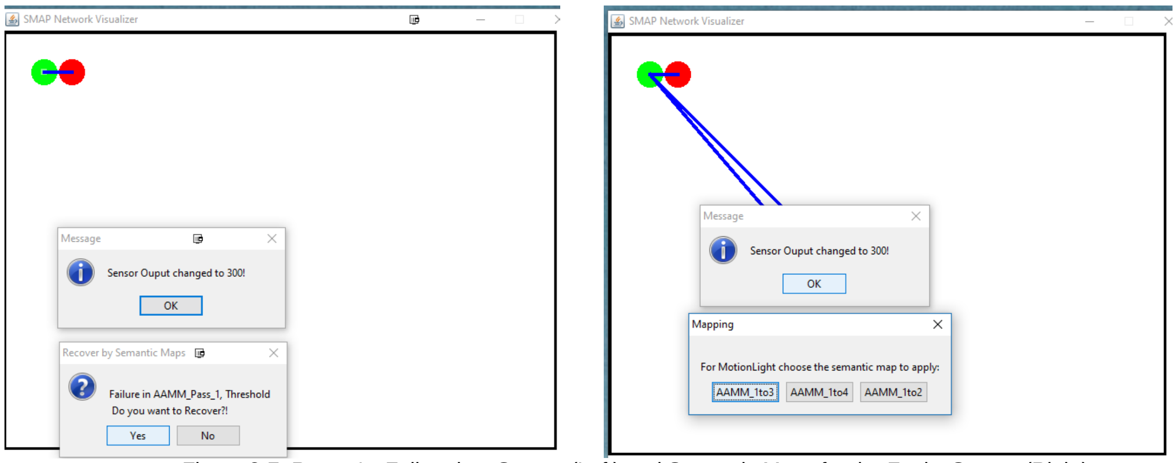


Please select a Room to Start: Room1

Name of the Sensor: AAMM_Pass_1
State: On
Position X: 2
Position Y: 2
Sensor Type: Motion
Last 3 Readings: 5, 5, 5

Figure 8-6: Room 1 – Start Room (Left) and Room Started (Right).

In the left image of Figure 8-6, shows the graphical user interface room selection process. The dropdown list of rooms' results from a query to the C2NET ontology to get all the existent rooms in the database file created for this implementation. Following that, in the right image of the Figure 8-6 shows *Room 1* functioning with the Log showing the measurements in real time from the two active sensors (only two are green), the SMAP Network Visualizer showing the existent sensors in the room (four motion sensors, two *On* and two *Off*) and by clicking on the green circle, on the top left corner, we get access to the sensor information from sensor *AAMM_Pass_1*, displayed in an additional box as seen in the mentioned figure.



Sensor Ouput changed to 300!

Failure in AAMM_Pass_1, Threshold
Do you want to Recover?!

Sensor Ouput changed to 300!

For MotionLight choose the semantic map to apply:

Figure 8-7: Room 1 - Failure in a Sensor (Left) and Semantic Maps for the Faulty Sensor (Right).

Next, in the left image of Figure 8-7, the button to show maps from *AAMM_Pass_1* was clicked and the interface showed redundancy to the nearby sensor, *AAMM_Pass_2*, with a blue line connecting the centers of the respective circles. To increase the complexity of this testing procedure, the “New Map” function was used to create two new semantic maps from *AAMM_Pass_1* to *AAMM_Pass_3* and *AAMM_Pass_4*, individually (this can be seen in the right image of Figure 8-7 and Figure 8-8, where two new blue lines appear, connecting the *AAMM_Pass_1* to the mentioned destination sensors). After that, the “Sensor Readings” button was used to change the sensor output of *AAMM_Pass_1* to 300 units. The fault detection methods were active, so, the program notices the failure in the sensor, by the Threshold fault detection method, and asks for recovery using semantic maps.

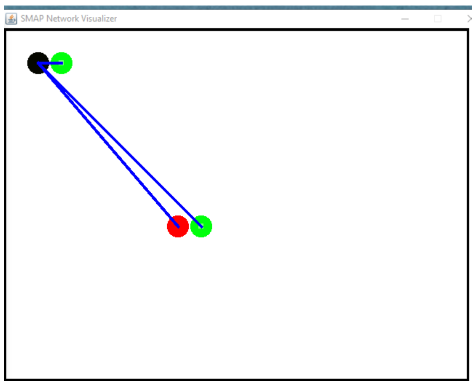


Figure 8-8: Room 1 - Result of the use of a Semantic Map.

In Figure 8-8, we can see that the user chose the semantic map for the *AAMM_Pass_2* (visible in the Log box, as active), and the origin sensor circle turned black to indicate the state of *Error*, this disables further semantic maps to target it as destination sensor. The destination sensor turned green and, in the Log, we can see its real-time measurements (the measurements from *AAMM_Pass_1* stopped because it is no longer active). This semantic map selection process is active for this test, but it can be automated using the previously mentioned weight component, present in every semantic map.

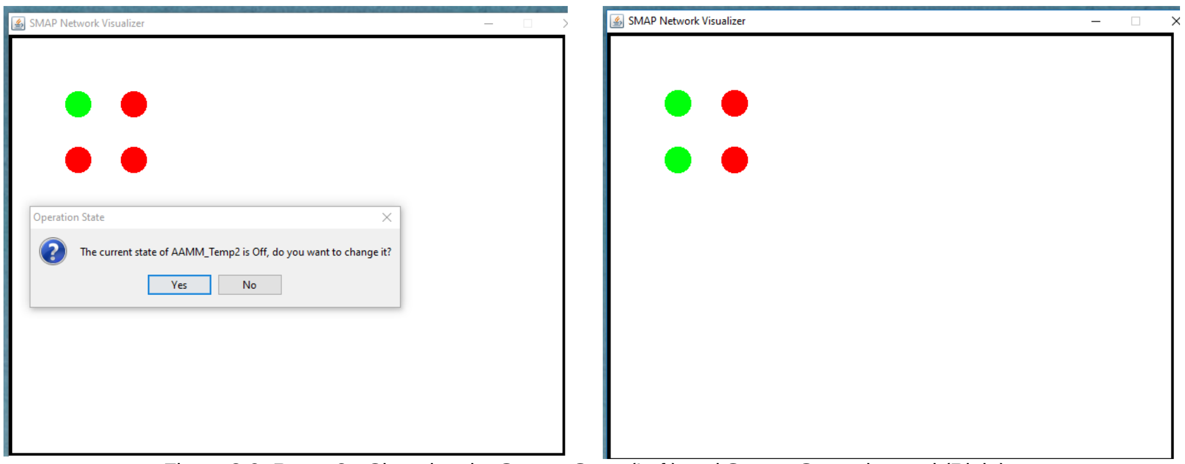
<u>Room 2: Sensor State Changes and Network Recovery</u>

As can be seen in the left image of Figure 8-9, this room has four temperature sensors:

- **AAMM_Temp1** - Thermistor sensor, active sensor on the top left corner of the SMAP Network Visualizer;
- **AAMM_ Temp2** - Resistance Thermometer sensor, inactive sensor on the bottom left corner of the SMAP Network Visualizer;
- **AAMM_ Temp3** - Thermistor sensor, inactive sensor on the top right corner of the SMAP Network Visualizer;
- **AAMM_ Temp4** - Resistance Thermometer sensor, the inactive sensor on the bottom right corner of the SMAP Network Visualizer.

This room represents the second scenario from the mentioned factory. It is a station with an industrial oven where the manufactured pieces dry after being painted. In strategical locations of the oven, four temperature sensors are used for calculating the average temperature of the room (this calculation is done by the CEP). The objective is to have enough active sensors to do this calculation properly and, if possible, leave others as non-active to allow redundancy. Initially, only one sensor, *AAMM_Temp1* is *On* (as can be seen in the left image of Figure 8-9).

In both images of Figure 8-9, a basic functionality, essential from the demonstrative and experimental point of view, is shown. That functionality is just the activation of a certain sensor, in this case, *AAMM_Temp2*, to better manipulate the simulation environment and force the desired following testing procedure.

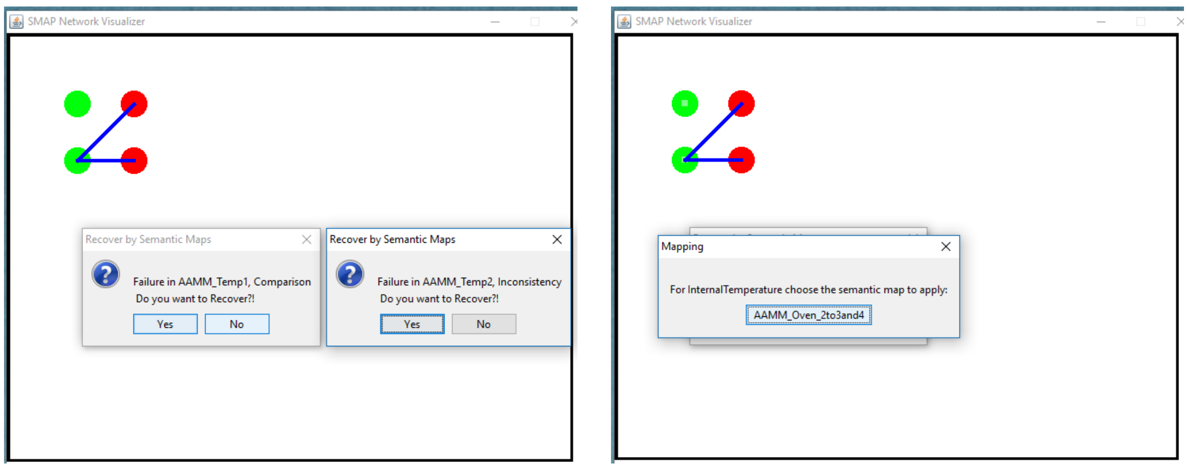


The current state of AAMM_Temp2 is Off, do you want to change it?

Figure 8-9: Room 2 - Changing the Sensor State (Left) and Sensor State changed (Right).

In the left image of Figure 8-9, shows the screen after clicking the "Sensor State" button and choosing the desired sensor, *AAMM_Temp2*. The sensor current state is presented (which also can be seen in the SMAP Network Visualizer) and in this case, it's *Off*. After that, the interface shows a message dialog which requests the new state for the selected sensor and, in this test, the *On* state is selected. And finally, right image of Figure 8-9 shows *AAMM_Temp2* as active and, in the Log, we can see the current sensor measurements.

In the left image of Figure 8-10 depicts the procedure and result of changing the reading value of the sensor, in this case, the *AAMM_Temp2*. It is visible in the SMAP Network Visualizer, that the three sensors share redundancy by a semantic map (depicted by the blue lines connecting the center of the circles). The new sensor output value is within the threshold of the sensor, contrarily to the previous example of fault detection. But the significant change from 5 units to 25 units triggers the Inconsistency method.


SMAP Network Visualizer

Recover by Semantic Maps

Failure in AAMM_Temp1, Comparison
Do you want to Recover?!

Recover by Semantic Maps

Failure in AAMM_Temp2, Inconsistency
Do you want to Recover?!

SMAP Network Visualizer

Mapping

For InternalTemperature choose the semantic map to apply:

AAMM_Oven_2to3and4

Figure 8-10: Room 2 - Inconsistency and Comparison Failures Detected (Left) and Use of a Semantic Map with two Destination Sensors (Right).

As seen in the message dialogs from the left image of Figure 8-10, the Inconsistency method was activated, but the Comparison fault detection mechanism was activated too. This occurred because the Comparison mechanism compares the output values of sensors related by geographical interest or proximity distance, using certain proximity thresholds. The developed program can compare in numerous nearby sensors (depending on the processing capability of the hardware running this interface). In this case, the four sensors can be considered due to their proximity but only the active sensors are compared for logic reasons. And with the change made previously, the difference between the two active sensors is 20 units, above the, stipulated for this room (10 units), and for that reason, the Comparison method was activated. To proceed with the use of the semantic map mentioned before, the user clicks “No” on the Comparison method and “Yes” on the message dialog presenting the activation of the Inconsistency fault detection rule.

In the left image of Figure 8-10, the recovery by semantic maps is activated by the Inconsistency fault detection method that compares the current reading of a certain sensor with its previous ones, considering a limited acceptable variation. In this case, the current sensor reading is 25 and all the previous ones are 5 units. The limit variation stipulated for this simulation is under 20 units, so the recovery process is automatically initiated.

