

TOWARDS SEMANTICS-DRIVEN MODELLING AND SIMULATION OF CONTEXT-AWARE MANUFACTURING SYSTEMS

Thèse N° 7150

Présentée le 7 juin 2019

à la Faculté des sciences et techniques de l'ingénieur
Groupe SCI STI DK
Programme doctoral en manufacturing

pour l'obtention du grade de Docteur ès Sciences

par

Damiano Nunzio ARENA

Acceptée sur proposition du jury

Prof. Y. Bellouard, président du jury
Prof. D. Kyritsis, directeur de thèse
Prof. B. Smith, rapporteur
Dr F. Demoly, rapporteur
Prof. C. Tucci, rapporteur

2019

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Prof Kiritsis for the support of my PhD study and related research, for his patience, motivation, and friendly attitude, which made me feel home from the very first day when I moved to Switzerland.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof Smith, Prof Demoly, Prof Tucci, and the President of the committee and of my Doctoral School, Prof Bellouard, for their insightful comments, but also for the hard questions which gave me the chance to show the true and multifaceted value of my research.

I would like to thank my parents, my sisters, my adorable four nephews, and my lovely niece for supporting me in every way throughout my academic career and my life in general. Thank you for the sunshine you bring into my life.

My sincere thanks also go to my old and new friends, who have been essential during this long journey. People with whom I have travelled, smiled, cried, had drinks, partied till sunrise, talked for hours and hours about serious and less serious stuff, learned about life and happiness. Thank you all very much, you were, are, and will always be the best part of life.

I thank all my fellow labmates and colleagues for the stimulating discussions, the sincere support and all the fun we have had in the last four years. Without their precious support, it would have not been possible to conduct this research.

Last but not least, I would like to thank my significant other, Claudia. We have been *fighting against all odds. We'll be alright this time. Darling, just hold my hand. Be my girl, I'll be your man. I see my future in your eyes.*

Lausanne, 7 June 2019

Abstract

Systems modelling and simulation are two important facets for thoroughly and effectively analysing manufacturing processes. The ever-growing complexity of the latter, the increasing amount of knowledge, and the use of Semantic Web techniques adhering meaning to data have led researchers to explore and combine together methodologies by exploiting their best features with the purpose of supporting manufacturing system's modelling and simulation applications.

In the past two decades, the use of ontologies has proven to be highly effective for context modelling and knowledge management. Nevertheless, they are not meant for any kind of model simulations. The latter, instead, can be achieved by using a well-known workflow-oriented mathematical modelling language such as Petri Net (PN), which brings in modelling and analytical features suitable for creating a digital copy of an industrial system (also known as "digital twin").

The theoretical framework presented in this dissertation aims to exploit W3C standards, such as Semantic Web Rule Language (SWRL) and Web Ontology Language (OWL), to transform each piece of knowledge regarding a manufacturing system into Petri Net modelling primitives. In so doing, it supports the semantics-driven instantiation, analysis and simulation of what we call semantically-enriched PN-based manufacturing system digital twins.

The approach proposed by this exploratory research is therefore based on the exploitation of the best features introduced by state-of-the-art developments in W3C standards for Linked Data, such as OWL and SWRL, together with a multipurpose graphical and mathematical modelling tool known as Petri Net. The former is used for gathering, classifying and properly storing industrial data and therefore enhances our PN-based digital copy of an industrial system with advanced reasoning features. This makes both the system modelling and analysis phases more effective and, above all, paves the way towards a completely new field, where semantically-enriched PN-based manufacturing system digital twins represent one of the drivers of the digital transformation already in place in all companies facing the industrial revolution.

As a result, it has been possible to outline a list of indications that will help future efforts in the application of complex digital twin support-oriented solutions, which in turn is based on semantically-enriched manufacturing information systems. Through the application cases, five key topics have been tackled, namely: (i) semantic enrichment of industrial data using the most recent ontological models in order to enhance its value and enable new uses; (ii) context-awareness, or context-adaptiveness, aiming to enable the system to capture and use information about the context of operations; (iii) reusability, which is a core concept through which we want to emphasize the importance of reusing existing assets in some form within the industrial modelling process, such as industrial process knowledge, process data, system modelling primitives, and the like; (iv) the ultimate goal of semantic Interoperability, which can be accomplished by adding data about the metadata, linking each data element to a controlled, shared vocabulary; finally, (v) the impact on modelling and simulation applications, which shows how we could automate the translation process of industrial knowledge into a digital manufacturing system and empower it with quantitative and qualitative analytical technics.

Keywords

Ontology Engineering, Petri Nets, Manufacturing Systems, Modelling, Discrete Event Simulation, Industry 4.0, Manufacturing System Digital Twins

Sintesi

Modellazione e simulazione dei sistemi sono due aspetti importanti al fine di analizzare approfonditamente ed efficacemente qualunque processo manifatturiero. La sempre piú crescente complessità di questi ultimi, l'aumento di informazioni e, non ultimo, l'utilizzo di tecniche orientate al web semantico che aggiungono ai dati maggior significato, hanno fatto si che la ricerca iniziasse ad esplorare e combinare tra loro metodologie al fine di trarre vantaggio dalle loro migliori funzionalità e allo scopo ultimo di supportare tutte quelle applicazioni orientate alla modellazione e simulazione di sistemi manifatturieri.

Negli ultimi due decenni, l'utilizzo di ontologie si è dimostrato uno strumento efficace per la rappresentazione di modelli contestuali e gestione delle informazioni. Ciononostante, le ontologie non sono destinate ad eseguire alcun genere di simulazione di tali modelli. Quest'ultima, invece, può essere ottenuta tramite l'utilizzo di un noto linguaggio matematico di modellazione orientato ai flussi di lavoro, quale il formalismo Petri Net (PN), le cui caratteristiche analitiche e di modellazione si prestano alla creazione di una copia digitale di qualunque sistema industriale.

Il quadro teorico presentato in questa tesi di dottorato mira dunque ad utilizzare standard W3C, ad esempio il Semantic Web Rule Language (SWRL) ed il Web Ontology Language (OWL), al fine di trasformare ogni informazione relativa ad un impianto manifatturiero in primitive di modellazione delle Reti di Petri. Così facendo, si supporta l'istanziamento basato su semantica, l'analisi e la simulazione di ciò che chiamiamo copie digitali di un sistema manifatturiero basato su Reti di Petri semanticamente arricchite.

L'approccio proposto in questo lavoro di ricerca esploratorio fa, dunque, leva sulle caratteristiche piú importanti introdotte dai recenti sviluppi degli standard W3C per i Linked Data, come OWL e SWRL, assieme ad uno strumento di modellazione grafica e matematica conosciuto come Reti di Petri. I primi sono utilizzati per raccogliere, classificare e conservare appropriatamente dati industriali, arricchendo quindi la nostra copia digitale di un sistema industriale con funzionalità avanzate di deduzione. Questo rende entrambe le fasi di modellazione e analisi molto piú efficaci e, soprattutto, apre la strada verso un campo di ricerca totalmente nuovo dove le "semantically-enriched PN-based manufacturing system digital twins" rappresentano uno dei driver della trasformazione digitale per le industrie 4.0.

Come risultato, è stato possibile redigere una lista di indicazioni e migliori prassi al fine di aiutare ogni attività futura legata all'applicazione di soluzioni orientate al supporto di complesse digital twin che, a sua volta, sono basate su sistemi informativi manifatturieri semanticamente arricchiti. Attraverso i casi studio, cinque argomenti chiave sono stati affrontati, ossia: (i) l'arricchimento semantico di dati industriali con l'ausilio dei modelli ontologici piú recenti al fine di rafforzare il loro valore e permetterne nuovi usi; (ii) la consapevolezza del contesto, o adattabilità al contesto, che mira a donare al sistema l'abilità di catturare ed utilizzare informazioni riguardo il contesto in cui si svolgono certe operazioni; (iii) la riutilizzabilità, concetto chiave attraverso il quale si suole enfatizzare l'importanza di riutilizzo di asset a risorse esistenti in qualche modo all'interno del processo di modellazione del sistema industriale, come conoscenze specifiche di processo, dati di processo, primitive di modellazione dei sistemi, e così via; (iv) l'obiettivo ultimo di interoperabilità semantica, ottenibile tramite l'aggiunta di dati in merito ai metadati, collegando ogni dato tramite un vocabolario condiviso e controllato; ed infine, (v) l'impatto su applicazioni di modellazione e simulazione di sistemi manifatturieri, che dimostra com'è stato possibile automatizzare il processo di traduzione di informazioni industriali in sistemi manifatturieri digitali aprendo dunque le porte a tecniche analitiche avanzate di carattere sia qualitativo che quantitativo.

Parole Chiave

Ingegneria Ontologica, Reti di Petri, Sistemi Manifatturieri, Modellazione, Simulazione ad Eventi Discreti, Industria 4.0, Copie Digitali di Sistemi Manifatturieri

Contents

Acknowledgements	i
Abstract	ii
Keywords	ii
Sintesi	iii
Parole Chiave	iii
Contents	iv
List of Figures	vi
List of Tables	ix
List of Equations	x
List of Acronyms	xi
Chapter 1 Introduction	1
Chapter 2 Research Method	3
2.1 Research Purpose	3
2.2 Research Questions	4
2.3 Research Strategy	4
2.3.1 Research Contribution	5
2.3.2 Contribution to Practice	5
2.4 Thesis Structure Outline	5
Chapter 3 Background in Ontology engineering	7
3.1 Methodologies for Ontology design, development, and use	8
3.2 Leading Tools for Ontology editing	12
Chapter 4 Background in Modelling and Simulation of Manufacturing Systems with Petri Nets	15
4.1 Modelling with Petri Nets	15
4.2 Manufacturing System Analysis with Petri Nets	17
Chapter 5 Ontologies in support of Manufacturing Systems Modelling and Simulation	23
5.1 Ontologies for Manufacturing Systems	23
5.2 Ontologies for Petri Net Modelling and Simulation	25
5.3 A Semantic Framework for Modelling and Simulation of Manufacturing Systems: the proposed approach ..	28
Chapter 6 Modelling and Simulation of Semantically-Enriched PN-based Digital Twins	29
6.1 Application Case 1: Human Resource Management through Semantically-Enriched Data	29
6.1.1 Design of a Semantic Framework for Smart HR Data Management	30
6.1.2 Walkthrough of the Human Resource Optimization and Task Scheduling Tool Implementation	37
6.1.3 Discussions around the application case	43

6.2	Application Case 2: Reliability Assessment of an Automated Assembly Station	44
6.2.1	Ontology-driven creation of OWL elements for PN-based manufacturing system models: design and preliminary validation tests.....	44
6.2.2	Walkthrough implementation for a real case scenario: Powertrain Systems	64
6.2.3	Discussions around the application case.....	67
6.3	Application Case 3: Automated HAZOP Analysis of a Chemical Plant.....	68
6.3.1	The HAZOP methodology	68
6.3.2	Coloured Petri Nets	71
6.3.3	Context-adaptive Coloured Petri Nets for HAZOP assessment	73
6.3.4	Discussions around the application case.....	90
Chapter 7	Conclusion	91
	References.....	93
	Annex A – List of worker's skills and worker groups	98
	Annex B1 – PN modelling patterns.....	100
	Annex B2 – SWRL-driven PN model creation	103
	Annex C – Typical deviations for process industry	120
	Curriculum Vitae.....	121