As seen in the right image of Figure 8-10, there is only one semantic map available for this situation, *AAMM_Oven2to3and4,* and the destination sensors are *AAMM_Temp3* and *AAMM_Temp4*. At this point, the readings from *AAMM_Temp2* are being disregarded because the simulator has undergone the semantic map redundancy procedure after the user has chosen to do so.

Finally, in Figure 8-11, we can see the resulting network state, where the *AAMM_Temp2* state is now *Off* and the destination sensors are now *On*. In the Log box, the output values of the three active sensors are now being displayed.

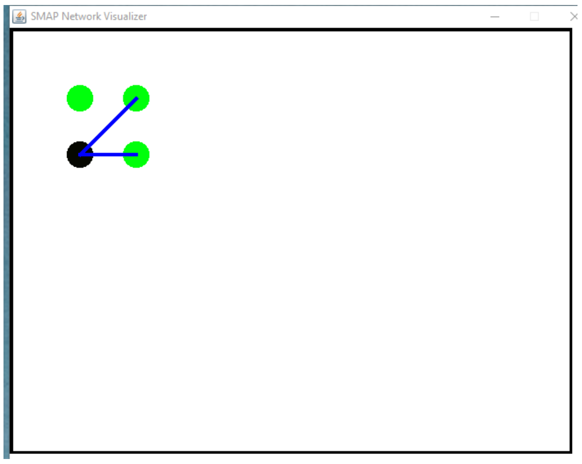


Figure 8-11: Room 2 - Resulting Network State.

In resume, the framework was able to virtualize the devices, as well as, monitor sensed data to trigger actions, according to the dynamic CEP rule creation. With this feature, it can identify this happening so that the system can act as soon as possible and avoid creating a problem in the network of devices. At the same time, it has the ability to simulate a device network, so it can see how the network will react in a situation where a device fails. Finally, through the mappings it is able to propose a solution to solve a problem of a device that is malfunctioning, proposing another device that is able to do the same task. This scenario is a validation of the architecture presented in Chapter 4.2, it represents the register of the physical devices that occur in Phase 0. Phase 3 identifies the services and events that each device needs to use, and it is also possible to simulate devices, finally, Phase 6 monitoring the devices. This work was validated through the C2NET project described in Chapter 1.4.2 and publication 18 represented in Chapter 8.1.

8.2.4. Scenario for the Monitoring of Morphisms to Support Sustainable Interoperability in Frame of CRESCENDO Project

The proposal to help maintain the interoperability status of business networks with the help of MAS relies on a framework that monitors mappings in pursuit of changes in the CM. To demonstrate the reliability of this framework, an application scenario based on CRESCENDO project (the description of this project is in Chapter 1.4.5) is presented, a scenario that simulates a collaborative network that is cooperating to manufacture a turbine engine for an airplane was created.

When a collaborative network is formed in the beginning it is decided who are the involved partners and the tasks that each one will fulfill, in this case, the objective is to manufacture a turbine engine and to produce three parts are needed, the nacelle, the engine, and the pylon. So it was decided to have four partners:

- *Supplier B* where the Nacelle is made;
- *Supplier C* where the Pylon is made;
- *Supplier D* where the Engine is made;
- *Manufacturer A* where all the three parts are assembled to produce the Turbine.

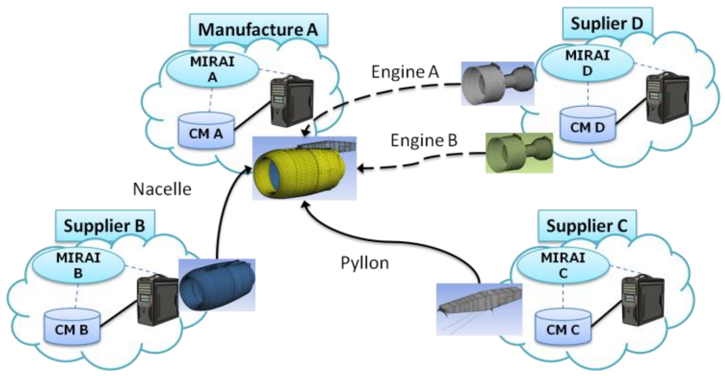


Figure 8-12: MIRAI Network for the Crescendo Scenario.

In a big production like this case coordination and discipline between the parts involved in the collaborative network are required, because a mistake can be catastrophic to the production and create a big loss in time and consequently, a loss in money. So, teams from different disciplines from design to aerodynamics, simulation, etc. need to cooperate, describing and modeling the several stages of the development life-cycle of the product.

In this scenario, the four enterprises work with different EIS and modeling languages for structural models. After storing the mappings in the corresponding CM's following the MapT description of Equation 4-1, the manufacture end ups with three high level morphisms and the rest of the suppliers have one high level morphisms (CMa: A→B, A→C and A→D; CMb: B→A; CMc: C→A; CMd: D→A) and the corresponding sub-morphisms to each of the relating entities and attributes.

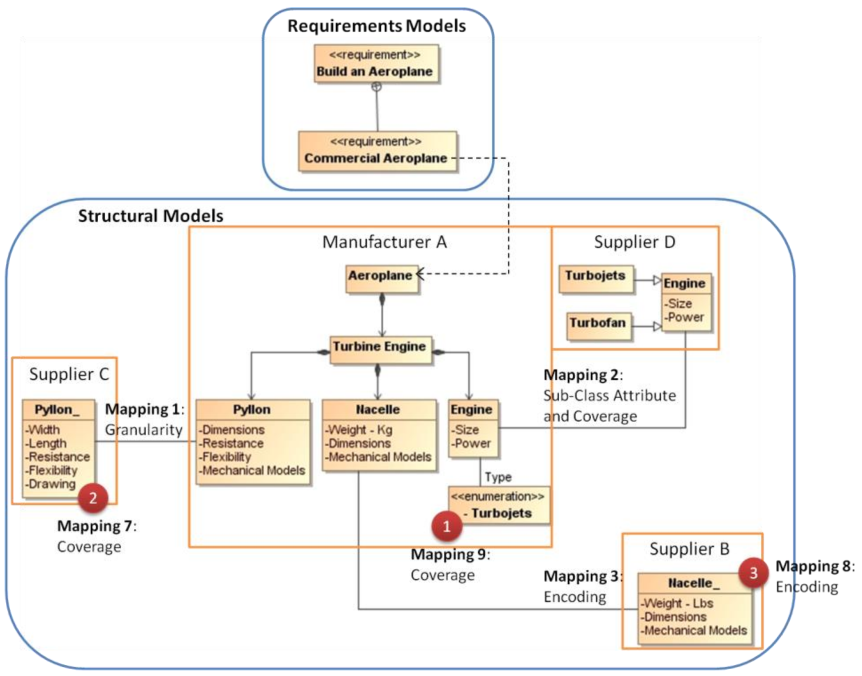

Requirements Models
<<requirement>> Build an Aeroplane
<<requirement>> Commercial Aeroplane
Structural Models
Manufacturer A
Aeroplane
Turbine Engine
Pyllon -Dimensions -Resistance -Flexibility -Mechanical Models
Nacelle -Weight - Kg -Dimensions -Mechanical Models
Engine -Size -Power
Type
<<enumeration>> - Turbojets
Supplier D
Turbojets
Turbofan
Engine -Size -Power
Supplier C
Pyllon_ -Width -Length -Resistance -Flexibility -Drawing
Mapping 1: Granularity
Mapping 2: Sub-Class Attribute and Coverage
Mapping 7: Coverage
Mapping 9: Coverage
Mapping 3: Encoding
Mapping 8: Encoding
Supplier B
Nacelle_ -Weight - Lbs -Dimensions -Mechanical Models
1
2
3

Figure 8-13: Representation of the mappings between the models without the evolution of the system.

This scenario demonstrates how the MIRAI reacts to changes in the requisites in a turbine-like it is shown in Figure 8-12, in the figure where the collaborative network is represented by the four enterprises and how they are collaborating to accomplish the final product. They are following the agreement made when the creation of the EC, and have the MIRAI integrated into the system.

In this scenario, a new requisite to the airplane was made, the buyer wants the airplane to be more economical and eco-friendly, so to achieve this, the manufacturer was forced to change the engine that was a Turbojet to a Turbofan engine. The modification was made because the Turbofan is a more economical, efficient and makes less noise than the other, thus it is in the request parameters. In Figure 8-12, a change of the Engine A to Engine B was made, and this forced an evolution in the model A to A', which after detection by Agent Monitor Mediator triggers a warning about the change, this has to be handled swiftly with the risk of jeopardizing the full network.

In Figure 8-13, the models to produce an airplane are presented, this is following the MBSE paradigm. The 1st one is the Requirement Model where the requirements of the user are described, and it is possible to see that the user wants to construct a commercial airplane. The other model is the Structural Model where the turbine engine that is to be used in the airplane is described, then a connection between the two models is needed, these connections are to describe the airplane following the requirements of the user, these connections are represented in the figure with a dashed line.

The Figure 8-14 represents a new evolution resulting of a new requirement of the user, this time the user is asking for an airplane more economic and friendly for the environment, to accomplish this, a modification in the airplane is required. To create an economic and eco-friendly airplane the turbojet engine was switched to a turbofan engine like it was shown in the figure (dashed lines).

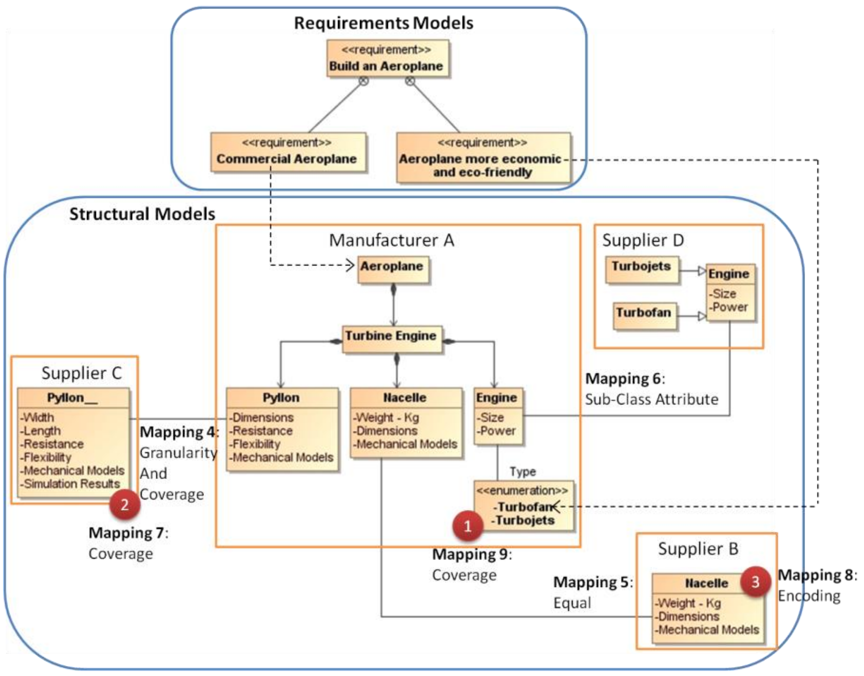
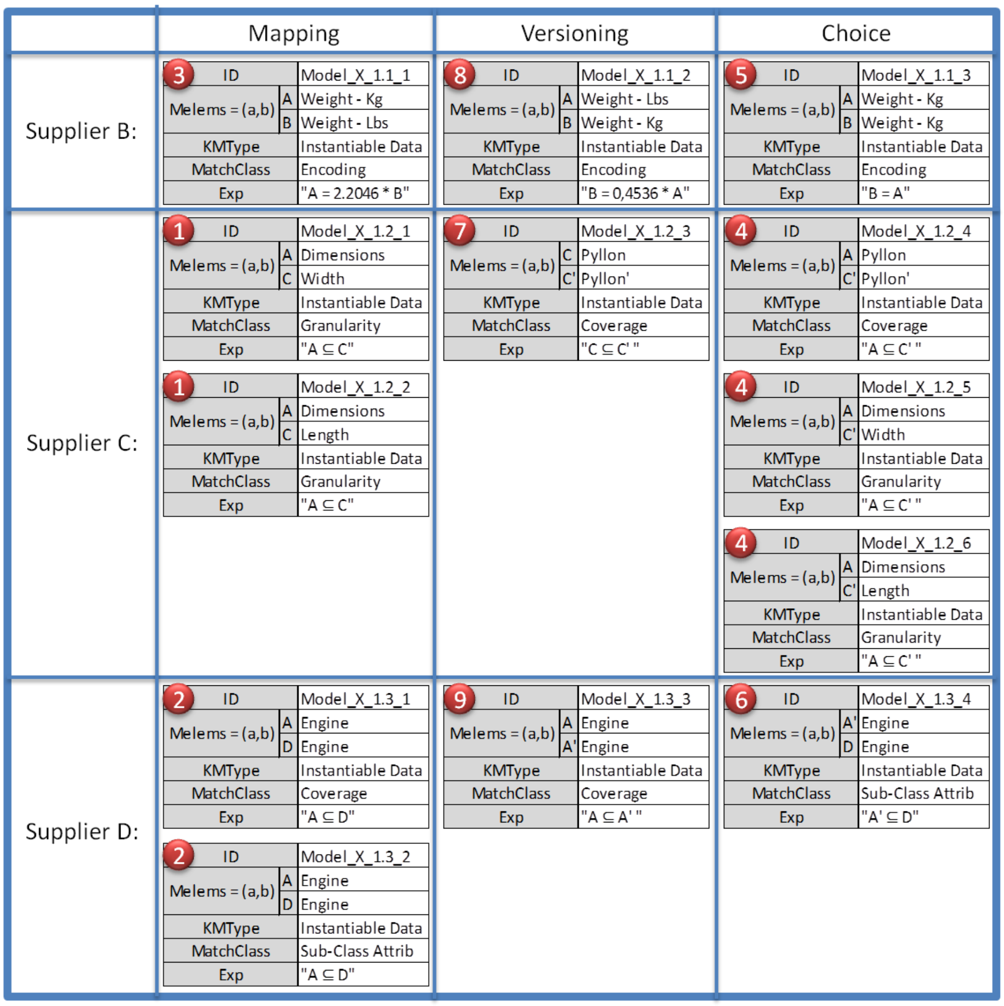


Figure 8-14: Representation of the mappings between the models with the evolution of the system.

Until now the models to construct the airplane were explained, to answer the request of the client, the model was modified. Although all the process follows the MBSE paradigm this does not mean that the different enterprises use the same model languages, for example, the Manufacturer A uses Model Language X and Supplier B use Model Language Y, and this will create some interoperability issues. So each enterprise has the mappings relating the models represented in the CM, for example, the Manufacturer A, in this case, has four mappings, A→A', A→B, A→C, and A→D, that are divided in different mappings between the two figures, the mappings 1, 2 and 3 is in Figure 8-12, the mappings 4, 5 and 6 is represented in Figure 8-13, the mappings 7, 8 and 9 are in both figures and are represented with red circles because they are connected between the two structural models, in Figure 8-15 are represented the different mappings used in this scenario. Below the first 6 mappings are explained, the red circles are explained in the test cases since that is the result of the MIRAI responding to the changes:



			ID	Melems = (a,b)		KMType	MatchClass	Exp
Supplier B:	Mapping	3	Model_X_1.1_1	A Weight - Kg	B Weight - Lbs	Instantiable Data	Encoding	"A = 2.2046 * B"
Supplier B:	Versioning	8	Model_X_1.1_2	A Weight - Lbs	B Weight - Kg	Instantiable Data	Encoding	"B = 0,4536 * A"
Supplier B:	Choice	5	Model_X_1.1_3	A Weight - Kg	B Weight - Kg	Instantiable Data	Encoding	"B = A"
Supplier C:	Mapping	1	Model_X_1.2_1	A Dimensions	C Width	Instantiable Data	Granularity	"A ⊆ C"
Supplier C:	Mapping	1	Model_X_1.2_2	A Dimensions	C Length	Instantiable Data	Granularity	"A ⊆ C"
Supplier C:	Versioning	7	Model_X_1.2_3	C Pyllon	C' Pyllon'	Instantiable Data	Coverage	"C ⊆ C' "
Supplier C:	Choice	4	Model_X_1.2_4	A Pyllon	C' Pyllon'	Instantiable Data	Coverage	"A ⊆ C' "
Supplier C:	Choice	4	Model_X_1.2_5	A Dimensions	C' Width	Instantiable Data	Granularity	"A ⊆ C' "
Supplier C:	Choice	4	Model_X_1.2_6	A Dimensions	C' Length	Instantiable Data	Granularity	"A ⊆ C' "
Supplier D:	Mapping	2	Model_X_1.3_1	A Engine	D Engine	Instantiable Data	Coverage	"A ⊆ D"
Supplier D:	Mapping	2	Model_X_1.3_2	A Engine	D Engine	Instantiable Data	Sub-Class Attrib	"A ⊆ D"
Supplier D:	Versioning	9	Model_X_1.3_3	A Engine	A' Engine	Instantiable Data	Coverage	"A ⊆ A' "
Supplier D:	Choice	6	Model_X_1.3_4	A' Engine	D Engine	Instantiable Data	Sub-Class Attrib	"A' ⊆ D"

Figure 8-15: Mappings of the scenario.

- **Mapping 1** – represents a part of the initial morphism A→C, which is represented by one mismatch, which is the type of Granularity. This is due to the same information (Dimensions) from A is decomposed in B (Width and Length);
- **Mapping 2** – represents a part of the initial morphism A→D, and two mismatches are represented: 1) One (Sub-Class Attribute) is due to the enumeration attribute "type" of engine from A being represented in D by a subclass hierarchy; 2) The other (Coverage) refers to the same attribute and is about the A don't know what is a Turbofan;

- **Mapping 3** – represents a part of the initial morphism A→B, and a morphism that is represented, that in this case is Encoding. This is due to in A is using Kg as a format and in B is using Lbs, causing a difference in the formats between the A and B;
- **Mapping 4** – this map is a consequence of the evolution of the Mapping 1, in this case, the Granularity continue to exist, but another mismatch appear, that is a Coverage consequence of a change in the model of C (Simulation Results) that does not exist in A, this brings loss of information;
- **Mapping 5** – is a consequence of the evolution of the Mapping 3, in this one a change in B was made, but this change made the model of B be equal to A, so this mismatch is Equal;
- **Mapping 6** – is a consequence of the evolution of the Mapping 2, in this one because of the evolution it has only a mismatch, the Sub-Class Attribute as in the other mapping.

As can be seen, it is missing three mappings to describe. These mappings are the evolutions, and as such deserve more attention, are described below:

- **Test Case 1 of the Mapping 9** – This test case is to explain the red circle 1, which is a mismatch of the type Coverage, consequence of the change in the requirements. Since it was needed to change the type of the engine, and in the Model A, the type Turbofan didn't exist. This forced to have an evolution in the model of the Manufacturer A (A→A'). So the MIRAI detected the Versioning in the model and proposed a new mapping (Choice), that is represented in Figure 8-14 with the Mapping 6. With this change comparing it is possible to see in the Mappings 2 and 6, that one of the Mismatch disappear, the Coverage, since the Model A' has the same information that the Model D;
- **Test Case 2 of the Mapping 7** – This test case is a change in the Model C (red circle 2), this is the supplier that produce the pylon for the turbine. This supplier started to do some tests in the pylon and introduced in the model a field with the result of this tests (Simulation Results), by doing this it was made an evolution (C→C'). This versioning caused a break in the system, the Model A doesn't know what this Simulation Results are, so the MIRAI reacts and proposes to create a morphism with the Mismatch of the type Coverage (Mapping 7), this new mapping will avoid problems of interoperability in the future (C'→A);
- **Test Case 3 of the Mapping 8** – This test case is about a change in the Model B (red circle 3), this supplier is using a unit that does not belong to the international system of units, so it was made an update and the unit was changed. It was passed the unit pound (Lbs) to kilogram (KG), and this is a versioning in the system (Encoding), the model suffers an evolution (B→B'). The MIRAI will react to this change and will propose a new morphism, that is the mapping 5 (Equal), it's equal since at this moment the two models have the same unit. Now, this new morphism creates a relation between the Model A and B'.

With this example, it is possible to see how the MIRAI reacts when some modification in the CM occurs, and the advantages that MIRAI brings when used to monitoring models in SE relations. In Figure 8-14 it is possible to see all the mappings explained above, and it is following the mapping described in Figure 8-15. In the CM several Mapping's that represented the relations between two or more models exists, when a Versioning in that model occurs, this forces the Mapping to be represented by another mapping which is his evolution, this new mapping was called the Choice because the user is the one that has to choose.

In resume, this scenario is a validation of the architecture presented in Chapter 4.2, it represents the identification of problems in the mappings and then proposes to the user a solution to recover the interoperability in the network, representing Phase 2. After the user confirms the change, executes the change and notifies the network of this situation, giving the other network enterprises the ability to verify that this change has an impact on them, representing Phase 5. Finally, to detect such changes, the framework has to monitor the mappings, this is Phase 6. This work was validated through the CRESCENDO project described in Chapter 1.4.5 and with the publications 1, 2, 3 and 5 represented in Chapter 8.1.

8.2.5. Scenario for Process Modeling Approach for the Liquid-Sensing Enterprise

The emergence of 3D Printers has made the market of customizable products grows exponentially, providing anyone (end user or manufacturing stakeholder) with the possibility to print a custom piece on demand. However, printers (especially low-cost ones) are still far from being a reliable option due to production times, very delicate conditions and configuration, and high failure rates. Hence, depending on the size or quality of the piece, printing can take many hours, and whenever an undetected problem occurs in the printing process, in addition to the huge waste of time there is the amount of wasted raw material. For this reason, it is important to monitor the printing and ensure the best possible approach to save time and material in face to such situations.

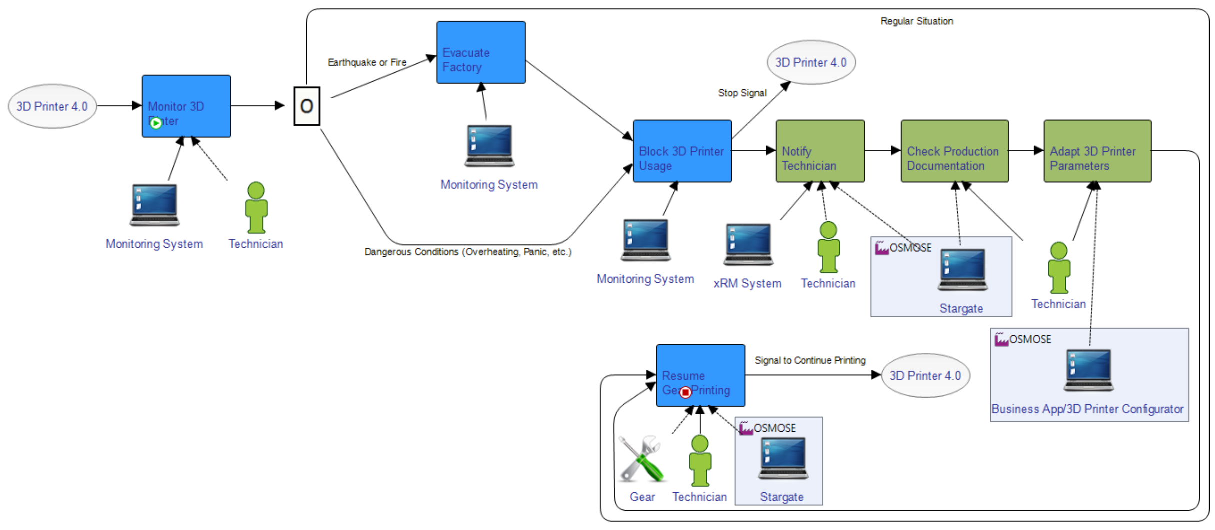


Figure 8-16: Actigram for an Emergency Management Process.

In open demonstrator scenario, OSMOSE (Chapter 1.4.3) is applied to better manage the process of monitoring a standard 3D printer that is producing a gear, providing the solutions to handle predictive maintenance and emergency management. In an emergency, the printer has to stop to ensure that the current the work is not ruined, and potentially saving many working hours. Also, if it is possible to predict possible printing error or hardware failure, then maintenance procedure can be triggered, avoiding significant losses. Throughout this sub-chapter is described the various phases of this scenario, following the description of Figure 6-9.

<u>OBM Models – Business Actigram Instances</u>

In this phase, the OSMOSE middleware framework enables to design and specify in a business-friendly interface, the activities that describe the different flows of monitoring 3D Printer process. The OSMOSE Process Modelling Toolbox is used to design and specify the OBM Model in EA* notation. For the specific user, the story described before, two main process flows have been identified: a) Emergency Management and b) Predictive Maintenance (only the Emergency Management example is going to be demonstrated in this dissertation).