List of Figures

Figure 1 From the first industrial revolution to Industry 4.0.....	1
Figure 2 Google Trends graph for “Industry 4.0” in the world.....	2
Figure 3 From RQs to Action Tasks	6
Figure 4 Fragment of a simple taxonomy of motorcycles.....	7
Figure 5 Basic Formal Ontology (BFO) hierarchy	10
Figure 6 Protégé - Ontograp Tab	13
Figure 7 OSF - Ontology properties Tab.....	13
Figure 8 ANZO - Lenses view.....	14
Figure 9 ISO 10303 and IEC 62264 standards within a manufacturing enterprise.....	24
Figure 10 PN on the Semantic Web - UML Diagram of the basic structure	25
Figure 11 PN on the Semantic Web - UML Diagram of the graphical features	26
Figure 12 PN on the Semantic Web - Protégé implementation	26
Figure 13 PNML Core Model.....	27
Figure 14 PNML PT Model	27
Figure 16 Overall proposed approach.....	28
Figure 16 Proposed approach – AS IS (top side), TO BE (bottom side)	30
Figure 17 CIDEM XSD showing the static information of the industry (Workers) [95]	32
Figure 18 CIDEM XSD showing the dynamic information of the industry (Events) [95].....	32
Figure 19 Ontology model excerpt from SatysFactory network of ontologies	33
Figure 20 Examples of skills-based analysis of the Worker Groups A and B	34
Figure 21 RDFizing shop floor data through XSLT	35
Figure 23 Query Structure	36
Figure 23 Worker Suitability Analysis - SPARQL query engine.....	36
Figure 24: Representation of a linear chain CRF	38
Figure 25 Steps of the task scheduling algorithm	39
Figure 26 UI of the HR optimization tool	40
Figure 27 Task schedule before (a) and after (b) the assignment of a new task.....	41
Figure 28: (a) Task list before the new task assignment (2 nd scenario), (b) Output results without semantic information and (c) Output results with semantic information	43
Figure 29 RAAS Ontology elements compliant to BFO.....	48
Figure 30 PN4R Ontology.....	49
Figure 31 Example of the network of object properties	51
Figure 32 Sequence of Assembly Activities and Involved Components.....	52
Figure 33 Stepwise description of the PN creation.....	55
Figure 34 From RDF to PNML: (left side) PN model resulting from the SWRL rules; (right side) PN model formalized according to a tool specific XML data structure	57

Figure 35 W3C: XSL Transformation process	57
Figure 36 Screenshot of the model used for the example of testing	58
Figure 37 Partial ‘Average number of token per place’ results of the example simulation of testing .	58
Figure 38 Partial ‘Transition throughput’ results of the example simulation of testing	59
Figure 39 Semantically enriched results data analysis.....	60
Figure 40 Probability of failure for each component shown through Protégé SPARQL Tab	61
Figure 41 System Failure probability shown through Protégé SPARQL Tab	62
Figure 42 System availability evaluation shown through Protégé SPARQL Tab.....	62
Figure 43 Number of failures for each component shown through Protégé SPARQL Tab.....	62
Figure 44 Total repairing hours grouped by component category	63
Figure 45 Detail of the mechanical ledger	64
Figure 46 Activity-tree representing the logical sequence of the activities	65
Figure 47 HAZOP study in a nutshell.....	70
Figure 48 CPN Tools screen shot (www.cs.au.dk/CPNTools)	72
Figure 49 From Component to Effect through Deviations	74
Figure 50 CPN model declarations.....	75
Figure 51 CPN models for the six deviation scenarios	76
Figure 52 Extract of nodes partition of the plants	78
Figure 53 CPN Model of the Valve	79
Figure 54 CPN Model of the Reactor	80
Figure 55 CPN model of the Filter.....	81
Figure 56 CPN model of the Pump.....	82
Figure 57 CPN model of the Vessel.....	83
Figure 58 CPN Model of the Heat-Exchanger.....	84
Figure 59 CPN-HZP Ontology	86
Figure 60 Machine reasoning-based OWL class assignment	87
Figure 61 (Left side) Context Linked Data on Protégé; (Right Side) CPN model.....	87
Figure 62 Simulation Report	88
Figure 63 Processing the simulation report (Causes and Consequences highlighted).....	88
Figure 64 HAZOP-like report generated from the Simulation Results	88
Figure 65 Proposed sequential pattern and representation in the model.....	100
Figure 66 Proposed parallel pattern and representation in the model	100
Figure 67 Proposed branch ultimate pattern and representation in the model.....	101
Figure 68 Proposed synchronisation pattern and representation in the model.....	101
Figure 69 Proposed repetitive pattern and representation in the model.....	102
Figure 70 SWRL Tab: Step 1. Components.....	103
Figure 71 SWRL Tab: Step 1. Modules from Components	104

List of Figures

Figure 72 SWRL Tab: Step 2. Failure State Component Place	105
Figure 73 SWRL Tab: Step 2. Working State Component Place	106
Figure 74 SWRL Tab: Step 3. Failure Component Transition.....	107
Figure 75 SWRL Tab: Step 4. Repair Component Transition	108
Figure 76 SWRL Tab: Step 5. Component PT Arc from FSCP to RCT.....	109
Figure 77 SWRL Tab: Step 6. Component PT Arc from WSCP to FCT	110
Figure 78 SWRL Tab: Step 7. Component TP Arc from FCT to FSCP	111
Figure 79 SWRL Tab: Step 8. Component TP Arc from RCT to WSCP	112
Figure 80 Step 9. System Transitions from Assembly Activities	113
Figure 81 SWRL Tab: Step 10. System operational States.....	114
Figure 82 SWRL Tab: Step 10. System Place from System States.....	115
Figure 83 SWRL Tab: Steps 11 & 12. System Transitions	116
Figure 84 SWRL Tab: Step 13 & 14. SystemPlace	117
Figure 85 SWRL Tab: Step 15. Component To System Arc	118
Figure 86 SWRL Tab: Step 16. System to Component Arc	119

List of Tables

Table 1 Summary of main Petri Net typologies	17
Table 2 Interpretations of Petri Net properties in manufacturing	17
Table 3 Summary of the main typologies of Petri Net with possible uses	19
Table 4 Summary of the three main possible deadlock approaches	21
Table 5 MGW index values	35
Table 6 SRE index values	35
Table 7 Static input parameters.....	37
Table 8 Dynamic input parameters.....	37
Table 9 Available workers for Task_5	41
Table 10 WSA results on Task_5 (designed for an Experienced Electrical Technician)	42
Table 11 RAAS Ontology: URIs, Classes and Definitions.....	45
Table 12 PNML and PN4R Ontologies: URIs, Classes and Definitions	46
Table 13 OWL object properties of the RAAS Ontology.....	50
Table 14 Data properties of Automated Assembly System Ontology	51
Table 15 Individuals on the initial scenario	52
Table 16 Assembly Activity Duration Time	53
Table 17 Mean Time Between Failures occurrence	53
Table 18 Mean Time To Replace occurrence	53
Table 19 Two extra terms in RAASO	59
Table 20 Instances of the AASO applied to the use case	66
Table 21 Instances of the PNO applied to the use case	66
Table 22 Components and Process Variables	75
Table 23 Process Variable Deviation Cause Event [CPNHZP:001]	85
Table 24 Process Variable Deviation from Expected State [CPNHZP:002]	86
Table 25 HAZOP Unit [CPNHZP:003]	86
Table 26 List of worker's skills.....	98
Table 27 List of worker's skills.....	99
Table 28 Typical deviations for process industry	120

List of Equations

Equation 1 Matchability index for X (available) and Y (required) Worker Groups.....	34
Equation 2 Probability of producing a state sequence y given an observation sequence x	38
Equation 3 Normalization function given the feature function f_k ; and weights λ_k	38
Equation 4 Selection cost components.....	39
Equation 5 selection cost of worker E_i	39
Equation 6 Worker's capability cost	40
Equation 7 Suitability Score	42
Equation 8 Confidence interval.....	56
Equation 9 PN model simulation: Total number of cycles	59
Equation 10 Probability that a component of a specific type fails.....	60
Equation 11 System availability	61
Equation 12 Mean Time Between Failures	66
Equation 13 Failure Rate.....	66
Equation 14 Replacing Rate	66
Equation 15 HAZOP reporting time evaluation.....	89

List of Acronyms

AC	Application Case
ADACOR	ADaptive holonic CONtrol aRchitecture for distributed manufacturing systems
AMS	Automated Manufacturing Systems
API	Application Programming Interface
BFO	Basic Formal Ontology
CAD	Computer-Aided Design
CAPN	Context-Adaptive Petri Nets
CIDEM	Common Information Data Exchange Model
CMMS	Computerized Maintenance Management System
CPN	Coloured Petri Net
CRF	Conditional Random Field
CSS	Cascading Style Sheets
CTMC	Continuous-Time Markov Chain
DTMC	Discrete-Time Markov Chain
EDAM	Doctoral Program in Advanced Manufacturing
EPN	Extended Petri Net
ERP	Enterprise Resource Planning
EU	European Union
FMECA	Failure Mode, Effects, and Criticality Analysis
FPN	Fuzzy Petri Net
GRDDL	Gleaning Resource Descriptions from Dialects of Languages
GSPN	Generalized Stochastic Petri Net
HAZOP	HAzard and Operability
HMM	Hidden Markov Models
HR	Human Resource
HRM	Human Resource Management
HTTP	HyperText Transfer Protocol
IAO	Information Artifact Ontology
ICT	Information and Communications Technology
IOF	Industrial Ontologies Foundry
M&S	Modelling and Simulation
MASON	MAanufacturing's Semantics Ontology
MPM	Manufacturing Process Management
MTBF	Mean Time Between Failures
MTTF	Mean Time To Failure
MTTR	Mean Time To Repair
MWG	Matchability index for Worker groups
OBO	Open Biological and Biomedical Ontology
OMG	Object Management Group

OSF	Open Semantic Framework
OTKM	On-To-Knowledge methodology
OWL	Web Ontology Language
P&ID	Piping and instrumentation diagram
PLM	Product Lifecycle Management
PN	Petri Net
PN	Classical Petri Net
PN4R	Petri Net for Reliability Assessment
PN4RO	Petri Net for Reliability Assessment Ontology
PNML	Petri Net Markup Language
POWDER	Protocol for Web Description Resources
QVT	Query/View/Transformation
R2RML	Relational Databases to RDF Mapping Language
RAASO	Reliability Assessment for Assembly Stations Ontology
RDF	Resource Description Framework
RDFa	Resource Description Framework in Attributes
RDFS	RDF Schema
RIF	Rule Interchange Format
RO	Relation Ontology
RQ	Research Question
SCADA	Supervisory Control and Data Acquisition
SKOS	Simple Knowledge Organization System
SOAP	Simple Object Access Protocol
SPARQL	SPARQL Protocol and RDF Query Language
SPN	Stochastic Petri Net
SQL	Structured Query Language
SRE	Suitability index for Required Experience levels
SS	Suitability Score
SVG	Scalable Vector Graphics
SWRL	Semantic Web Rule Language
TPN	Timed Petri Net
UML	Unified Modeling Language
W3C	World Wide Web Consortium
WOFF	Web Open Font Format
WSA	Worker Suitability Analysis
XML	eXtensible Markup Language
XSD	XML Schema Definition
XSLT	eXtensible Stylesheet Language Transformations

Chapter 1 Introduction

Throughout history, manufacturing has held an important role in society, driving innovation and human progress forward, however, the manufacturing domain is rapidly changing and soon it may look very different and virtually unrecognisable compared to today. History has taught us how fast and radical such change can be.

Nowadays, the industrial domain is experiencing the so-called 4th industrial revolution (Industry 4.0), which is fundamentally changing the way we live, work and relate to one another. While previous industrial revolutions freed human beings from dependence on animal power, made mass production possible and provided digital access to billions of people, this industrial revolution is different in essence (Figure 1). It is characterized by a range of new technologies that are fusing the physical, digital and biological worlds, impacting all disciplines, economies and industries, and even challenging ideas about what it means to be human [1].