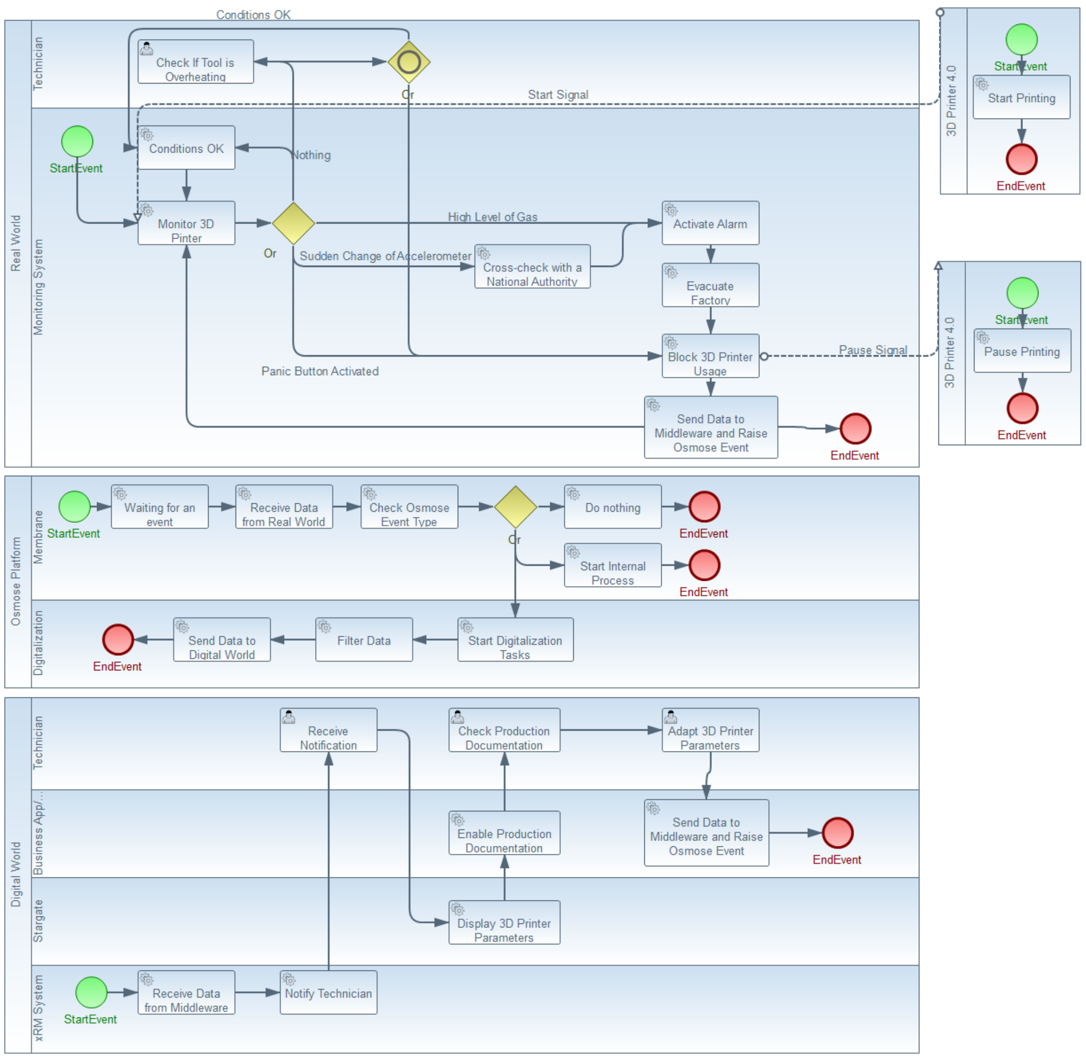


Figure 8-17: Digitalization Process to Notify Technician.

In Figure 8-16 depicts the high-level overview of the Emergency Management process, where sensors available in the printer are used to track and monitor the Gear printing, hence enabling to detect when and which problems occur. The sensors used for this process are the temperature sensor, accelerometer sensor, gas sensor and a panic button. Using these sensors, the idea is to develop a system capable of detecting the real-world events bellow and managing the subsequent activities to prevent material waste:

- **Earthquake** – In this situation an accelerometer is used to detect abrupt oscillations to determine if it is an earthquake, stopping the production in the case of occurring one. This prevents the workers, product and the printer itself from damage (external to the earthquake);
- **Fire** – In this situation, the temperature and gas sensors are used to detect a fire. This is made by validating a high temperature together with an increase of CO2 in the air. In the case of detecting a fire the system stops the production;
- **Panic button** – In the case of a dangerous situation (e.g. burnt hand in the printer bed; hand stuck in the printer, etc.) or if the worker identifies that the piece presents flaws during the printing, he/she can press the panic button to pause the production and resolve the situation;
- **Printer overheating detection** – The system is prepared to detect whether the printer reaches very high temperatures, which can cause long-term damage. This option prompts the user to check what is happening to the printer and if possible return it to the ideal temperature.

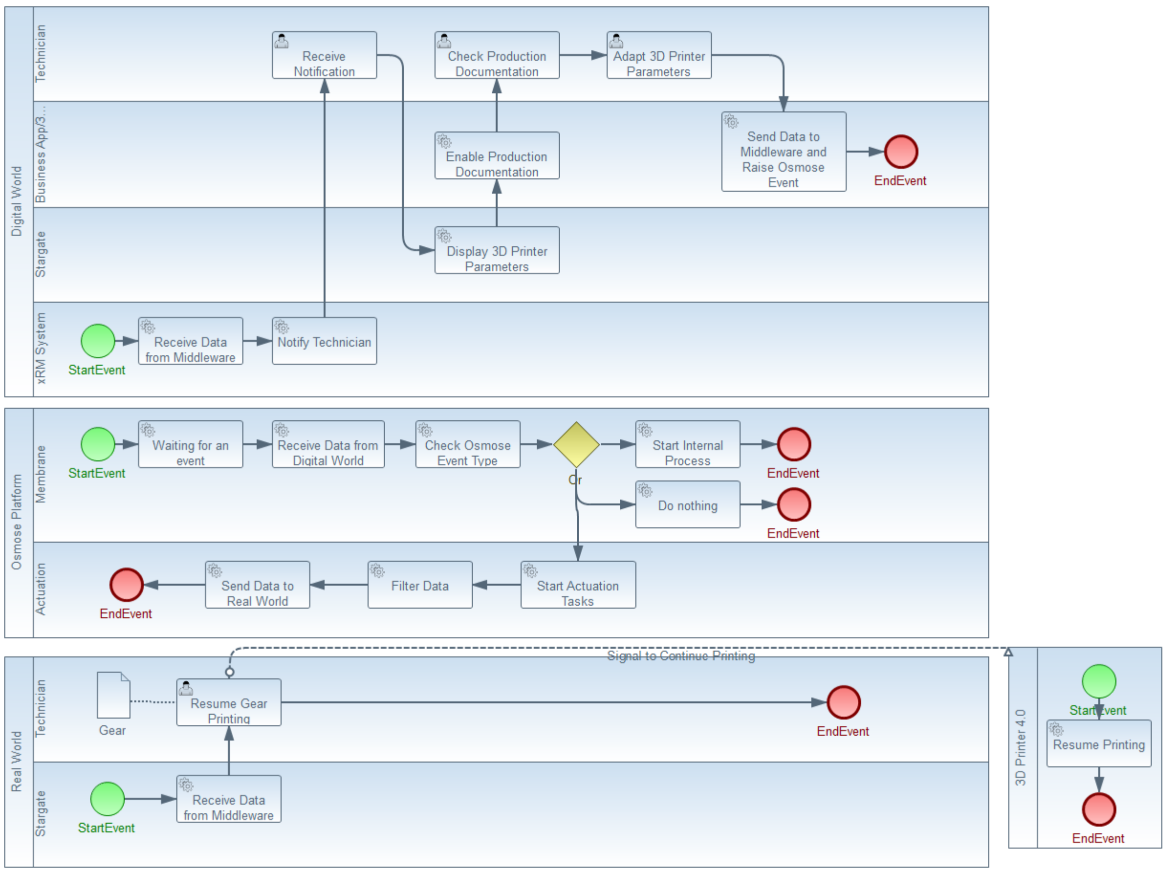
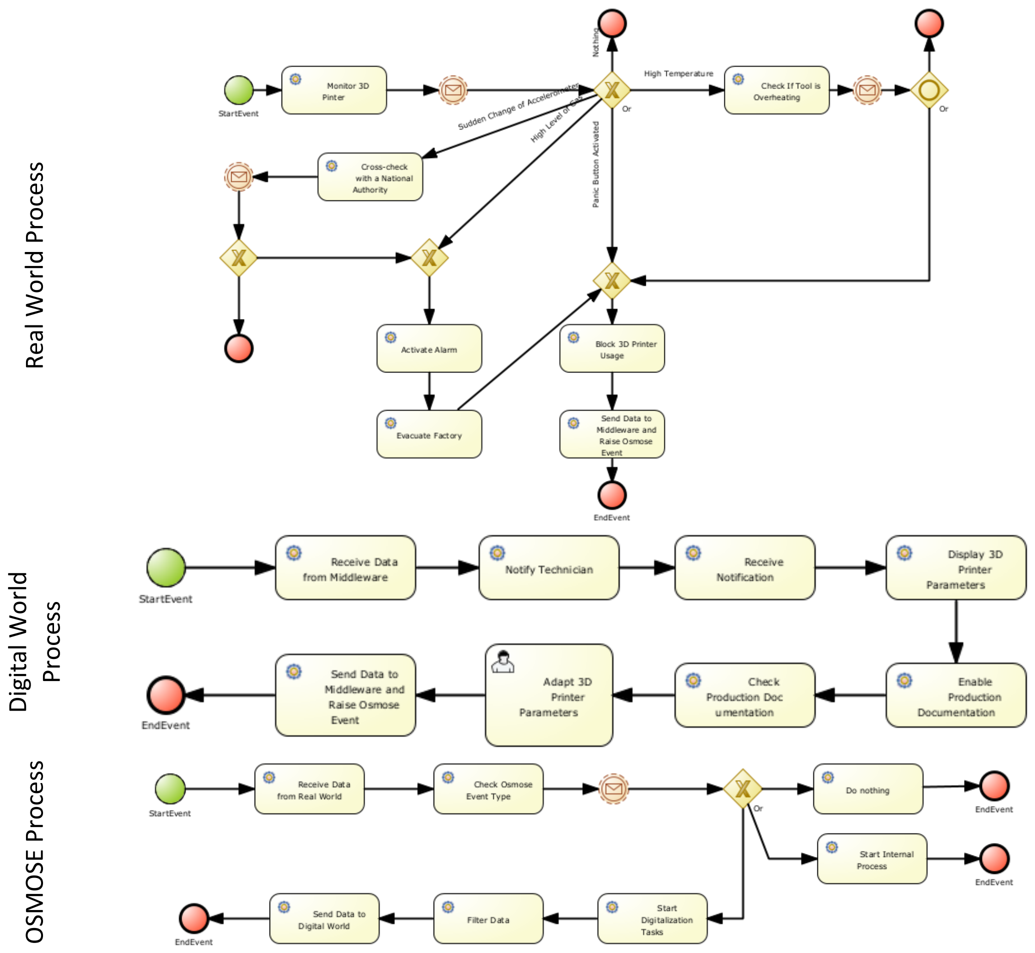


Figure 8-18: Actuation Process to Adapt 3D Printer Parameters.

OTIM – BPMN Process Refinement

In this stage, one automatically derives BMPN2.0 Technical Models from the Actigram Business Models, reusing concepts already defined and simplifying the design of the detailed osmotic behavior. Each BPMN process represents an Osmotic process defined.

The Figure 8-17 represents the Actigram model of the Emergency Management process. By looking to the model, it is possible to identify two Osmotic Processes (a Digitalization Process – from blue to green activities - and an Actuation Process – from green to blue activities), which are represented in Figure 8-17 and Figure 8-18, respectively. Figure 8-17 represents the refined Digitalization process, defining the printer monitoring activities and identifying situations in which the system can block the printer (described in the previous section). It is possible to see that comparing with the Actigram model, this includes much more detail specified by the system architect. At the same time, OTIM describes the transition between worlds. In this case, one can see that the Digitalization crosses the boundaries of the Real World into the Digital World going through the OSMOSE membrane (the transition is made by the detection of osmotic events).



Real World Process
StartEvent
Monitor 3D Pinter
Nothing
High Temperature
Check If Tool is Overheating
Or
Or
Sudden Change of Accelerometer
High Level of Gas
Panic Button Activated
Cross-check with a National Authority
Activate Alarm
Block 3D Printer Usage
Evacuate Factory
Send Data to Middleware and Raise Osmose Event
EndEvent
Digital World Process
StartEvent
Receive Data from Middleware
Notify Technician
Receive Notification
Display 3D Printer Parameters
Enable Production Documentation
Check Production Doc umentation
Adapt 3D Printer Parameters
Send Data to Middleware and Raise Osmose Event
EndEvent
OSMOSE Process
StartEvent
Receive Data from Real World
Check Osmose Event Type
Or
Do nothing
EndEvent
Start Internal Process
EndEvent
Start Digitalization Tasks
Filter Data
Send Data to Digital World
EndEvent

Figure 8-19: BPMN Emergency Management Process - a) Real World, b) Digital World and c) Osmotic World.

Once, the process has identified the problem (real-world event) and the printer is locked, the technician is notified and he becomes responsible to restore the production following a certain set of actions. This is an Actuation process and it is represented in Figure 8-18.

<u>OTSM – BPMN Process Deployment and Parameterization in jBPM</u>

The last phase is the implementation model, which is the refinement of the BPMN so that it can be executed in jBPM concurrently. In this model, the configuration of the BPMN with the services and events is made, to be used in the run-time mode. The representation of the executable Real-World processes is illustrated in Figure 8-19 (for space restrictions we did not include the DW and the OSMOSE membrane, where the model is complemented with more detail and ready to be launched and tested in jBPM, making the osmosis process runnable.

The process is divided into three different processes, as illustrated in Figure 8-19. In this case, the Digitalization process is going to be divided into, Real World, Osmosis World, and Digital World processes. For each situation, it was needed to identify the services needed and configure them in the BPMN to be ready for execution. After this, it is ready for deployment and them for execution.

<u>Open Demonstrator Process Execution</u>

In here, it is briefly described the execution of the processes in the jBPM process execution engine. It is important to note that this step requires that the ones described in the previous sections need to be properly designed and implemented.

Figure 8-20 shows a screenshot of the execution of the digitalization process in the emergency management. In the first image is represented the Real-world process, and it is monitoring the 3D printer was started and the process waits for events from the monitoring. In this example (following the dark tasks), it was detected a high temperature in the extruder, in response the system blocks the printer, with this the Real-world process ends.

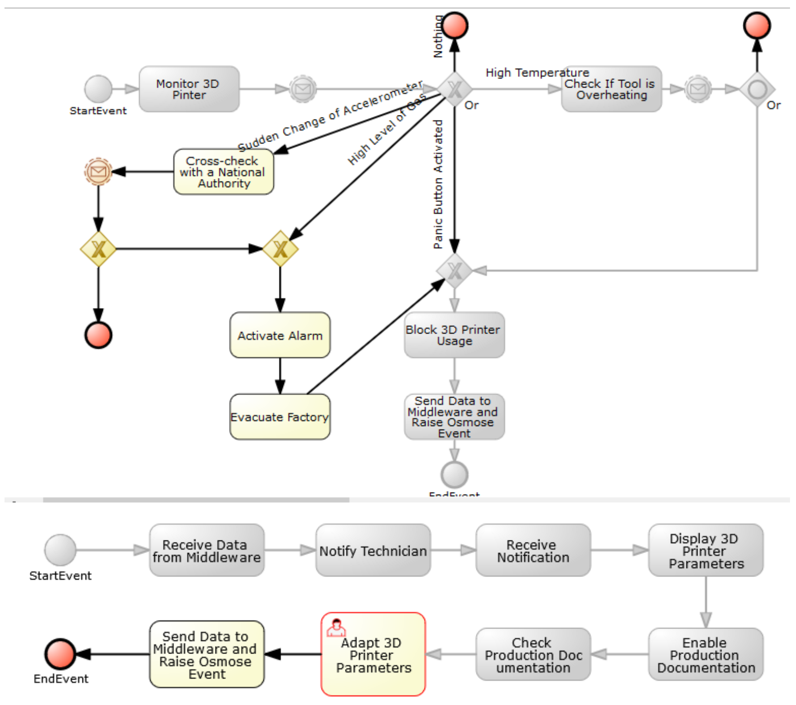

StartEvent
Monitor 3D Pinter
Nothing
High Temperature
Check If Tool is Overheating
Or
Or
Sudden Change of Accelerometer
High Level of Gas
Cross-check with a National Authority
Panic Button Activated
Activate Alarm
Block 3D Printer Usage
Evacuate Factory
Send Data to Middleware and Raise Osmose Event
EndEvent
StartEvent
Receive Data from Middleware
Notify Technician
Receive Notification
Display 3D Printer Parameters
Enable Production Documentation
Check Production Documentation
Adapt 3D Printer Parameters
Send Data to Middleware and Raise Osmose Event
EndEvent

Figure 8-20: Execution of Real and Digital Worlds Emergency Management Process.

At this moment, the Osmosis Membrane started and it is executing the middleware actions to allow to proceed into the next phase of the Digitalization, which in this case is the Digital World Process. In the Digital world, the technician is notified about the possible problem and he is called to verify the state of the printer. Until the technician checks and detects the problem and solve it, the process is blocked until the user checks the validation option. The process finishes with the confirmation by the user, then it restarts to continue to monitor the printer.

This scenario is a validation of the architecture presented in Chapter 4.2, it represents Phase 3 where it is possible for an enterprise to design its products and services processes. Once the processes have been defined, it is possible to parameterize and execute them, this action represents Phase 4 and Phase 5. This work was validated through the CRESCENDO project described in Chapter 1.4.3 and with the publications 13 and 15 represented in Chapter 8.1.

8.3. Hypothesis Validation

In Chapter 1.3.3 the hypothesis of this dissertation was defined, and the objectives of this work were presented. To better recall and revise it, it is here included again:

- "If a framework is able to assess the unity of a network (to verify how the interoperability is implemented), monitor the communications (with the aim of finding changes in the system), and be able to propose solutions to recover the interoperability, then the framework will be able to create a sustainable interoperability system in the network."

The validation of the hypothesis is divided into three discussion points: 1) Assess the enterprise interoperability; 2) Monitor the communications; and 3) to recover the interoperability. By achieving these three main points it is possible to reach the main goal of this dissertation, a sustainable interoperability environment inside of an enterprise network.

The first point is presented in Chapter 7.3, in the Interoperability Assessment corresponding to the Phase 6 of the Methodology developed in this work. By defining the mappings and knowing each enterprise is, it was possible to identify the interoperability between two enterprises and to know if they are interoperable or not. This is done by identifying the number of mappings that can be created and can quantify the ability of two enterprises to be interoperable, hence allowing to validate this first part of the hypothesis. In the same chapter appears the answer to the second point of the hypothesis, the communications monitor. Several types of monitoring mechanisms were developed in order to be able to control the network the monitoring of the EIS models fundamental to data exchange is made through the existing mappings defined between them, whenever there is an evolution in one of the models, one or more mappings become obsolete and the system, proposes solutions to maintain the harmonization of the network. In the monitoring of the devices, a mapping was developed that when detecting that a device has failed or is having inconsistent readings, the system searches which of the existing devices in the network that can replace it without compromising its functionalities, giving the user time to solve this problem. Another monitoring feature is the detection of service failures, allowing to detect and recover the services in that moment. Given that today, web services are an important way of changing the information between the different partners, it is a good achievement of this work.

The last point is how to recover the interoperability in a network in response to the detection of a problem by the monitor phase, this point goes through several stages. After detecting a problem in monitoring, the methodology identifies the cause of the problem and proposes a solution. With this third point, it is shown that it is possible to achieve a sustainable interoperable system in the network, responding to the hypothesis and coming to the conclusion that the author proposed to do is possible.

In the case of external validation, several scientific experts recognized and accepted the various publications that were submitted during this work. In the industrial contribution, there was collaboration in several international projects, where it was possible to validate and contribute to different ideas. In Table 8-1 a collection of validations of the described scenarios is done, demonstrating in this way what the scenarios brought as a solution to the enterprises and as validation for this dissertation.

Table 8-1: Before-Now Scenarios Validation.

Scenarios	Validation Questions	Before	Now
Chapter 8.2.1: Scenario for Innovative Furniture Product Design in Frame of IMAGINE Project	How does the enterprise search for new enterprises to collaborate with?	Contact each other through emails, telephone, fax	Search new partners and contact them via the i-platform
	How does the enterprise share data information with each other?	Send data information by email or fax	Data information is shared with partners as they begin the collaboration
	How does the enterprise share the information?	Send email or fax	The gateway is prepared to upload the information, allowing the interoperability of the information
	How is maintained the interoperability between different data information?	They need to agree and understand the data information exchanged	The gateway is responsible for the transformation of the data information and to be interoperable between the different enterprises
Chapter 8.2.2: Scenario of a Metalwork Manufacturing in Frame of C2NET Project	How does the enterprise know the device network?	Does not know	The devices are registered in the framework, knowing the device network
	How does the enterprise "sense" the devices?	Do not know	Through the framework, the enterprise receives warnings about the state of the devices and the recovered data
Chapter 8.2.3: Scenario for Detection of a Device Malfunction in Frame of C2NET Project	How does the enterprise know if the device network is working properly?	Do not know	The framework has fault tolerance, being able to warn the user
	How does the enterprise recover from a fault tolerance?	need to wait for a human to change the device, which can take a few days	The framework search for a device that can do the same role, as long as the damage is not replaced, avoiding the harmonization breaking
	How does the enterprise choose a new device to replace with the damaged one?	Substitutes it with the same or a similar one	Through the mappings, the framework is able to identify one or more devices that can replace the damaged and propose to the user
Chapter 8.2.4: Scenario for the Monitoring of Morphisms to Support Sustainable Interoperability in Frame of CRESCENDO Project	How does the enterprise warn the network of a problem it has identified?	By telephone or email, and it may take time for others to be warned	The framework has a communication capability that is able to warn the whole network about the problem, giving the ability to other enterprises to verify if this problem has an impact on the network
	How does the enterprise respond to a change in one of the models?	Communicates with the others to advise on the change in the models	The framework identifies the change through the mappings, then searches for a solution and presents the user, managing to maintain interoperability
Chapter 8.2.5: Scenario for Process Modeling Approach for the Liquid-Sensing Enterprise	How does the enterprise can be sensing of the production?	Through a human that accompanies the production	By modeling the processes, it is possible to sense the production
	How can the enterprise monitor production?	A human is responsible for tracking production, which can make it difficult to keep up with in real time	Through the process it is possible to define critical points in the framework, being possible to follow and identify the problems in real time, warning the user

9. FINAL CONSIDERATIONS AND FUTURE WORK

Adaptive systems theory is taken as a basis for this work, considering that collaborative networks are a complex macroscopic collection' of relatively similar and partially connected microstructures, formed to adapt to the changing environment and increase their survival as a macro-structure. It is particularly difficult to maintain the interoperability in such enterprise collaborative network. This is mainly because of how EIS systems are designed since each system usually has different information models and interfaces.

To assure the sustainability of interoperability in a network of EIS, it is necessary to create and maintain seamless relations between the heterogeneous systems, its standards, and models. In this dissertation, it was made a study on several EIS modeling paradigms, to identify how they work and how they represent the enterprises and their own resources, allowing the reuse of proven methodologies and systems' models in a semi-automatic generation of software adapters to enable SI. However, when they want to cooperate with other enterprises, EIS and data exchange services can become problematic, since they hardly find another enterprise with the same the model's structure. In these situations, it is necessary to understand how to make this data exchange compliant developing services that automate the exchange of data. Although this situation becomes effective as it reaches the desired result, it is not free of problems whenever an enterprise changes something in its informal system, or when there are failures.

To identify the extent to which enterprises are interoperable, a study was carried out on the interoperability evaluation methodologies in the enterprise. The purpose of this study was to find a way to identify the level of interoperability between two enterprises to know what information can be extracted from each data model to convey information between enterprises. At the same time, it was necessary to verify to what extent this process can be automated, as the system identifies what is interoperable and how to to use this information automatically. This methodology needs to be able to evaluate, quantify and qualify the interoperability process from the initial phase to the implementation of the work. Once this state is obtained, the systems know the level of interoperability in the network, allowing knowing where problems can occur in each enterprise and what type of information can be exchanged between them.

By being able to classify the level of interoperability in businesses and their data models, a step towards achieving interoperability is given, and the network can be harmonized so that enterprises can exchange information between them. This interoperability is achieved by defining the mappings between the different data models. These mappings allow the identification of the information that is possible to transmit from one side to another and how this information is sent. The next step is to use these mappings to implement data exchange services in a framework that could create the much-desired interoperability on the network. Of course, maintenance is necessary because there may be situations that can destabilize the network, and create a harmonization breaking.