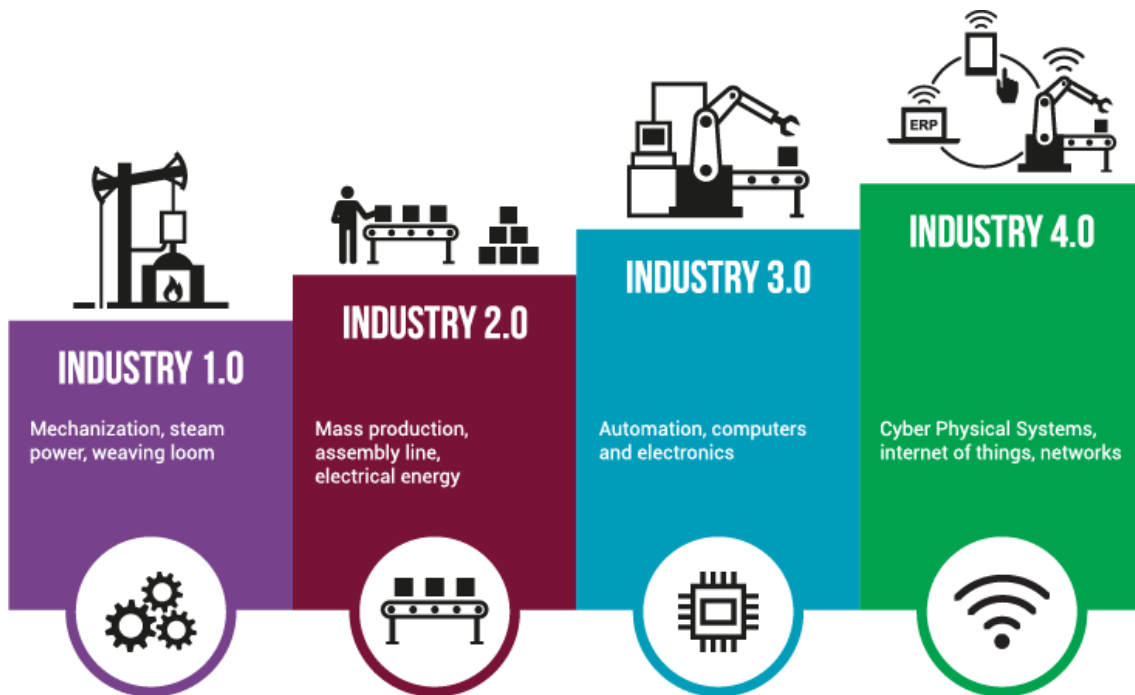


Figure 1 From the first industrial revolution to Industry 4.0¹

According to a recent study by McKinsey, Industry 4.0 consists of a confluence of disruptive digital technologies set to change the manufacturing sector beyond recognition and which are driven by the astonishing rise in data volumes, computational power, and connectivity; by the emergence of advanced analytics and business intelligence capabilities; by new forms of human-machine interaction and improvements in the transfer of digital instructions to the physical world. This is why, to capture emerging opportunities and keep pace with the rapidly-advancing technological frontier, industrial players need to build foundations for the organization's digital transformation by developing digital capabilities, enabling collaboration in the ecosystem, and managing data as a valuable asset [2].

¹ Found on <http://gmformazione.it/competenze-industry-4-0-cosi-le-pmi-le-adegua-no-al-mercato/> [Last access 17 January 2019]

All over the world, the interest in Industry 4.0 is rising exponentially month by month. For instance, the share of Google searches for “Industry 4.0” relative to total searches has shot up in the last 4 years, giving us an idea of global interest in the phenomenon but also the enormous long-term potential waiting to be unearthed and the far-reaching implications of Industry 4.0 initiatives for production systems.

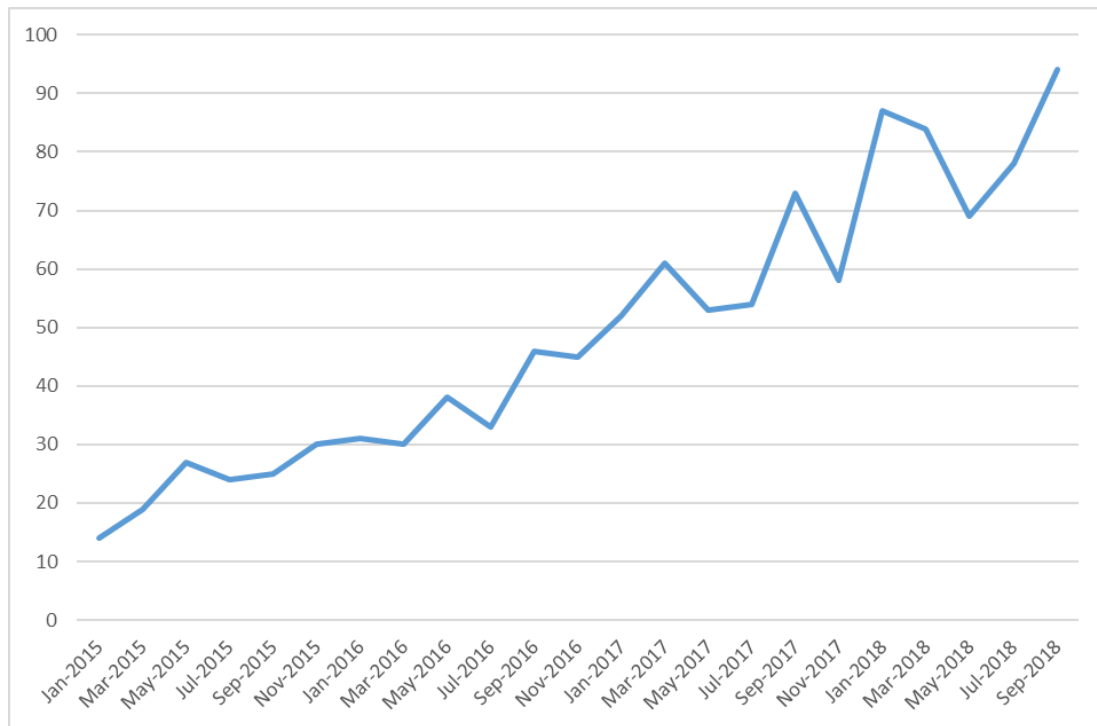


Figure 2 Google Trends graph for “Industry 4.0” in the world

In light of all this, two points can be made. On the one hand, the ever-growing granularity of manufacturing information, the increasing amount of available knowledge, and the use of Semantic Web techniques adhering meaning to data represent three crucial factors accelerating the growth of complexity of industrial systems. On the other hand, the plethora of existing standards and modelling languages together with the need for re-usable, shareable, and formal knowledge representation are catalysing the growth complexity in ICT solutions for Industry 4.0 [3] and the factories of the future.

Such a growth in complexity of industrial systems and ICT solutions inevitably has significant implications for the design of Product Lifecycle Management (PLM) oriented software, and thus impacts in the way Computer-Aided Design (CAD), Manufacturing Process Management (MPM), and Enterprise Resource Planning (ERP) systems are conceived and interoperate one another to manage each aspect of a product development from design to production. In this framework, the Manufacturing Process Management - which can be reasonably considered as a link between CAD and ERP systems - should be exploited to accelerate new product introductions, shorten time-to-volume, optimize production executions, decrease operating costs, and ensure overall product and process quality. In particular, as manufacturing processes become increasingly digital, the notion of digital twin becomes increasingly topical: a near-real-time digital image of a physical object or process that helps optimize business performance [4], or even a complete digital footprint of processes or products which enables companies to detect physical issues sooner and predict outcomes more accurately.

That said, the need to allow industrial systems to meet the ever-changing needs and conditions due to technological evolution, together with the need to exploit digital manufacturing processes and perform domain-specific analysis, lead to research into and development of ad hoc solutions. These do not necessarily introduce new modelling paradigms but are rather based on the integration of the existing ones with the aforementioned enabling tools, thus paving the way towards the development of a new analytical framework.

Chapter 2 Research Method

The research method refers to the overall strategy chosen to integrate the different elements of a research study in a coherent and logical way, thereby, ensuring the research problem to be effectively addressed. Following, the current study is described in terms of Purpose (Section 2.1), Research Questions (Section 2.2), and Strategy (Section 2.3), therefore, defining the methodological point of departure, the boundaries and the level of detail of this research work. The thesis structure is provided in Section 2.4.

2.1 Research Purpose

There are many reasons why research is done, however, answering the question “what is the purpose of it?” is not a trivial task. The following research work starts with these two fundamental questions:

- *Why do we have the need to develop an entire group of sciences to understand advanced manufacturing processes?*
- *What is the role of ICT while acquiring and using knowledge regarding advanced manufacturing processes?*

The Doctoral Program in Advanced Manufacturing (EDAM), which I have joined in 2016, addresses the science and engineering of advanced manufacturing processes. Advanced manufacturing is diverse, multiscale and multidisciplinary. It requires fundamental knowledge in materials, solid and fluid mechanics, surface and interface science, multiscale modelling, process and system engineering. There is a concrete and reasonable need for understanding advanced manufacturing processes because manufacturing engineering knowledge is rapidly changing and we need to match up emerging needs and technologies. In this context, ICT solutions should meet emerging manufacturing needs and leverage on cutting-edge technologies in order to facilitate a rapid understanding of manufacturing processes, enhance existing systems with decision quality, knowledge sharing, inter-organizational links, and the contribute to the resolution of the implicit conflict between sustainability and economic growth [5].

Therefore, the purpose of advanced manufacturing research is to explore, to describe and to explain how and why emerging engineering technologies can bring a novel contribution in this field whereas - from a business perspective – MPM-oriented technologies should be exploited to accelerate new product introductions, shorten time-to-volume, optimize production executions, decrease operating costs, and ensure overall product and process quality. When it comes to MPM, Modelling and Simulation (M&S) represent two vital processes on which researchers should particularly draw their attention. These two aspects are becoming even more relevant nowadays since digital manufacturing represents not only an enabling technology but also an added value for the factories of the future [6]. However, there is often a lack of interoperability both across the PLM phases [7] as well as between modelling and analysis phases in MPM applications [8]. This can be addressed by semantic technologies, which may also bring up more added value by empowering the framework with the capability of giving meaning to the knowledge that characterizes this middle element of the PLM [9]. Indeed, the exploitation of semantic technologies becomes relevant when it comes to reengineering business (manufacturing) processes [10], allowing manufacturing systems interoperability [11], extracting implicit and explicit knowledge of an enterprise [12], aligning different modelling languages [13], [14], and supporting the analysis and validation of business process models [15]. However, formal analysis and simulation of a manufacturing process require the application of techniques built upon strong mathematical foundations. In this regard, Petri Net (PN) is a well-established modelling language for manufacturing systems, and, furthermore, PN-based models provide a robust math-based notation. For this reason, it can be chosen, from among a number of reasonable solutions, as the modelling language to develop, verify and simulate the manufacturing system analysed in this research study [16]–[18].

To sum up, the ever-growing complexity of manufacturing information, the increasing amount of available knowledge, and the need to allow industrial systems to meet the ever-changing needs and conditions due to technological evolution lead to explore alternative approaches that can support both design and simulation of digital twins, which are considered nowadays as a key asset in any manufacturing company. The present research study, therefore, aims to leverage the best features of ontological models and digital twins towards the formalization of new practical approaches for modelling and simulation of semantically-enriched manufacturing system models.

2.2 Research Questions

As introduced above, the investigation of alternative theoretical approaches aiming at supporting both modelling and simulation of (manufacturing system) digital twins leads to face many open issues, which seek solution, hence, improvement. The research objectives of this dissertation will be hereby given in the form of research questions:

- RQ1. How can we **semantically enrich** manufacturing system models and exploit **context-awareness** for process analysis?
- RQ2. How can semantics be used to foster **reusability** and **interoperability** of manufacturing systems models?
- RQ3. How does the analysis of semantically-enriched manufacturing systems **impact on modelling and simulation** applications?

Six research drivers have been therefore identified and explained in detail hereafter:

- **Semantic Enrichment** is about giving data value through assigning meaning to it, moving away from seeing it as something static and linear. Instead approaching it as something dynamic and flexible which can be used to create multiple outputs, therefore, enhancing its value and enabling new uses.
- **Context Awareness**, here, is meant to be the ability of a system to capture and use information about the context of operations, thus, reaching a reasonable level of reasoning on context conditions
- **Reusability** is the use of existing assets in some form within the industrial modelling process. These assets may include industrial process knowledge, process data, system modelling primitives, modelling language selection, translation, and simulation mechanisms.
- **Semantic Interoperability** is the ability of computer systems to exchange data with unambiguous, shared meaning. This can be accomplished by adding data about the metadata, linking each data element to a controlled, shared vocabulary.
- **Impact on Modelling** is related to the selection, adaptation, and exploitation of a representational language (or mathematical formalism) to achieve the research purpose.
- **Impact on Simulation** evaluated through the specific application studies, it involves different parameters for quantitative (and qualitative) assessment of the application results.

2.3 Research Strategy

Generally speaking, we may distinguish two different types of research: Explanatory and Exploratory [19]. As an Exploratory Research, the study here presented aims to lay the initial groundwork for future research. Starting from the observations above - the ever-growing complexity of manufacturing information, the increasing amount of knowledge, and the extensive use of semantic web techniques adhering meaning to data - this dissertation provides a new angle and new ways of looking at things. Either from a theoretical and from a practical perspective, paving the way towards the development of a new analytical framework for modelling and simulating manufacturing systems.

Based on the nature of the research questions and objectives, the extent of existing knowledge, and the available amount of time and other resources, the research strategy adopted in this work is a so-called Action research, whose strengths are the focus on understanding the change, the recognition that time needs to be devoted to diagnosing, planning, taking action and evaluating, and the involvement of practitioners throughout the process [20].

Finally, the context and purpose of this work reside within the research questions (RQs) – shaped on the initial hypothetical ideas on which this exploratory research work is funded –, linked to one or two research drivers, which in turn open the way to an initial action planning that is cyclically evaluated (along the first three years of this research work) in terms of requirements and technology maturity and allows further – and continuous – refinement of Research Questions, Research Drivers and Action Task (Figures 4).

2.3.1 Research Contribution

The contribution to theoretical knowledge, here, can be summarized in two main conclusions, as follows:

- In the past two decades, there has been a big effort towards the promotion, use and development of ontological models aiming to support the manufacturing domain. However, ontologies can be used to provide a better understanding of the represented domain (or portion of reality) in a way which is both human and machine understandable. By answering the question above, an investigation on how to empower manufacturing system ontology with quantitative/qualitative analysis capabilities is put into practice, hence, designing a conceptual framework with the mission of supporting modelling and simulation of manufacturing systems through an ontology-based solution.
- Besides the obvious need of matching engineering requirements based on the specific application and use cases, when it comes to ontologies, it is important to take into consideration the philosophical principles on which such technologies have been built upon. Due to likely inconsistencies of the actual approaches and solutions from an ontological perspective, there is a need a formal ontology that can and should support an ontology-based solution for creating semantically-enriched digital twins.

2.3.2 Contribution to Practice

Broadly speaking, the Contribution to Practice becomes clear, above all, in Chapter 6, which describes three Industrial Application Cases (ACs) against which this research work has been benchmarked as well as validated through real data and information collected in three manufacturing sub-domains:

AC 1 Human Resource Management through Semantically-enriched Data

AC 2 Reliability Assessment of a Semi-Automated Assembly Station

AC 3 Automated HAZOP Analysis of a Chemical Process Plant

Even though these application cases already show some significant practical implications in the abovementioned application domains, I believe these can be proven to be more profound on a larger scale.

Besides this, the exploitation of a - workflow and/or industrial system - modelling language based on strong math foundations (i.e. Petri Net) plays a relevant role as it enables the evaluation of the efficiency of the modelled systems as well as its effectiveness, however, the so-called semantic enrichment of manufacturing system models represents the pivotal and most critical element of this research work, and this leans on the early stage adoption of semantic technologies in manufacturing applications.

In the past two decades, significant budgets have been invested in the development and employment of semantic technologies. To date, there is some uptake of the technology, however, the mainstream market is still not reached. This is partly because of a lack of understanding by industries in what situations semantic technologies can add value to their business. Digitization which implies the use of semantic technologies does not just allow companies to get more out of their existing production processes. It is also changing the way manufacturing is done. This is the reason for introducing a new framework that is aimed at enabling industries to more easily relate semantic technologies to their business.

2.4 Thesis Structure Outline

Chapter 2 has described the Research Method as a multi-layered and structured activity that comprises the definition of purpose, method, questions and strategy. Then, Chapter 3 and Chapter 4 introduce namely the background in Ontology Engineering methods and technologies and the background in Modelling and Simulation methods and technologies for manufacturing systems with Petri Net language. 0 provides a detailed analysis of the development and use of ontologies for diverse aspects of the current research, eventually resulting in the design of a conceptual framework which exploits their best features. Finally, Chapter 6 presents three Application Cases on which the diverse clusters of the presented framework are tested and validated.

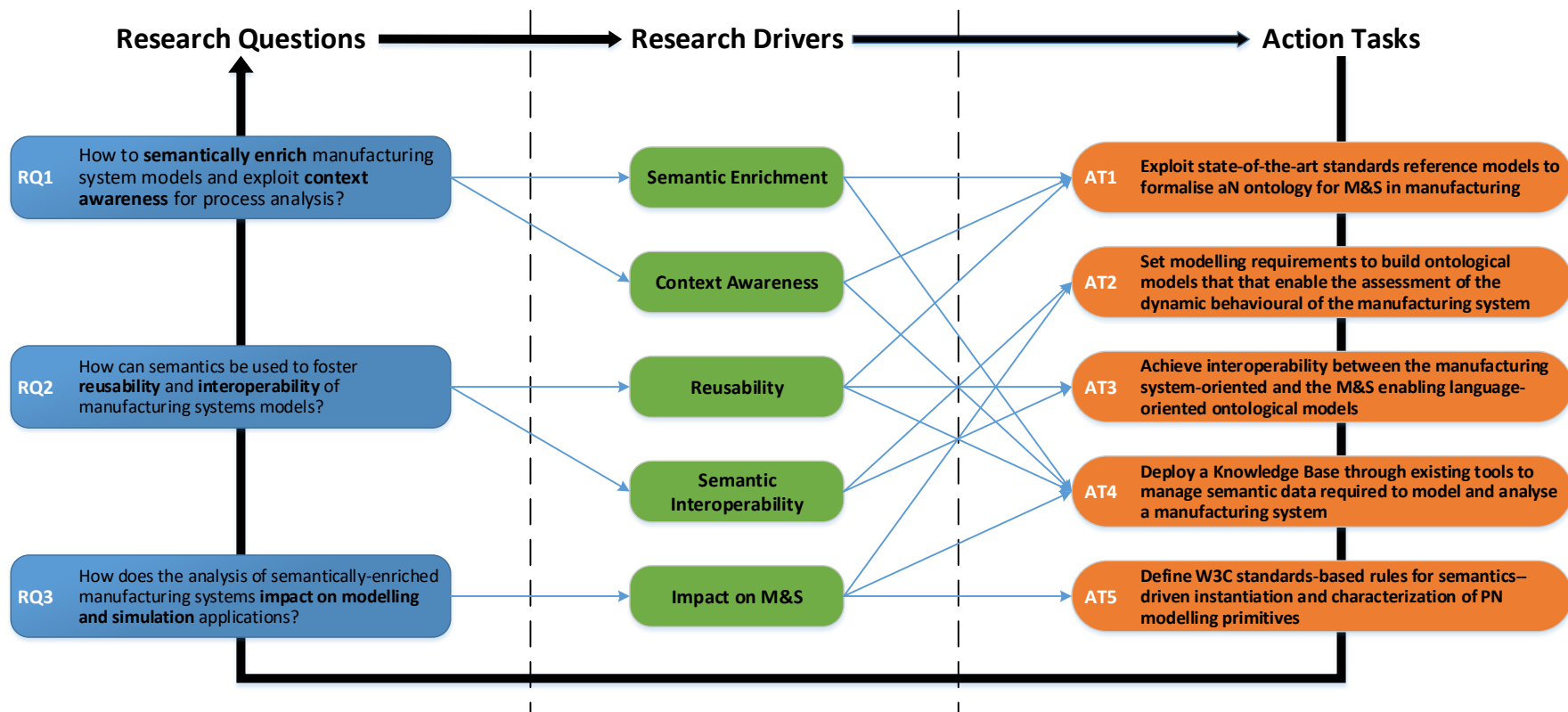


Figure 3 From RQs to Action Tasks

Chapter 3 Background in Ontology engineering

In philosophical contexts, “ontology” has traditionally been defined as the theory of what exists (or of “being qua being”) [5], in other words, the study of the kinds of entities in reality and of the relationships that these entities bear to one another. In literature, it is possible to find many definitions of “ontology”. Although such definitions have changed and evolved over the years [6], [7], Studer and colleagues [8] provide one of the most well-known, i.e. “An ontology is a formal, explicit specification of a shared conceptualization”. *Conceptualization* refers to an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon. *Explicit* means that the type of concepts used and the constraints on their use are explicitly defined. *Formal* refers to the fact that the ontology should be machine-readable. *Shared* reflects the notion that an ontology captures consensual knowledge, i.e. it is not private for some individual, but accepted by a group. In recent times, however, as the term “ontology” has become widely adopted in information and communication technologies, it has taken on a new meaning which refers to a standardized representational framework providing a set of terms for the consistent description of data and information across disciplinary and research community boundaries [9].

In this research work, we adopt the definitions provided by [9], which emphasizes the need of promoting greater consistency in the description of data, hence, resulting in a graph-like representation, in which terms serve as nodes in the graph and ontological relations serve as edges. Following, we recall a few definitions that help the reader better understand the topics dealt with in the next sections.

Ontology = A representational artefact, comprising a taxonomy as proper part, whose representations are intended to designate some combination of universals, defined classes, and certain relations between them.

Representational artefact = An artefact whose purpose is one of representation.

Taxonomy = A hierarchy consisting of terms denoting types linked by subtype relations.

By “types” (or universals or classes) we mean the entities in the world referred to by the nodes (appearing here as boxes) in a hierarchy; in the case of Figure 4, motorcycles models.