To ensure that these situations do not happen, a monitoring study was conducted to identify ways to monitor the collaborative network. This study looks at ways to monitor model mappings, device problems, or a failing service. These situations can create a disruption in the network's interoperable environment. However, when looking for problems before they cause damage, it is possible to avoid complicated situations for the network and save time, which is a valuable time in a collaboration. By completing this circle, which has been described so far, it is possible to create a sustainable interoperable environment in an enterprise network.

This work implemented proposes to create a system that can realign information, accommodating changes of devices and knowledge available through the different EISs, regardless of the standards or information models in use within digital and sensing enterprise networks. With this type of system, an enterprise can evolve at its own pace, as well as manage daily issues without decreasing the collaboration efficiency. Enterprises can enter and exit the network, and the sustainable system, in a semi-automatically approach, identifies the various components of the collaborative environment, providing the context for complex system behavior. This system was called Framework to Enable IoT in a Sustainable Interoperable Environment.

This SSIF framework is constituted by six phases, used to manage the collaborative network and maintain it interoperable. These phases identify the enterprises, their EIS models, devices, and services to know their structure and identify the next approach to add the enterprise to the network without putting the sustainability of the same in jeopardy. In this way, it is possible to create an interoperability environment based on the identification made through the enterprise's systems and the evaluation made in each of them, knowing it needs to map the data models and devices. At the moment interoperability is achieved, it is necessary to create measures to keep it sustainable, in this way it is used to monitor the cooperative network, gaining the ability to detect problems that occur before creating greater damage.

Throughout this dissertation were demonstrated different implementations, each of these implementations to be able to validate the different phases proposed in the methodology. It was possible to validate some phases, others, however, were encountered some difficulties, as for example in phase 1 in which the device registration is validated, however, the part of data models, was not fully validated, lacking the automation of the extraction of the data. In phase 3 the simulation of the devices was presented a solution and validated, but in relation to the data, models need more study to identify the best way to simulate these failures and being able to calculate the impact it has on the network. From the tests part, industrial tests were done, but unfortunately, not everything was possible to have been tested at the industrial level, in this situation is the SMAP and the MIRAI that has potential to be validated in the industry, such an opportunity did not arise, waiting that in the future it will happen.

The hypothesis was answered in Chapter 8.3 and explained that it was validated throughout this dissertation. Demonstrating that it is possible to create a sustainable interoperable environment in a collaborative network, with the SSIF framework proposed and validating with the different industrial scenarios presented.

9.1. Main Scientific and Technical Contributions

The main contributions of this dissertation are:

- **Framework to Enable IoT in a Sustainable Interoperable Environment** is the major contribution of this dissertation. Being an evolution of the CAS-based Framework to Support Sustainable Interoperability (CAS-SIF), this framework provides the means to register and integrate both systems and devices in a collaborative network in an interoperable way, enabling them to start interacting with each other. After its integration, the framework to enable IoT in a sustainable interoperable environment enables to monitor the collaborative network and identify and react to situations that can generate harmonization breakings;
- **Methodology for a Sustainable Interoperable Network** appears in response to the developed framework, having been designed with the intention of describing each of the six phases, starting with the extraction of knowledge of the models and ending with the evaluation of the enterprise. Throughout this dissertation was described of the advantage of each phase and its methodology and architecture, being demonstrated that when the six phases can be implemented in a collaborative network, it is possible to reach a sustainable and interoperable environment. Although it has been referenced at all stages, the methodology has never been presented as a whole, hence it is possible to find it in the Annex **Error! Reference source not found.**;
- **Device Knowledge Base** serves as an abstract interface to connect the devices regardless of the technology used. Using this KB it is possible to register the devices and facilitate their insertion in the network;
- **DEVMAN,** a framework with an interface to support users to registers their devices. Uses the device knowledge base to store these logs and to support the user throughout the registration process;
- **Semantic Mapping for Device** is used to represent the mappings between devices that can perform the same or similar tasks. Whenever a device fails it is possible to present a solution within the existing devices in the network;
- **SMAP** is an interface framework to support users in the use of semantic mappings. With these mappings, it is possible to present solutions for failing IoT devices. Another feature is to be able to simulate a network of devices, allowing to anticipate failures and verify how the framework reacts and what mappings it proposes;
- **Methodological Framework for Detection of Harmonization Breaking in Service Environments** is a framework to detect service failures, being able to identify the failure and suggest solutions to recover the service.

9.2. Future Work

Despite having proven the concept proposed, the work is far from complete, having some open points that will continue to be developed and improved, in other points that need to incorporate new ideas that can facilitate the previous ones. The future work still to be done was divided into the different phases described in the framework. The **Phase 0** is related to the registration of the enterprises. In this phase, several parts can be improved or it can be changed approach used. An interesting feature to add to the registration of devices could be the study of neural networks to improve the intelligence behind the identification of a device to accommodate more branches and nodes in the data analysis tree in order to increase identification efficiency. Many devices are considered to be "dumb", with very low processing or connectivity capabilities, therefore as IoT devices become smarter it will be possible to decentralize the processing features of cloud platforms, therefore focusing on the knowledge. With the adoption of reference solutions that can implement standard protocols with privacy and security for real-time data collection, the manufacturing paradigm will forever change. In my opinion, industries are actively trying to improve their process from top to bottom, however, the economic and adaptation steps are making the transition to modern systems difficult. There's no question that in the future industries and business will benefit from highly technological systems like artificial intelligence or real-time analysis of big data. In the case of the enterprise, registration could be to improve the framework including the addition of knowledge reasoning functionalities to the adapter, namely multi-language capabilities, which will enable the platform to avoid erroneous translations between concepts described in different languages. This will allow manufacturing enterprises from various countries to connect to the platform, regardless of the native language used in their legacy systems. Another requirement that is missing at this stage is to define minimum requirements for an enterprise to be able to access the platform. Given that if the enterprise is not able to create interoperability with another enterprise, it becomes difficult for the framework to react and begin to create the necessary conditions to achieve this goal.

Phase 1 is related to the knowledge extraction from the models and devices, the knowledge extraction part of the methodology is to be further developed, creating a solid basis for automatic reasoning. Also, the management of such complexity and number of relations requires an evolved model management human-interface, supporting decision making and enabling simulation of transients in the different networks each enterprise belongs to. Complex Event Processing (CEP) and MAS (Multi-Agent System) are examples of candidate technology that may improve the monitoring of networks. The **Phase 2** is related to the interoperability driver specification, in this part is used the two different mappings to relate the models and the devices. In this part, the identification of the mismatches can be improved, by choosing an intelligent system to support the user in the decision of the mappings, improving and accelerating the insertion of mappings. The **Phase 3** is related to the modeling and simulation, in the modeling design is intended to enrich LSE environment tool with the osmosis events pallet, so the osmosis processes modeling can be facilitated. In the devices simulation, can intelligence behind the matching of the device, to accommodate more branches and nodes in the data analysis tree in order to increase matching efficiency, by using (for example) neuronal net. With the adoption of reference solutions that can implement standard protocols with privacy and security, which allow real-time data collection, the manufacturing paradigm will forever change. With these changes will arise huge challenges, since industries will demand more intelligent systems. **Phase 4** is related with the runtime configuration, can be improved the deployment to jBPM, enabling the import of the BPMN directly to the jBPM reducing the time to execution and improving the instantiation and parameterization of the activities, since it is made at the same level. **Phase 5** is related with the system evolution, where the modifications of the system are executed and notify the network about these modifications, in this sense, it is possible to improve the communication and the sending of the messages in order to be able to accelerate the recovery of the harmony. The **Phase 6** is related with interoperability assessment, as already described, this work needs more development, since it is in a very preliminary stage, both at the methodology level and at the level of validation with enterprises, in order to verify if this developed method is viable.

10. REFERENCES

Adler, G.H. & Levins, R. (1994) The Island Syndrome in Rodent Populations. *The Quarterly Review of Biology*, 69 (4), pp.473–490.

Agostinho, C., Correia, F. & Jardim-Goncalves, R. (2010) Interoperability of Complex Business Networks by Language Independent Information Models. In: *17th ISPE International Conference on Concurrent Engineering CE2010*. Springer-Verlag. Available from: <http://www.springerlink.com/content/q775010855762526/>.

Agostinho, C. & Jardim-Goncalves, R. (2009) Dynamic Business Networks: A Headache for Sustainable Systems Interoperability. *In 4th Int. Workshop on Enterprise Integration, Interoperability and Networking (EI2N'2009) of the OTM Conference*.

Agostinho, C. & Jardim-Goncalves, R. (2015) Sustaining interoperability of networked liquid-sensing enterprises: A complex systems perspective. *Annual Reviews in Control*, 39, pp.128–143.

Agostinho, C., Pinto, P. & Jardim-goncalves, R. (2014) Dynamic Adaptors to Support Model-Driven Interoperability and Enhance Sensing Enterprise Networks. In: *9th World Congress of the International Federation of Automatic Control (IFAC'14)*. Cape Town, South Africa.

Agostinho, C., Sarraipa, J., D'Antonio, F. & Jardim-Goncalves, R. (2007) Enhancing STEP-based Interoperability Using Model Morphisms. *3rd International Conference on Interoperability for Enterprise Software and Applications IESA'07 (2007)*, 67.

Agostinho, C., Sarraipa, J., Goncalves, D. & Jardim-Goncalves, R. (2011) Tuple-Based Semantic and Structural Mapping for a Sustainable Interoperability. In: *Ifip International Federation For Information Processing*. pp.45–56.

Agostinho, C., Sesana, M., Jardim-Gonçalves, R. & Gusmeroli, S. (2015) Model-Driven Service Engineering Towards the Manufacturing Liquid-Sensing Enterprise. In: *Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development (ModelsWard 2015)*. Angers, France, pp.608–617.

Agostinho, C.M.M. (2012) Sustainability of Systems Interoperability in Dynamic Business Networks. Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa (FCT/UNL). Available from: <http://hdl.handle.net/10362/8582>.

Andres, B., Sanchis, R. & Poler, R. (2016) A Cloud Platform to support Collaboration in Supply Networks. *International Journal of Production Management and Engineering*, 4 (1), pp.5–13.

Asare, P., Broman, D., Lee E., A., Torngren, M. & Sunder, S. (2012) Cyber-Physical Systems [Internet]. Available from: <http://cyberphysicalsystems.org/> [Accessed 12 September 2013].

ATHENA (2010) MDI - Model Driven Interoperability.

Athena IP (2004) EC FP6-507849. Available from: <http://cordis.europa.eu/project/rcn/72762_en.html>.

ATHENA IP (2007a) ATHENA Public Deliverable D.A4.2 - Specification of Interoperability Framework and Profiles, Guidelines and Best Practices.

ATHENA IP (2007b) *Business Interoperability Framework Work package – B3.1-4*.

ATHENA IP (2012) Enterprise interoperability maturity model (EIMM) [Internet]. Available from: <http://athena.modelbased.net/methodology/eimm.html> [Accessed 30 September 2013].

ATHENA Project (2007) *D.B3.1 - Business Interoperability Framework*.

Bazoun, H., Zacharewicz, G., Ducq, Y. & Boye, H. (2013) Transformation of Extended Actigram Star to BPMN2.0 and Simulation Model in the frame of Model Driven Service Engineering Architecture. In: *DEVS'13 - Proceeding of the Symposium on Theory of Modeling & Simulation*.

Bellifemine, F., Caire, G. & Greenwood, D. (2007) *Developing Multi-Agent Systems with JADE*.

Bellifemine, F., Caire, G., Greenwood, D., Bellifemine, D., Caire, F. & Greenwood, G. (2007) *Developing Multi-Agent Systems with JADE*. Wiley.

Benguria, G. & Larrucea, X. (2007) Data model transformation for supporting interoperability. In: *Commercial-off-the-Shelf (COTS)-Based Software Systems, 2007. ICCBSS '07. Sixth International IEEE Conference*.

Bermudez-edo, M., Elsaleh, T., Barnaghi, P. & Taylor, K. (2015) IoT-Lite Ontology [Internet]. Available from: <https://www.w3.org/Submission/iot-lite/>.

Bernstein, P.A. (2003) Applying Model Management to Classical Meta Data Problems. In: *1st Biennial Conference on Innovative Data Systems Research*. California.

Bharadwaj, A., Choobineh, J., Lo, A. & Shetty, B. (1993) Model Management systems a survey. *Journal of Annals of Operations Research*, 38 (1).

British Standards (2007) Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality requirements.