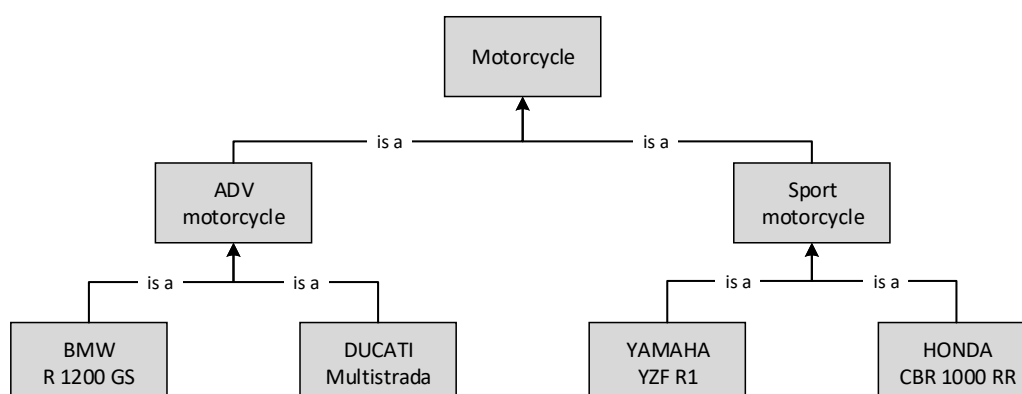


Figure 4 Fragment of a simple taxonomy of motorcycles

3.1 Methodologies for Ontology design, development, and use

Starting from the late 90s, some research groups have been proposing a series of steps and methodologies for designing ontologies. The first methodological outlines were proposed in 1995 and later on refined in [10] and [11]. In 1996, Bernaras and colleagues [12] presented a method used to build an ontology in the domain of electrical networks as part of the Esprit KACTUS project. METHONTOLOGY [13] appeared at the same time, but was extended later in [14]–[16]. In 1997, that is, one year later, a methodology was proposed for building ontologies based on the SENSUS approach [17]. With this regard, Fernandez López M. published in 1999 a detailed analysis of the most those methodologies [18] at that time. He claims that the work of Bernaras et al. - set within the Esprit **KACTUS** project - aims to investigate the feasibility of knowledge re-use in complex technical systems and the role of ontologies to support it [19]. This approach to developing ontologies is conditioned by applications development. So, every time an application is built, the ontology that represents the knowledge required for the application is built. This ontology can be developed by reusing others and can also be integrated into the ontologies of later applications. The **SENSUS** Ontology, instead, is an ontology for use in natural language processing and was developed at the ISI (Information Sciences Institute – USC University of Southern California) natural language group to provide a broad-based conceptual structure for developing machine translators [20], [21]. When an ontology is to be built in a particular domain according to the SENSUS approach, 5 (five) steps are taken, from the selection of the domain terms, their link to SENSUS and final refinement of the ontological model. For those two methodologies, the main shortcomings can be summarized as follows. The SENSUS-based methodology does not deal with the life cycle of ontologies or with features such as formatting and IRI conventions. In addition, this methodology has hardly been used to build actual ontologies or ontology-based applications, which is a crucial point when it comes to fostering model reusability and shareability [12].

Till the early 2010s, mainly due to the fact that Ontology Engineering was still a relatively immature discipline, each work group ended up employing its own methodology. In the very last few years, however, METHONTOLOGY², On-To-Knowledge³, DILIGENT⁴, and NeOn⁵ Methodology were up to 2009 the most referred methodologies for building ontologies. The latter suggests pathways and activities for a variety of scenarios, instead of prescribing a rigid workflow, whereas the first three methodologies mainly include guidelines for single ontology construction ranging from ontology specification to ontology implementation and they are mainly targeted to ontology researchers. In particular, **METHONTOLOGY** was developed within the Laboratory of Artificial Intelligence at the Polytechnic University of Madrid. The METHONTOLOGY framework [14]–[16], [22] enables the construction of ontologies at the knowledge level. It includes (a) the identification of the Ontology Development Process (ODP) which tasks should be performed when building ontologies; (b) a life cycle based on evolving prototypes; and (c) some techniques to carry out management, development-oriented, and support activities. In addition, METHONTOLOGY includes a list of activities to be carried out during ontology re-use and re-engineering processes, but it does not provide detailed guidelines for such activities, nor does it consider different levels of granularity during the re-use of ontological resources (e.g. modules or statements). Moreover, METHONTOLOGY neither considers the re-use and re-engineering of non-ontological resources nor the re-use of ODP.

The **On-To-Knowledge** methodology (OTKM) [23] proposes to build ontologies taking into account how these are going to be used in knowledge management applications. The processes proposed by this methodology are the following: feasibility study, kickoff, where ontology requirements are identified, refinement, where a mature and application-oriented ontology is produced, evaluation, and maintenance. With respect to the re-use of knowledge resources, in the kickoff process, it is mentioned that developers should look for potentially reusable ontologies. However, this methodology does not provide detailed guidelines for identifying such ontologies nor for reusing them. Besides, the methodology neither explicitly mentions guidelines for the re-use and re-engineering of non-ontological resources, nor for the re-use of ontology design patterns.

The **DILIGENT** methodology [24] is intended to support domain experts in a distributed setting in order to engineer and evolve ontologies. This methodology is focused on collaborative and distributed ontology engineering. Its ontology development process includes the following five activities: building, local adaptation, analysis, revision, and local update. With regard to the re-use of knowledge resources, the methodology does not include guidelines for the re-use and re-engineering of existing knowledge resources.

² METHONTOLOGY - <http://semanticweb.org/wiki/METHONTOLOGY> [Last Access 7 January 2018]

³ On-To-Knowledge - <http://www.ontotext.com/research/otk> [Last Access 7 January 2018]

⁴ DILIGENT - <http://semanticweb.org/wiki/DILIGENT> [Last Access 7 January 2018]

⁵ NeOn - <http://www.neon-project.org/> [Last Access 7 January 2018]

The **NeOn Methodology** [25] for building ontology networks is a scenario-based methodology that supports a knowledge re-use approach, as well as collaborative aspects of ontology development and dynamic evolution of ontology networks in distributed environments. A network of ontologies is a collection of ontologies related together via a variety of relationships such as alignment, modularization, version and dependency. The key assets of the NeOn Methodology are:

- A set of nine scenarios for building ontologies and ontology networks, emphasizing on the re-use of ontological and non-ontological resources, the reengineering and merging, and taking into account collaboration and dynamism.
- The NeOn Glossary of Processes and Activities, which identifies and defines the processes and activities carried out when ontology networks are collaboratively built by teams.
- Methodological guidelines for different processes and activities of the ontology network development process, such as the re-use and reengineering of ontological and non-ontological resources, the ontology requirements specification, the ontology localization, the scheduling, etc. All processes and activities are described with (a) a filling card, (b) a workflow, and (c) examples.

Despite the maturity of such methodologies, it has been proven that the interoperability is still a continuing challenge for information systems and those systems that rely heavily on the use, or production of, information. Moreover, industrial companies have remained hesitant to use ontologies. This is due both to 1) a continued lack of knowledge of the advantages ontologies can bring, and 2) to the persistent immaturity of the field. For, in industrial domains, ontologies have been typically segmented, not standardized, and inconsistent in quality. This poses a series of risks in industrial contexts. At the same time, however, the problems created by failures of interoperability of the systems being developed in areas such as digital manufacturing continue to accumulate. To promote interoperability, ontologies need a solid base of definitions with cross industrial domains support. This support can come about through a community-based effort toward building this pre-competitive technology. In 2017 the initiative called **Industrial Ontologies Foundry (IOF)** was taken with the mission of creating a suite of high quality core and open ontologies covering the entire domain of digital manufacturing⁶. The goals of the IOF include:

- Providing principles and best practices by which quality ontologies can be developed that will support interoperability for industrial domains,
- Creating a suite of open and principles-based ontologies from which other sub-domain or application ontologies can be derived in a modular fashion, remaining 'generic' (i.e., non-proprietary, non-implementation specific) so they can be re-used in any number of industrial domains or manufacturing specializations,
- Instituting a governance mechanism to maintain and promulgate the goals and principles,
- Providing an organizational framework and governance processes that ensure conformance to principles and best practices for the development, sharing, maintenance, evolution, and documentation of IOF ontologies.

The intent of these reference ontologies is to allow extensions to be progressive to more specific or constrained sub-domains (e.g., particular industry 'verticals' or applications). To meet this intent, the IOF ontologies are expected to have an architecture that starts from alignment with a domain neutral ontology, also referred to as an Upper Ontology or Foundational Ontology, from which subsequent IOF ontologies can be developed (newly or adapted from existing ones) that are ontologically consistent, coherent, and modular allowing for reusability.

Building from the Basic Formal Ontology (BFO) the first 'layer' of IOF ontologies will be a collection of Reference Ontologies covering notions and relations including time, units of measure, logistics, information, geospatial, etc. The next layer will be a collection of Domain Specific Ontologies covering notions specific to industrial domains. Extensions following the domain specific reference ontologies, Application Ontologies, will address more focused sub-domains of industrial and manufacturing.

BFO (today also referred to as BFO-ISO) is an upper level ontology that is designed for use in supporting information retrieval, analysis and integration in scientific and other domains. BFO is a genuine (or domain neutral) upper ontology. This means that BFO does not

⁶ Industrial Ontologies Foundry - <https://www.industrialontologies.org/> [Last Access 24 March 2018]

contain terms which would be properly covered by domain specific ontologies. BFO has been used intensively in many ontology development efforts, serving as top-level in the OBO Foundry (<http://www.obofoundry.org/>), the Common Core Ontologies (<https://github.com/CommonCoreOntology/CommonCoreOntologies>), and a number of other hub-and-spokes ontology suites. BFO has existed from the beginning in an OWL formalization, however, there is also a CLIF/FOL version (<https://github.com/BFO-ontology/BFO>).

The dichotomy between *continuant* and *occurrent* forms the central organizing axis of the BFO ontology. A continuant is an entity that persists, endures, or continues to exist through time while maintaining its identity. Continuants can preserve their identity even while gaining and losing material parts. If *a* is a continuant, then there is some temporal interval during which *a* exists [26]. The occurrent portion of BFO represents entities that occur, happen, unfold, or develop in time. In commonsensical terms, these entities are occurrences or happenings or the processes of change. A BFO: occurrent is, more precisely, either an entity that unfolds itself in time, or it is the instantaneous boundary of such an entity (for example, a beginning or an ending) along what we can think of as the time axis, or it is a temporal or spatiotemporal region that such an entity occupies [27].

BFO does not claim to be a complete coverage of all entities. It seeks only to provide coverage of those entities studied by empirical science together with those entities which affect or are involved in human activities, such as data processing and planning. This coverage is sufficiently broad to provide assistance to those engaged in building domain ontologies for purposes of data annotation [28] or representation and reasoning in science, medicine, and many areas of administration and commerce[29].

In this work, BFO has been selected as the Foundational Ontology from which the domain specific ontologies addressing the issues and facing the challenges arising from the three application cases presented in Chapter 6 were developed. This ensures the consistency of these ontologies as well as their reusability in the domain.

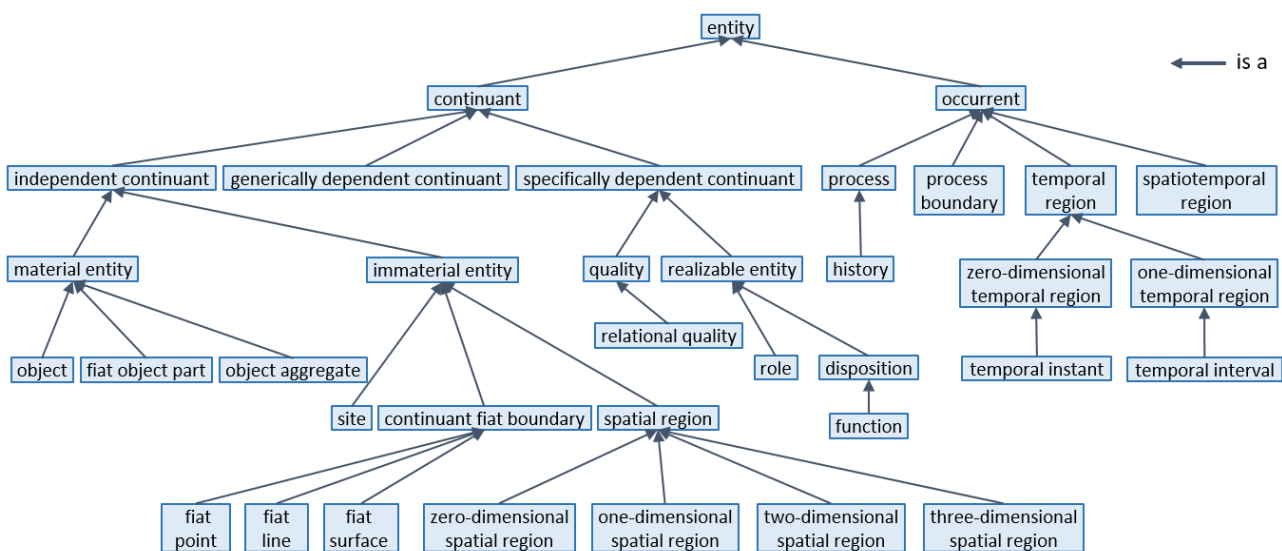


Figure 5 Basic Formal Ontology (BFO) hierarchy

W3C standards define an Open Web Platform for application development that has the potential to enable developers to build rich interactive experiences, powered by vast data stores, which are available on any device. Although the boundaries of the platform continue to evolve, industry leaders speak nearly in unison about how HTML5 will be the cornerstone for this platform. But the full strength of the platform relies on many more technologies that W3C and its partners are creating, including CSS, SVG, WOFF, the Semantic Web stack, XML, and a variety of APIs.