CalgaryScientify (2013) Interoperability Could Reduce U.S. Healthcare Costs by Thirty Billion [Internet]. Available from: <https://www.calgaryscientific.com/blog/bid/284224/interoperability-could-reduce-u-s-healthcare-costs-by-thirty-billion>.

Camarinha-Matos, L. (2010) Scientific Research Methodologies and Techniques - Unit 2: Scientific Method. *Phd Program in Electrical and Computer Engineering*.

Chen, D., Doumeingtsb, G. & Vernadat, F. (2008) Architectures for enterprise integration and interoperability: past, present and future. *Journal of Computer Science*, 59 (7).

Chen, D. & Vernadat, F. (2002) Enterprise Interoperability: A Standardisation View. In: *Proceedings of IFIP TC5/WG5.12 International Conference on Enterprise Integration and Modeling Technique: Enterprise Inter- and Intra-Organizational Integration: Building International Consensus*.

Compton, M., Barnaghi, P. & Bermudez, L. (2012) The SSN Ontology of the Semantic Sensor Networks Incubator Group. *Web Semantics: Science, Services and Agents on the World Wide Web*, 17, pp.25–32.

CRESCENDO (2010) *BDA Model Store Specifications Dossier – (intermediate version M12). Public Deliverable D5.2.4.*

Danila, C., Stanescu, A.M., Jardim-Goncalves, R., Agostinho, C., Florea, A. magda & Serbanescu, C. (2013) Sustainable Interoperability of Sensing Enterprise. In: *In Proceedings of 7th IFAC Conference on Manufacturing Modelling, Management, and Control*. St. Petersburg, Russia.

Davenport, T.H. (1998) Putting the enterprise into the enterprise system. *Harvard business review*, 76 (4), pp.121–31. Available from: <http://www.ncbi.nlm.nih.gov/pubmed/10181586>.

Ding, Y. & Fensel, D. (2001) Ontology Library Systems : The key to successful Ontology Re-use.

Ducq, Y., Chen, D. & Alix, T. (2012) Principles of Servitization and Definition of an Architecture for Model Driven Service System Engineering. In: M. van Sinderen, P. Johnson, X. Xu, & G. Doumeingts eds. *4th International IFIP Working Conference on Einterprise Interoperability (IWEI 2012)*. Lecture Notes in Business Information Processing. Harbin, China, China, Springer, pp.117–128.

Ducq, Y., Chen, D. & Alix, T. (2012) Principles of Servitization and Definition of an Architecture for Model Driven Service System Engineering. In: *4th International IFIP Working Conference on Enterprise Interoperability (IWEI 2012)*. Harbin, China, Springer.

Edwards, G., Garcia, J., Tajalli, H., Popescu, D., Medvidovic, N., Sukhatme, G. & Petrus, B. (2009) Architecture-Driven Self-Adaptation and Self-Management in Robotics Systems. In: *ICSE'09 Workshop in 31st International Conference on Software Engineering*.

ENSEMBLE Project (2011) Deliverable D2.3: EISB Basic Elements Report. Available from: <http://cordis.europa.eu/docs/projects/cnect/8/257548/080/deliverables/001-ENSEMBLED23EISBBasicElementsReportv100.pdf>.

ETSI (1995) ETS 300 406 - Methods for Testing and Specification (MTS); Protocol and profile conformance testing specifications ; Standardization methodology.

ETSI (2003) TS 102 237-1. Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; Interoperability test methods and approaches; Part 1: Generic approach to interoperability testing.

Del Fabro, M.D., Albert, P., Bézivin, J. & Jouault, F. (2009) Achieving Rule Interoperability Using Chains of Model Transformations. In: pp.249–259.

Ferreira, J., Agostinho, C., Sarraipa, J. & Jardim-Goncalves, R. (2012) Monitoring Morphisms to Support Sustainability of Interoperability in the Manufacturing Domain.

Ferreira, J., Agostinho, C., Sarraipa, J. & Jardim-Goncalves, R. (2011) Monitoring Morphisms to Support Sustainable Interoperability of Enterprise Systems. *On the Move to Meaningful Internet Systems OTM 2011 Workshops EI2N 2011*, 7046 (Springer), pp.71–82. Available from: <http://www.lisealab.it/tecnopolo/download/SustainableInteroperability/Ferreira et al. - 2011 - Monitoring morphisms to support sustainable interoperability of enterprise systems-annotated.pdf> [Accessed 7 December 2011].

Ferreira, J., Beca, M.F. De, Agostinho, C. & Jardim-goncalves, R. (2012) Monitoring Morphisms for Advanced Interoperability in Factories of the Future (FoF) Platforms. In: *MITIP'12 - Modern Information Technology in the Innovation Processes of Industrial Enterprises*.

Filipe Correia (2010) Model morphisms (MoMo) to enable language independent information models and interoperable business networks.

FInES Cluster (2009) Future Internet Enterprise Systems (FInES) Cluster Position Paper: Final Version [Internet]. Available from: <http://ec.europa.eu/digital-agenda/events/cf/ict2010/document.cfm?doc_id=12512> [Accessed 7 October 2013].

FInES Cluster (2011) Future Internet Enterprise Systems (FInES) Position Paper on Orientations for FP8 'A European Innovation Partnership' for European Enterprises: Version 3.0 [Internet]. Available from: <http://ec.europa.eu/digital-agenda/events/cf/ict2010/document.cfm?doc_id=12512> [Accessed 7 October 2013].

FInES Cluster (2012) Sensing Enterprise [Internet]. Available from: <http://www.fines-cluster.eu/mw/index.php/Sensing_Enterprise> [Accessed 15 November 2013].

FITMAN (2015) FITMAN - DynamicCEP. Available from: <http://demos.txt.it:8095/?portfolio=dynamic-cep-dycep>.

Frankston, B. (2009) Ambient Connectivity: An Introduction. Available from: <http://frankston.com/public/?n=IAC>.

Frankston, B. (2013) The Internet of Things Versus the Access Framing. *IEEE Consumer Electronics Magazin*.

FunStep (1999) TVI - Testing and validation of funStep standard implementations. Available from: <http://gris-public.uninova.pt:8080/funStepServices/TVI.jsp>.

GemStone (2005) GemFire Real-Time Events: Technical White Paper. , p.16. Available from: <http://www.gemstone.com/pdf/GemFire_RTE_WP.pdf>.

Van Gigch, J.P. (1991) *System Design Modelling and Metamodeling*. Springer.

Gilad, B. (2008) The Future of Competitive Intelligence: Contest for the Profession's Soul. *Competitive Intelligence Magazine*, 11 (5), p.22.

Grangel, R., Bigand, M., Bourey, J.-P.J.P. & J.P.Bourey (2008) A UML Profile as Support for Transformation of Business

Process Models at Enterprise Level. In: *MDISIS 2008*. pp.73–87.

H. Bazoun, G. Zacharewicz, Y.D. and H.B. (2013) Transformation of Extended Actigram Star to BPMN 2.0 in the frame of Model-Driven Service Engineering Architecture. In: *Symposium on Theory of Modeling and Simulation (TMS/DEVS 2013)*. San Diego, USA.

Hallsteinsen, S., Stav, E. & Floch, J. (2004) Self-Adaptation for Everyday Systems. In: *Proceeding WOSS '04 Proceedings of the 1st ACM SIGSOFT workshop on Self-managed systems*.

Honour, E. (2008) Systems Engineering and Complexity. In: *INCOSE Insight*. INCOSE Insight, p.20.

HTC Global Services (2016) Managing Real-time Data Streaming Effectively with CEP Solutions - White Paper. , p.13. Available from: <http://www.htcglobalservices.com/accept/files/realtime_datastreaming.pdf>.

Huebscher, M. & Mccann, J. (2008) A survey of Autonomic Computing - degrees, models and applications. *Journal ACM Computing Surveys (CSUR)*, 40 (3).

Hveding, J. (2008) An Aspect-Oriented Approach to Adaptive Systems.

IBM (2003) An architectural blueprint for autonomic computing. In: IBM Tech. rep.

INCOSE (2011) INCOSE [Internet]. Available from: <www.incose.org>.

Initiative, I.I., Minerva, R., Biru, A. & Rotondi, D. (2015) *Towards a Definition of the Internet of Things (IoT)*. IEEE. Available from: <http://iot.ieee.org/images/files/pdf/IEEE_IoT_Towards_Definition_Internet_of_Things_Revision1_27MAY15.pdf>.

International Telecommunication Union (2005) *The Internet of Things*. Available from: <https://www.itu.int/net/wsis/tunis/newsroom/stats/The-Internet-of-Things-2005.pdf>.

ISO (2011a) Advanced automation technologies and their applications — Part 1 : Framework for enterprise interoperability (ISO/DIS 11354-1:2011). Available from: <www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=50417>.

ISO (1993a) INTERNATIONAL STANDARD ISO / IEC 10641.

ISO (1993b) INTERNATIONAL STANDARD ISO / IEC 7498-1.

ISO (2005) ISO/IEC 25000:2005 Software Engineering - Software product Quality Requirements and Evaluation (SQuaRE) - Guide to SQuaRE. Available from: <http://www.iso.org/iso/catalogue_detail.htm?csnumber=35683>.

ISO (2011b) ISO TC184SC5 - Advanced automation technologies and their applications Part 1 Framework for enterprise interoperability. Available from: <www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=50417>.

ISO (2012) Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) – Measurement of quality in use.

ISO, J. & JTC, I. (1994) International Standard ISO/IEC 9646-1. Information Technology-Open Systems Interconnection-Conformance testing methodology and framework-Part 1: General concepts.

iSURF Project (2010) *iSURF Deliverable 8.2.1 Functional and NonFunctional Evaluation Criteria for iSURF Pilot Application*.

Jacobson, I., Ericsson, M. & Jacobson, A. (1994) *The Object Advantage: Business Process Reengineering with Object Technology*. New York, NY, USA, ACM Press/Addison-Wesley Publishing Co.

Jardim-Goncalves, R., Agostinho, C., Sarraipa, J.J., Togores, A.R. de, Nuñez, M.J., Panetto, H.H. & NuneZ, M.J. (2011) *Standards Framework for Intelligent Manufacturing Systems Supply Chain*. S. Renko ed. InTech.

Jardim-Goncalves, R., Agostinho, C. & Steiger-Garcao, A. (2011) A Reference Model for Sustainable Interoperability in Networked Enterprises: Towards the Foundation of EI Science Base. *International Journal of Computer Integrated Manufacturing (IJCIM)*, 25.

Kahan, B. & Kael Consulting (2008) *Review of Evaluation Frameworks*. Available from: <http://idmbestpractices.ca/pdf/evaluation-frameworks-review.pdf>.

Katasonov, A., Kaykova, O., Khriyenko, O., Nikitin, S. & Terziyan, V. (2006) Smart Semantic Middleware for the Internet of Things.

Knoblock, C.A. & Szekely, P. (2013) Semantics for big data integration and analysis. In: *Proceedings of the AAAI Fall Symposium on Semantics for Big Data*.

Kotze, P. & Neaga, I. (2010) Towards an Enterprise Interoperability Framework. In: *Advanced Enterprise Architecture and Repositories and Recent Trends in SOA Based Information Systems: In Conjunction with ICEIS 2010*.

Lampathaki, F., Athens, H.P., Kokkinakos, P., Tucci, C., Alvertis, I., Psarras, J., Koussouris, S. & Viscusi, G. (2015) Future Enterprise, A Roadmap for Sensing Enterprise.

Lampathaki, F., Koussouris, S., Agostinho, C., Jardim-Goncalves, R., Charalabidis, Y. & Psarras, J. (2012) Infusing scientific foundations into Enterprise Interoperability. *Computers in Industry*, 63 (8), pp.858–866.

Lassila, O. (2005) Using the Semantic Web in Mobile and Ubiquitous Computing.

Lemrabet, Y., Bigand, M., Clin, D., Benkeltoum, N. & Bourey, J.-P. (2010) Model Driven Interoperability in Practice: Preliminary Evidences and Issues from an Industrial Project. In: *Proceedings of the First International Workshop on Model-Driven Interoperability*. MDI '10. New York, NY, USA, ACM, pp.3–9.

Lezoche, M., Aubry, A. & Panetto, H. (2012) Formal Fact-Oriented Model Transformations for Cooperative Information Systems Semantic Conceptualisation. In: *Enterprise Information Systems Lecture Notes in Business Information Processing*. Springer, pp.117–131.

Man-Sze, L. (2012) The Sensing Enterprise: Living on the Edge, Doing Business with Dust, and in Search of New Values with

Things.

Marcelino-Jesus, E., Sarraipa, J. & Jardim-Goncalves, R. (2013) An Evaluation Approach for Research Project Pilot Technological Applications. In: *Proceedings of the International Conference on Numerical Analysis and Applied Mathematics 2013 (ICNAAM-2013)*. Rhodes, Greece.

Marques-Lucena, C., Agostinho, C., Marcelino-Jesus, E., Sarraipa, J. & Jardim-Goncalves, R. (2015) Collaborative Management of Requirements Using Semantic Wiki Modules. In: *Control Systems and Computer Science (CSCS), 2015 20th International Conference on*. pp.665–672.

Marques-Lucena, C., Ferreira, J., Sesana, M., Fischer, K. & Agostinho, C. (2016) Process Modelling Approach for the Liquid-Sensing Enterprise. In: *I-ESA'16 - Interoperability for Enterprise Systems and Applications*. Guimarães, Portugal.

Mell, P. & Grance, T. (2011) *The NIST definition of cloud computing*. Available from: <http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>.

Methodologies Corporation (2011) Service-Oriented Modeling Framework (SOMF).

Metzger, A. (2009) *PO-JRA-1.2.1 State of the Art Report, Gap Analysis of Knowledge on Principles, Techniques, and Methodologies for Monitoring and Adaptation of SBAs*.

Microsoft (2004) Understanding Service-Oriented Architecture [Internet]. Available from: <http://msdn.microsoft.com/en-us/library/aa480021.aspx> [Accessed 30 September 2013].

Miller, J. & Mukerji, J. (2003) MDA Guide Version 1.0.1. *OMG*, (June).

Miller, J. & Mukerji, J. (2001) Model Driven Architecture. *Architecture Board ORMSC*, pp.1–31.

Missikoff, M., Schiappelli, F. & Taglino, F. (2003) A Controlled Language for Semantic Annotation and Interoperability in e-Business Applications. In: *2nd Int. Semantic Web Conference (ISWC-03)*. Florida, pp.1–6.

MSEE Project (2012) *D11.2 Service concepts, models, and method: Model-Driven Service Engineering*.

Muccini, H., Sharaf, M. & Weyns, D. (2016) Self-adaptation for cyber-physical systems. In: *Proceedings of the 11th International Workshop on Software Engineering for Adaptive and Self-Managing Systems - SEAMS '16*. New York, New York, USA, ACM Press, pp.75–81. Available from: <http://dl.acm.org/citation.cfm?doid=2897053.2897069>.

Muller, G. (2008) A Reference Architecture Primer. , pp.0–20. Available from: <http://www.gaudisite.nl/>.

Munir, S. & Stankovic, J.A. (2014) FailureSense: Detecting sensor failure using electrical appliances in the home. In: *11th IEEE Int. Conf. Mob. Ad Hoc Sens. Syst. MASS 2014*. pp.73–81.

Nallon, J. (2003) An Overview of AP-233 : The Systems Engineering Data Exchange Standard based on STEP (ISO-10303). *Systems Engineering*. Available from: <www.ap233.org>.

NEC Corporation (2012) User's Guide: System Monitor - Performance Monitoring Services 5.0.

Nikolaus, K. (2013) Building the Nuts and Bolts of Self-Organizing Factories. In: *Siemens Pictures of the Future: Manufacturing and Innovation (Industry 4.0)*.

Ogren, I. (2000) On Principles for Model-Based Systems Engineering. *Systems Engineering Journal*, 3 (1), pp.38–49. Available from: <http://doi.wiley.com/10.1002/(SICI)1520-6858(2000)3:1%3C38::AID-SYS3%3E3.0.CO;2-B>.

Onofre, S. (2007) Plataforma para testes de conformidade de sistemas baseados em módulos conceptuais STEP.

Operations, T. & Crisp II, H. (2007) Systems Engineering Vision 2020. *INCOSE*, Version 2. (September).

ORACLE (2009) Interoperability Guidelines [Internet]. Available from: <http://docs.oracle.com/cd/E21764_01/web.1111/e15184/interop.htm> [Accessed 28 September 2013].

Ould, M.A. & Ould, M.A. (1995) *Business processes: modelling and analysis for re-engineering and improvement*. Wiley Chichester.

Panetto, H. (2007) Towards a classification framework for interoperability of enterprise applications. *International Journal of Computer Integrated Manufacturing*, 20 (8), pp.727–740. Available from: <http://www.tandfonline.com/doi/abs/10.1080/09511920600996419>.

Peristeras, V. & Tarabanis, K. (2006) The Connection, Communication, Consolidation, Collaboration Interoperability Framework (C 4 IF) For Information Systems Interoperability Defining interoperability. *Ibis*, 1 (1), pp.61–72.

Petzmann, A., Puncochar, M., Kuplich, C. & Orensanz, D. (2007) Applying MDA ® Concepts to Business Process Management. *Interchange*, pp.103–116.

Pineda, R., Lopes, A., Tseng, B. & Salcedo, O. (2005) Service Systems Engineering: Emerging Skills and Tools. *Journal in Procedia Computer Science*, 8, pp.420–427.

Preece, J., Rogers, Y. & Sharp, H. (2002) *Interaction Design: Beyond Human-Computer Interaction*. New York, NY, USA, John Wiley & Sons, Inc.

Ray, S. & Jones, A. (2006) Manufacturing Interoperability. *Journal of Intelligent Manufacturing*, 17, pp.681–688.

Reh, F.J. (2013) Key Performance Indicators (KPI) [Internet]. Available from: <http://management.about.com/cs/generalmanagement/a/keyperfindic.htm> [Accessed 1 October 2013].

Sacevski, I. & Veseli, J. (2007) Introduction to Model Driven Architecture (MDA). *Architecture*, (June).

Sakao, T. & Shimomura, Y. (2007) Service Engineering: a novel engineering discipline for producers to increase value combining service and product. *Journal of Cleaner Production*, 15 (6).

Santucci, G., Martinez, C. & Vlad, D. (2012) The Sensing Enterprise. In: *In FInES Workshop at FIA 2012*. Available from:

<http://www.theinternetofthings.eu/sites/default/files/[user-name]/Sensing-enterprise.pdf>.

Sarraipa, J., Jardim-Goncalves, R. & Steiger-Garcao, A. (2010) MENTOR: An Enabler for Interoperable Intelligent Systems. *International Journal of General Systems*, 39, pp.557–573.

Sarraipa, J., Zouggar, N., Chen, D. & Jardim-Goncalves, R. (2007) Annotation for Enterprise Information Management Traceability. In: *ASME 2007 IDETC & CIE*.

Selic, B. (2003) The Pragmatics of Model-Driven Development. *IEEE Software*, 20 (5), pp.19–25. Available from: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1231146>.

Selic, B. (2003) The Pragmatics of Model-Driven Development. *IEEE Software*, 20 (5), pp.19–25. Available from: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1231146>.

Service Web 3.0 Project (2009) *The Future of the Internet of Services for Industry: the ServiceWeb 3.0 Roadmap*.

ServiceNow (2013) KPI Library [Internet]. Available from: <http://kpilibrary.com/> [Accessed 1 October 2013].

Shapiro, J. (2003) Monitoring and Evaluation. Available from: <http://www.civicus.org/view/media/Monitoring and Evaluation.pdf>.

Silberberg, D.P. & Mitzel, G.E. (2005) Information Systems Engineering. *Journal of Johns Hopkins APL Technical Digest*, 26 (5).

Smolik, P.C. (2000) MAMBO Metamodeling Environment. Brno University of Technology.

Spirito, M., Pastrone, C., Soldatos, J., Giaffreda, R., Doukas, C., Muñoz, L., Polidura, V.G., Gusmeroli, S., Sola, J. & Agostinho, C. (2014) Internet of Things Applications - Research and Innovation to Market Deployment (Chapter 7). In: O. Vermesan & P. Friess eds. *Internet of Things – From Research and Innovation to Market Deployment*. River Publishers, pp.243–286.

Staudt, C., Schumm, A., Meyerhenke, H., Gorke, R. & Wagner, D. (2012) Static and Dynamic Aspects of Scientific Collaboration Networks. In: *ASONAM '12 Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*. Washington, DC, USA, pp.522–526.

Sundaram, D. & Wolf, E. (2009) Enterprise Model Management Systems. In: *Proceeding EOMAS '09 Proceedings of the International Workshop on Enterprises & Organizational Modeling and Simulation*.

Terzidis, O., Oberle, D. & Kadner, K. (2012) The Internet of Services and USDL. In: *Handbook of Service Description : USDL and its Methods*.

The Open Group (2013) Service-Oriented Architecture [Internet]. Available from: <http://www.opengroup.org/subjectareas/soa> [Accessed 29 September 2013].

Theunis, J., Stevens, M. & Botteldooren, D. (2017) Participatory Sensing, Opinions and Collective Awareness. *Participatory Sensing, Opinions, and Collective Awareness*, pp.21–46.

Truyen, F. (2006) The Fast Guide to Model Driven Architecture.

Unis, M., Nettsträter, A., Iml, F., Stefa, J., Suni, C.S.D., Salinas, A. & Sapienza, U. (2013) Internet of Things – Architecture IoT - A Final architectural reference model for the IoT v3.

Unis, M., Nettsträter, A., Iml, F., Stefa, J., Suni, C.S.D., Salinas, A. & Sapienza, U. (2013) *Internet of Things – Architecture IoT - A Final architectural reference model for the IoT v3*. 257521 IoT-A Project.

United Nations (1987) *Our Common Future*. Available from: <http://www.un-documents.net/our-common-future.pdf>.

UptimeSoftware (2013) The Truth about Agent vs. Agentless Monitoring - A Short Guide to Choosing the Right Solution.

US Departement of Energy (2011) DOE Standard - Content of System Design Descriptions.

Vermesan, O. & Friess, P. (2013) *Internet of Things - Converging Technologies for Smart Environments and Integrated Ecosystems*. River Publishers.

Vernadat, F. (2001) UEML: Towards a Unified Enterprise Modelling Language. In: *3^{o} Conference Francophone de MOdelisation et SIMulation 'Conception, Analyse, et Gestion des Systemes Industriels', MOSIM'01*.

Vida, M., Stoicu-Tivadar, L. & Bernad, E. (2012) Measuring the Interoperability Degree of Interconnected Healthcare Information Systems Using the LISI Model. In: *INTELLI 2012 : The First International Conference on Intelligent Systems and Applications*. Chamonix, France.

W3Schools (2013) Introduction to Web Services [Internet]. Available from: <http://www.w3schools.com/webservices/ws_intro.asp> [Accessed 30 September 2013].

Weidlich, M., Ziekow, H., Mendling, J., Gunther, O., Weske, M. & Desai, N. (2011) Event-Based Monitoring of Process Execution Violations. In: *9th International Conference, BPM 2011*. Clermont-Ferrand, France, Springer Berlin Heidelberg, pp.182–198.

Wiesner, S., Guglielmina, C., Gusmeroli, S. & Doumeingts, G. (2014) *Manufacturing Service Ecosystem: Achievements of the European 7th Framework Programme FoF-ICT Project MSEE: Manufacturing Service Ecosystem (Grant No. 284860)*. Bremer Sch. http://www.verlag-mainz.de.

WikiMedia (2009) Cloud computing [Internet]. Available from: <http://commons.wikimedia.org/wiki/File:Cloud_computing.svg> [Accessed 30 September 2013].

Wong, E. (1997) Network Monitoring Fundamentals and Standards [Internet]. Available from: <http://www.cse.wustl.edu/~jain/cis788-97/ftp/net_monitoring/index.htm> [Accessed 2 October 2013].

Wooldridge, M. & Jennings, N. (1995) Intelligent Agents: Theory and Practice. *The Knowledge Engineering Review*, 10:2, pp.115–152.

Wooldridge, M., Jennings, N., Wooldridge, N. & Jennings, M. (1995) Intelligent Agents: Theory and Practice. *The Knowledge Engineering Review*, 10:2, pp.115–152.

WSO2 (2015) WSO2 Complex Event Processor. Available from: <http://wso2.com/products/ complex-event-processor/>.

Wuttke, J., Brun, Y., Gorla, A. & Ramaswamy, J. (2012) Traffic routing for evaluating self-adaptation. In: *2012 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. Zurich, pp.27–32.

Yahia, E., Aubry, A. & Panetto, H. (2012) Formal measures for semantic interoperability assessment in cooperative enterprise information systems. *Computers in Industry*, 63 (5), pp.443–457.

Zdravkovic, M., Panetto, H. & Trajanovic, M. (2013) Semantic Interoperability for Dynamic Product-Service. In: *International Conference on Information Systems and Technology (ICIST 2013)*. Kopaonik, Serbia.