W3C develops these technical specifications and guidelines through a process designed to maximize consensus about the content of a technical report, to ensure high technical and editorial quality, and to earn endorsement by W3C and the broader community⁷.

⁷ W3C Standards - <http://www.w3.org/standards/> [Last Access 7 January 2018]

In addition to the classic “Web of documents” W3C is helping to build a technology stack to support a “Web of data,” the sort of data you find in databases. The ultimate goal of the Web of data is to enable computers to do more useful work and to develop systems that can support trusted interactions over the network. The term “Semantic Web” refers to W3C’s vision of the Web of linked data. Semantic Web technologies enable people to create data stores on the Web, build vocabularies, and write rules for handling data. Linked Data⁸ are empowered by technologies such as RDF, SPARQL, OWL, and SKOS.

The Semantic Web is a Web of Data — of dates and titles and part numbers and chemical properties and any other data one might conceive of. The collection of Semantic Web technologies (RDF, OWL, SKOS, SPARQL, etc.) provides an environment where applications can query that data, draw inferences using vocabularies, and the like.

However, to make the Web of Data a reality, it is important to have the huge amount of data on the Web available in a standard format, reachable and manageable by Semantic Web tools. Furthermore, not only does the Semantic Web need access to data, but relationships among data should be made available too, to create a *Web* of Data (as opposed to a sheer collection of datasets). This collection of interrelated datasets on the Web can also be referred to as Linked Data.

To achieve and create Linked Data, technologies should be available for a common format (RDF), to make either conversion or on-the-fly access to existing databases (relational, XML, HTML, etc.). It is also important to be able to setup query endpoints to access that data more conveniently. W3C provides a palette of technologies (RDF, GRDDL, POWDER, RDFa, the recent R2RML, RIF, and SPARQL) to get access to the data.

Linked Data lies at the heart of what Semantic Web is all about: large scale integration of, and reasoning on, data on the Web. Almost all applications listed in, say collection of Semantic Web Case Studies and Use Cases are essentially based on the accessibility of, and integration of Linked Data at various level of complexities.

On the Semantic Web, vocabularies define the concepts and relationships (also referred to as “terms”) used to describe and represent an area of concern. Vocabularies are used to classify the terms that can be used in a particular application, characterize possible relationships, and define possible constraints on using those terms. In practice, vocabularies can be very complex (with several thousands of terms) or very simple (describing one or two concepts only).

There is no clear division between what is referred to as “vocabularies” and “ontologies”. The trend is to use the word “ontology” for more complex, and quite possibly formal collection of terms, whereas “vocabulary” is used when such strict formalism is not necessarily used or only in a very loose sense. Vocabularies are the basic building blocks for inference techniques on the Semantic Web.

The role of vocabularies on the Semantic Web is to help data integration when, for example, ambiguities may exist on the terms used in the different data sets, or when a bit of extra knowledge may lead to the discovery of new relationships. Consider, for example, the application of ontologies in the field of health care. Medical professionals use them to represent knowledge about symptoms, diseases, and treatments. Pharmaceutical companies use them to represent information about drugs, dosages, and allergies. Combining this knowledge from the medical and pharmaceutical communities with patient data enables a whole range of intelligent applications such as decision support tools that search for possible treatments; systems that monitor drug efficacy and possible side effects; and tools that support epidemiological research.

Another type of example is to use vocabularies to organize knowledge. Libraries, museums, newspapers, government portals, enterprises, social networking applications, and other communities that manage large collections of books, historical artefacts, news reports, business glossaries, blog entries, and other items can now use vocabularies, using standard formalisms, to leverage the power of linked data.

The complexity of the vocabulary that is used depends on the application. In some cases, it might be decided not to use even small vocabularies, and rely on the logic of the application program. In other cases, it might be chosen to use very simple vocabularies, and let a general Semantic Web environment use that extra information to make the identification of the terms. Finally, some applications may need more complex ontologies with complex reasoning procedures. It all depends on the requirements and goals of the applications.

⁸ Linked Data - Connect Distributed Data across the Web - <http://linkeddata.org/> [Last Access 7 January 2018]

To satisfy these different needs, W3C offers a large palette of techniques to describe and define different forms of vocabularies in a standard format. These include RDF and RDF Schemas, Simple Knowledge Organization System (SKOS), Web Ontology Language (OWL), and the Rule Interchange Format (RIF). The choice among these different technologies depends on the complexity and rigor required by a specific application.

In the framework of the Semantic Web, the meaning of “query” is related to technologies and protocols that allows information retrieval. The Resource Description Framework (RDF) provides the foundation for publishing and linking data. Mechanisms such as RDFa, GRDDL, and so on, can then be used to expose what is stored in databases, or to make it available as RDF files (triplets).

The SPARQL has been designed to send queries and receive results, e.g. through HTTP or SOAP, within the Semantic Web, which is typically represented using RDF as a data format. This query language is based on (triples) patterns that are similar to RDF triples, and the results of a SPARQL query will be the resources for all triples that match those patterns. Thus, it provides a powerful tool that allows extracting complex information (i.e., existing resource references and their relationships) and present them in a different friendly format (i.e. tables).

3.2 Leading Tools for Ontology editing

The focus is given on the new generation of ontology engineering environments, particularly on Protégé described hereafter. They have extensible, component-based architectures, where new modules can easily be added to provide more functionality to the environment.

Protégé⁹ is an open platform for ontology modelling and knowledge acquisition (Figure 6). It is an open source, standalone application (also available on-line through Web Protégé), with an extensible architecture. The core of this environment is the ontology editor, and it holds a library of modules that can be plugged, called plug-ins, to add more functions to the environment.

The main Protégé functions are to:

- Load and save OWL and RDF ontologies;
- Edit and visualize classes, properties, and SWRL (Semantic Web Rule Language) rules;
- Define logical class characteristics as OWL expressions;
- Execute reasoners such as description logic classifiers, and edit OWL individuals for Semantic Web markup.

Protégé is available in different versions, each including different plug-ins and features. A quick overview of the stepwise enhancements introduced by the last two versions are presented below:

- Protégé version 4.3 supports for editing multiple OWL2 Ontologies and RDF. Moreover, it allows in-memory connection to Pellet, HermiT and FaCT++ reasoners.
- Protégé version 5.0 offers enhancements and improvements in search, annotation viewing & editing, hierarchy viewing, ontology loading & saving, accessibility, logging and performance, as well as general user interface improvements. It also contains two new bundled plugins, "Cellfie" and the "SWRL Tab". Finally, Protégé 5.0 uses the OWL API version 4.2.5 and Java 8.

Open Semantic Framework (OSF)¹⁰ is an integrated software stack using semantic technologies for knowledge management, which includes an ontology editor. In particular, the OSF Ontology Tool provides a user-friendly environment provided by OSF through Drupal to create, modify, annotate or delete ontologies.

OSF for Drupal provides a set of already uploaded ontology sources that can be used towards the development of a custom semantic environment. The organization of the OSF Ontology application presents all the currently available and active ontologies listed in a specific panel (Figure 7). These are organized into three categories:

⁹ Protégé software - <http://protege.stanford.edu> [Last Access 7 January 2018]

¹⁰ Open Semantic Framework (OSF) software stack <http://opensemanticframework.org/> [Last Access 7 January 2018]

- Local Ontologies - these are the domain ontologies governing the use of the user's specific portal. They generally contain the "world view" and content organization specific to the user's problem or domain. Most of the editing and modifications will occur in this section. This is also the basis for the screen examples below;
- Reference Ontologies - these, too, are content ontologies, but generally, represent external ontologies that can be re-used in the user's specific portal for interoperability or sharing purposes;
- Administrative Ontologies - these are internal ontologies that guide and govern the use of widgets, permissions, components and other administrative functions of the user's site. Most site administrators will have little or no reason to make any changes to ontologies in this category.

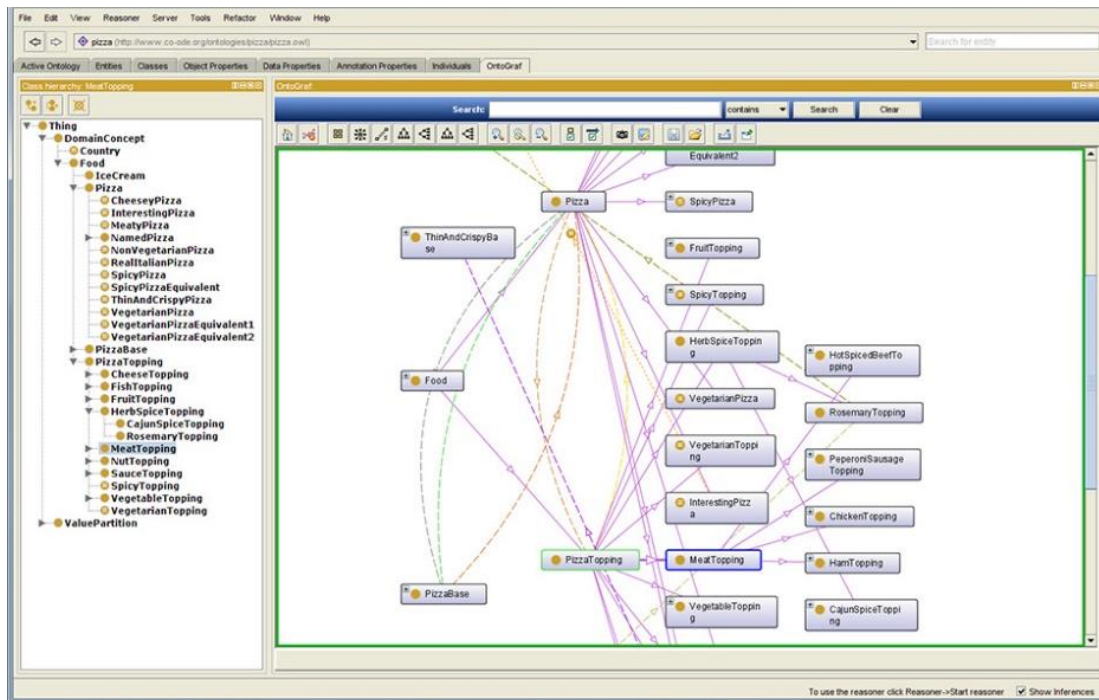


Figure 6 Protégé - Ontograf Tab

Figure 7 OSF - Ontology properties Tab

The **Anzo**¹¹ software suite bundles a Semantic Web middleware platform and database with easy-to-use-tools that allow end users to convert existing data to Semantic Web data and to visualize, analyse, and act on the data (Figure 8). It includes the Anzo on the Web dashboard, reporting, and faceted browsing tool and the Anzo for Microsoft Excel component. It also includes such staples as an RDF database, a straightforward ontology editor, two rules engines, workflow capabilities, pub sub, and integration with XML data, relational databases, and Web services.

Anzo for Excel includes an (RDFS and OWL-based) ontology editor that can be used directly within Excel. In addition to that, Anzo for Excel includes the capability to automatically generate an ontology from existing spreadsheet data, which is very useful for quick bootstrapping of an ontology.

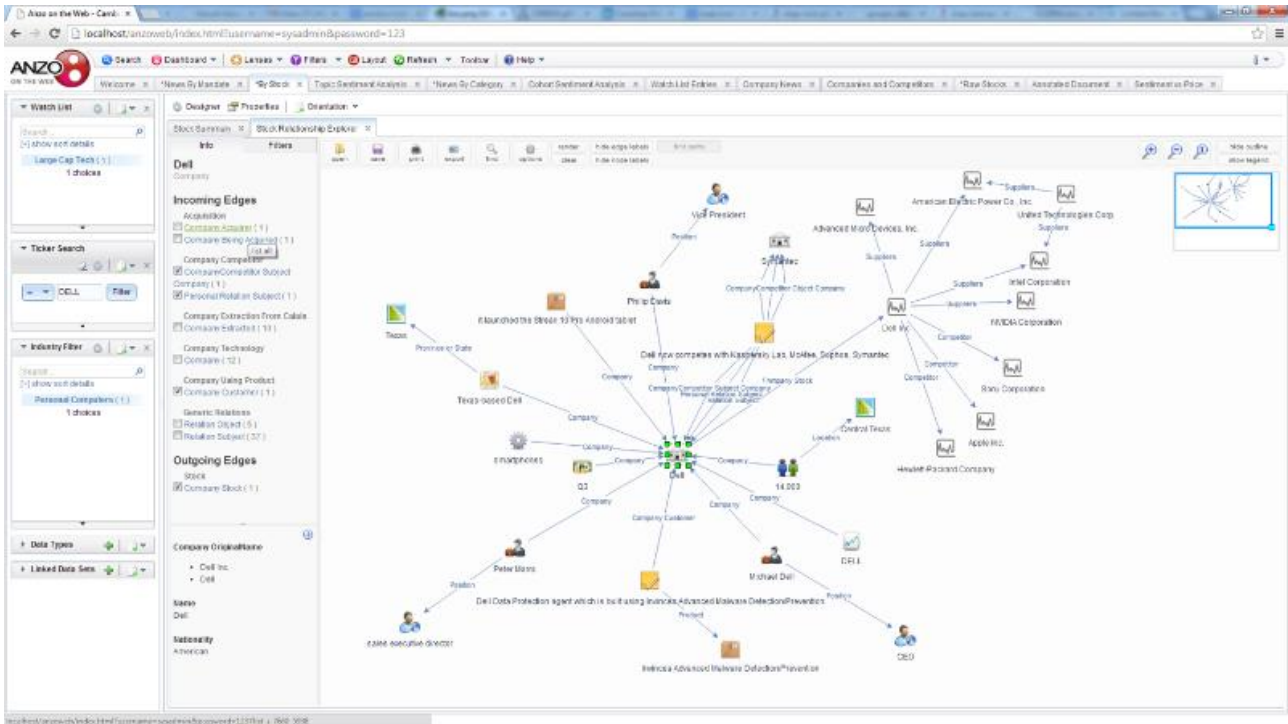


Figure 8 ANZO - Lenses view

¹¹ ANZO Solutions <https://www.cambridgesemantics.com/solutions-3/anzo/> [Last Access 7 January 2018]

Chapter 4 Background in Modelling and Simulation of Manufacturing Systems with Petri Nets

4.1 Modelling with Petri Nets

The doctoral dissertation of Carl Adam Petri in 1962 titled “Kommunikation mit Automaten” (in English, “Communication with Automata”) introduced the basis for a modelling formalism, hence the name Petri Net (PN). Petri Net is a multipurpose graphical and mathematical modelling tool applicable to many types of system, especially if concurrent, asynchronous, distributed, parallel, nondeterministic or stochastic.

A Petri net is a tuple $PN = (P, T, F, W, m_0)$, where

- P is a finite set of places,
- T is a finite set of transitions,
- the places P and transitions T are disjoint ($P \cap T = \emptyset$),
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation,
- $W: ((P \times T) \cup (T \times P)) \rightarrow \mathbb{N}$ is the arc weight mapping
(where $W(f) = 0$ for all $f \notin F$, and $W(f) > 0$ for all $f \in F$), and
- $m_0: P \rightarrow \mathbb{N}$ is the initial marking representing the initial distribution of tokens¹².

Kamath and Viswanadham [30] have discussed several advantages of Petri nets for general dynamic systems:

- PN allows the graphical representation and thus easy visualization of complex systems,
- PN can model a system hierarchically (systems can be represented in a top-down fashion at various levels of abstraction and detail),
- PN-based analysis techniques provide a systematic and complete qualitative analysis of the system,
- the existence of well-formulated schemes for Petri net synthesis facilitates system design and synthesis, and timed PN can evaluate system performance.

Desrochers, Al-Jaar, and DiCesare [31], [32] introduced the concept that Petri nets are useful for the modelling, performance evaluation, and control of manufacturing systems with the following characteristics:

- *Concurrency or parallelism* - In a manufacturing system, many operations that are enabled and do not interact take place at the same time.

¹² See the definition of a PN “token” in Table 12 (Section 6.2.1)

- *Asynchronous and synchronous* - Machines need variable amounts of time to complete their operations, and this variability must maintain the partial ordering of the occurrence of operations.
- *Deadlock* - A state can be reached where none of the processes can continue. This situation can occur when two jobs share two resources. More specifically, when one job takes the first resource and the other job takes the second resource, then both jobs cannot go further because two resources are occupied. The order in which two resources are used and released for two jobs needs to be arranged.
- *Conflict* - This may occur when two or more jobs require a common resource at the same time. For example, two workstations might share a common transport system or might simultaneously want access to the same database.
- *Event driven* - The manufacturing system can be viewed as a sequence of discrete events. Since operations occur concurrently, the order of occurrence of events is not necessarily unique; it is one of many allowed by the system structure.
- *Real-time control* - Petri net models can also be used to implement real-time control systems for an automated manufacturing system. They can sequence and coordinate the subsystems as a programmable logic controller does.
- *Mathematical foundation* - Petri net models have a well-developed mathematical foundation that allows a qualitative and quantitative analysis of the system.

The end user can rapidly learn the operational principles and the rules underlying these PN models, given the intuitive mechanisms and the simple elements (or modelling primitives). The graphical representation is used as a visual aid similar to flow charts, block diagrams and networks and assists both the designer and the final user of the net in its comprehension [33]. Moreover, thanks to the usage of the tokens, one of the main element of a Petri net, the user can analyse the dynamic behaviour of the system. The possibility of analysing the evolution of the system over time, combined with its mathematical foundation, makes the Petri Net formalism a powerful modelling and analytic instrument.

In addition to the advantages of Petri nets, Peterson [34] summarized some of the disadvantages as follows:

- The concurrency of operations has become more and more common. This has generally improved utilization and throughput but consequently increases the complexity.
- Subclasses of Petri nets increase the decision power, but at the cost of being unable to model a large number of systems. Extended Petri net models increase the modelling power, but in all known cases at the expense of decision power, since most analysis questions become undecidable.
- Petri net models have limitations in their inability to test for exactly a specific marking in an unbounded place and to take action on the outcome of the test.

In light of all this, Petri nets have been proposed for a very wide variety of applications. However, careful attention must be paid to a trade-off between modelling generality and analysis capability. That is, the more general the model, the less amenable it is to analysis. In fact, a major weakness of Petri nets is the complexity problem, i.e., Petri-net-based models tend to become too large for analysis even for a modest-size system. In applying Petri nets, it is often necessary to add special modifications or restrictions suited to the particular application [35]. For these reasons, over time are born different Petri net typologies from the original model, and each of them tries to avoid and overcome a particular original limitation.

Taking inspiration from the original work of Carl Adam Petri in 1962, over time many different typologies of Petri Net models were created. Each of these models had the aim to add specific characteristics that, according to the creator, could overcome some limits of the original archetypes. Historically, some of these models were combined in order to obtain a sort of "hybrid" one that could contain the characteristics of both the originating models, summing the advantages coming from them. In Table 1 a non-exhaustive list of Petri Net typologies founded in literature is presented in chronological order.

Table 1 Summary of main Petri Net typologies

Petri Net Typology	Acronym	Characteristics
<i>Classical Petri Net</i>	PN	A classical PN is the Petri net model derived from the original work of Carl Adam Petri in 1962 [36].
<i>Timed Petri Net</i>	TPN	Merlin (1974) and Ramchandani (1973) were the first who independently extended PNs to include time delays in different ways. Currently, the two proposed approaches which are named as firing durations and enabling durations are the basis of representing time in PNs. In timed PNs, although time delays can be associated to any transitions, places or arcs, it is often associated to transitions [37], [38], [39].
<i>Stochastic Petri Net</i>	SPN	A stochastic PN (SPN) is a Petri net where each transition is associated with an exponentially distributed random variable that expresses the delay from the enabling to the firing of the transition. Due to the memoryless property of the exponential distribution of firing delays, [40] showed that the reachability graph of a bounded SPN is isomorphic to a finite Markov chain [40].
<i>Fuzzy Petri Net</i>	FPN	PN structure is a four tuple consisting of places, transitions, tokens and arcs, and theoretically, each of these can be “fuzzified”. In other terms, Fuzzy PNs introduce uncertainty inside the net [41].
<i>Extended Petri Net</i>	EPN	A particular version of the classical PN model, enriched with more and different places, tokens, arcs and transitions [42].
<i>Generalized Stochastic Petri Net</i>	GSPN	A generalized SPN is basically an SPN with transitions that are either timed (to describe the execution of time consuming activities) or immediate (to describe some logical behaviour of the model) [39] [43].
<i>Coloured Petri Net</i>	CPN	In a Coloured Petri Net, each token has a value – called its ‘colour’ – to which a particular piece of information is attached [44].

4.2 Manufacturing System Analysis with Petri Nets

As already introduced in the previous sections, it is possible to list some general properties that can be linked with the Petri Nets models and that can be considered useful for the analysis of real systems: in some cases, these properties are utilised as necessary conditions for enabling the analysis. Variants of the definitions provided below can be found in literature, however, here we adopt the ones provided by [45] and [35].

These properties can be interpreted as concepts linked with the production systems (Table 2). Here the interpretations provided by [46] for some of the above cited properties:

Table 2 Interpretations of Petri Net properties in manufacturing

Petri Net Properties	Meanings in Manufacturing Systems
<i>Reachability</i>	A certain state can be reached from the initial conditions
<i>Boundedness</i>	No capacity such as a buffer, storage or workstation overflow
<i>Safeness</i>	Availability of single resource or no request to start and ongoing process
<i>Liveness</i>	Freedom from deadlock and guarantee the possibility of a modelled process or activity to be ongoing
<i>Conservativeness</i>	Conservation of no consumable resources such as machines, equipment and AGVs
<i>Reversibility</i>	Re-initialization and cyclic behaviour

In [47] the authors identified the following advantages in using these Petri net models for representing manufacturing systems:

- Petri Nets can model concurrency, asynchronous events, logical precedence relations, and structural interactions in a natural and simple way. Conflicts, blocking, finite buffers, synchronization, priorities, and assembly /disassembly operations can be modelled easily and efficiently.
- Petri Net models represent a hierarchical modelling tool with a well-developed mathematical and practical foundation. Structural analysis (deadlocks, mutual exclusion, liveness, and boundedness) and temporal analysis (performance measures such as throughput rate, resource utilizations, and in-process inventory) can be carried out using Generalized Stochastic Petri Nets (GSPNs), which are an extension to Stochastic Petri Nets.
- Since GSPN's are isomorphic to embedded Markov Chain's, using them does not require a deep knowledge of Markov Chains or stochastic systems. More importantly, simple Petri Net models can represent very complex Markov Chains.
- Their graphical nature helps us to visualize such complex systems: with respect to other techniques of graphical system representation like block diagrams or logical trees, Petri nets are more suited to represent in a natural way logical interactions among parts or activities in a system [48].
- The reachability graph can be automatically generated from the GSPN model and used to solve the equivalent Markov Chain. This frees the modeller from having to painstakingly account for all possible states of the system.
- Incremental changes to a PN model are done simply by adding tokens, places, and/or transitions. On the other hand, minor changes in a Markov Chain model usually require redefining all the states in the model.
- Petri Net models can also be used to implement real-time control systems.

Beyond the advantages related to the visualisation and modelling properties most characteristic of this methodology, Petri net models allow us to perform analysis on the system, for both what concerns effectiveness and efficiency analysis [49]. These two can be respectively translated into qualitative and quantitative analysis. Quantitative analysis concerns the evaluation of the efficiency of the modelled system whereas qualitative analysis concerns the effectiveness of the modelled system [50]. Quantitative analysis looks for performance properties such as throughput, average completion times, average queue lengths or utilization rates. Qualitative analysis searches for structural properties like the absence of deadlocks, the absence of overflows or the presence of certain mutual exclusions in case of resource sharing. For an exhaustive overview of the performance analysis that could be derived from the Petri nets, it is possible to refer to [51]

Therefore, due to its graphical nature, its ability to describe static and dynamic system characteristics and system uncertainty, and the presence of mathematical analysis techniques, Petri nets provide an appropriate conceptual infrastructure for the modelling and analysis of Manufacturing Systems [50] and over years they were widely used in several fields and applications.

In order to understand in which way and how PN models can be used, we combined different state-of-the-arts approaches [39], [52], [53] and [54] together with our requirements and results of the preliminary analysis. The output of this study is summarized in the following table, which represent a brief summary of main proposed typologies of Petri Net models found in the literature with the main characteristics and, for each type, the applications found in literature:

Table 3 Summary of the main typologies of Petri Net with possible uses

Type Of Petri Net	Characteristics	Uses
<i>Classical PN</i>	A classical PN is the Petri net model derived from the original work of Carl Adam Petri in 1962	<ul style="list-style-type: none"> • Modelling Generic Manufacturing systems • Modelling FMS • Process Planning • Modelling Shared-resource automated manufacturing systems
<i>Extended PN</i>	A particular version of the classical PN model, enriched with more and different places, tokens, arcs and transitions	<ul style="list-style-type: none"> • Uses are similar to the ones of classical PN
<i>Deterministic Timed PN</i>	Timed Petri nets associate deterministic times information with transitions and/or places in a straightforward way	<ul style="list-style-type: none"> • Cycle time • Throughput • Scheduling
<i>Stochastic PN</i>	A stochastic PN (SPN) is a Petri net where each transition is associated with an exponentially distributed random variable that expresses the delay from the enabling to the firing of the transition. Due to the memoryless property of the exponential distribution of firing delays, Molloy (1982) showed that the reachability graph of a bounded SPN is isomorphic to a finite Markov chain	<ul style="list-style-type: none"> • Modelling and analysis of complex manufacturing systems • The probability of a certain state (i.e. machine utilization) • The expected value of tokens (i.e. average inventory of the product)
<i>Coloured PN</i>	In a Coloured Petri Net each token has a value often referred to as 'colour'	<ul style="list-style-type: none"> • Modelling AGV networks in FMSs

<i>Coloured Timed PN</i>	In a coloured Timed PN (CTPN) the classical PN is enriched with time and attributes value	<ul style="list-style-type: none"> • Evaluation of the resources for achieving a plan: Are there enough resources (human operators and tools) to achieve a plan? • Evaluation of the time to achieve a plan • Evaluation of the workload of the operators: Are there any periods where some operators are overloaded or waiting idly? 	<ul style="list-style-type: none"> • Deadlock avoidance: What deadlocks should be avoided to ensure that the goal is achieved? What activities should be done in parallel to speed-up a task without creating deadlocks? • Heuristic algorithms for balancing an assembly line
<i>Coloured timed-oriented object nets (CTOONs)</i>	In CTOONs models, PNs are seen as objects in classes, and where new objects can be derived from other objects	<ul style="list-style-type: none"> • Facilitate reconfiguration of the PNs models • Modelling automated manufacturing systems and their performance evaluation 	
<i>Hybrid PN</i>	A hybrid PN can be obtained if one part is discrete and another part is continuous	<ul style="list-style-type: none"> • Modelling Networked manufacturing systems and control system architecture 	
<i>Fuzzy PN</i>	A PN structure is a four tuple consisting of places, transitions, tokens and arcs. The theor allows each of these to be <i>fuzzified</i> . Fuzzy PNs, thereby, also allow uncertainty to be represented	<ul style="list-style-type: none"> • Monitoring the shop floor: checking in real time if the events correspond to the normal behaviour 	<ul style="list-style-type: none"> • Monitoring the manufacturing schedule: checking the occurrence of unforeseen events or incidents
<i>Fuzzy Coloured PN</i>	Combinations between the concepts derived from the CPN and FPN	<ul style="list-style-type: none"> • Control of robot arms 	

The most relevant studies in control using Petri nets are focused on deadlock control for manufacturing systems. A deadlock is a situation, which occurs when a process or thread enters a waiting state because a resource requested is being held by another waiting process, which in turn is waiting for another resource [52]. In other words, if a process is unable to change its state indefinitely because the resources requested by it are being used by another waiting process, then the system is said to be in a deadlock. To overcome the deadlocks in manufacturing systems, there are three major applied strategies which are deadlock detection and recovery, deadlock avoidance and deadlock prevention [55], [56], [57] (Table 4).

Table 4 Summary of the three main possible deadlock approaches

Approach	Description
<i>Deadlock Detection and Recovery</i>	From the aspect of deadlock detection and recovery, resources allocation is processed without any verification . Therefore, the status and requests of these resources are checked periodically to determine if there is a set of deadlocked processes. This approach permits the occurrence of deadlocks but one deadlock occurs, a detection system determines the resources involved and recovery actions are then used to re-allocate these resources.
<i>Deadlock Avoidance</i>	Deadlock avoidance methods are of vital importance for automated manufacturing system control in order that correct deadlock-free resource allocation decisions can be made and continuing system operations can be guaranteed. At each system state, a control policy is used to determine on-line those system evolutions, among the set of feasible ones, which are correct.
<i>Deadlock Prevention</i>	Deadlock prevention has been widely achieved for automated manufacturing systems (AMS) and led to a huge number of results. Deadlock prevention is usually achieved by using a computational mechanism to control the request for resources to ensure that deadlocks never occur . The deadlock prevention is achieved either by effective system design or by using an off-line mechanism to control requests for which a deadlock situation is unacceptable.

Chapter 5 Ontologies in support of Manufacturing Systems Modelling and Simulation

In the manufacturing domain, ontologies are in their early stage of development. Several ontologies have been proposed with the objective of facilitating knowledge management and information exchange across the extended enterprise. As an example, information models such as Process Specification Language (PSL) [58] serve as a neutral language for integrating several process-related applications (including production planning, process planning, workflow management and project management) throughout the product life cycle. Some others information models, instead, are aimed at providing a shared vocabulary for communication between machine control and process planning software applications [59]. In general, manufacturing ontologies can vary due to the granularity level employed in the representation scheme, the extents and goal of the application, and the continuously evolving requirements and needs. Some ontologies are mainly aimed at providing terminological means for information integration while some others are geared toward enabling advanced reasoning through providing sophisticated knowledge structures [60]. Achieve full interoperability, however, remains a major challenge with respect to information representation in modelling and simulation of manufacturing systems. In this context, ontologies play a role in facilitating these activities [61]:

- Establish Purpose and Scope → Terminology harmonization to enable shared and clear understanding.
- Formulate Conceptual Model → Ontology knowledge is used to determine the unambiguously differentiated abstraction levels. Ontological analysis helps to map system objects to model objects and to identify appropriate object roles. Ontological analysis helps reason with system constraints to facilitate determination of model logic
- Acquire and Analyse Data → Ontologies play an important role in detailed data analysis, especially in disambiguating terminology and interpreting text data descriptions.
- Design Detailed Model → Ontologies will facilitate detailed analysis of objects and constraints including mapping the simulation model constraints to evidence (or specifications) of real world constraints in the domain descriptions.

Over the past two decades, some robust solutions have been proposed, covering most of the activities above. Hereafter, we present a summary of those solutions which are focused on two main topics: Ontologies designed for representing manufacturing system's (processes and products) information, and Ontologies designed to support Modelling and Simulation of PN-based manufacturing systems. Therefore, leveraging ontologies to identify model boundaries, level of modelling abstraction, model objects, and determine model structure and logic, and eventually refining modelling details, validate model structure and model logic.

5.1 Ontologies for Manufacturing Systems

The ADACOR (ADaptive holonic COntrol aRchitecture for distributed manufacturing systems, 2006) [62] is a holonic architecture that proposes an adaptive production control approach balances from a stationary state to a transient state, in normal and unexpected conditions, respectively, combining the benefits of hierarchical and heterarchical control structures using an adaptive mechanism. An up-to-date description of the current holonic and multi-agent systems solutions can be found in the surveys of [63]–[67], where a more detailed description of the aforementioned and other not mentioned solutions is made.

Generally speaking, the ontology here presented serves as a common vocabulary to address the interoperability of the holons, thus, supporting the sharing of local knowledge by the distributed holons. However, no detail is provided regarding the URI, the format, the eventual upper ontology, but it rather appears to be just a list of terms commonly agreed within the so-called communication

component. A so-called upper ontology for manufacturing domain named it MASON (MANufacturing's Semantics Ontology¹³, 2006) has been proposed in 2006 by Lemaignan et al. The ontology is built upon three elements: entities, operations and resources. *Entities* are all the common helper concepts. They aim to provide concepts to specify the product. It gives an abstract view of the product. *Operations* relate to process description. They cover all processes linked to manufacturing in a wide acceptance (Manufacturing, Logistic, Human, and Launching operation). Finally, *resources* stand for the whole set of manufacturing linked resource. Two applications of this ontology are exposed in [68]: automatic cost estimation and semantic-aware multivalent system for manufacturing. However, there is still a certain unclear on how such practical results are achieved, as well as, the methodological framework under which the model has been designed and the technical principles according to which the ontology has been developed.

Based on the assumption that an ontological model of a product may be considered as a facilitator for interoperating all application software that shares information during the physical product lifecycle, in 2012 the authors of the ONTO-PDM propose an approach for facilitating systems interoperability in a manufacturing environment [69].

To achieve this goal, the methodology used to develop the ONTO-PDM follows a two-step approach:

- Conceptualization of existing standards, related to product technical data modelling for the definition of products information, providing a "product-centric" information model to represent knowledge and concepts.
- Formalization of the proposed "product-centric" information model in terms of a Product Ontology, thus including business rules, to express and share product knowledge among systems [70].

The standards chosen for this scope are the ISO 10303 [71] and the IEC 62264 [72], [73]. These standards are commonly accepted to allow information exchange between ERP, CAD, PDM and MES applications, providing a sort of application-driven interoperability architecture (Figure 9).

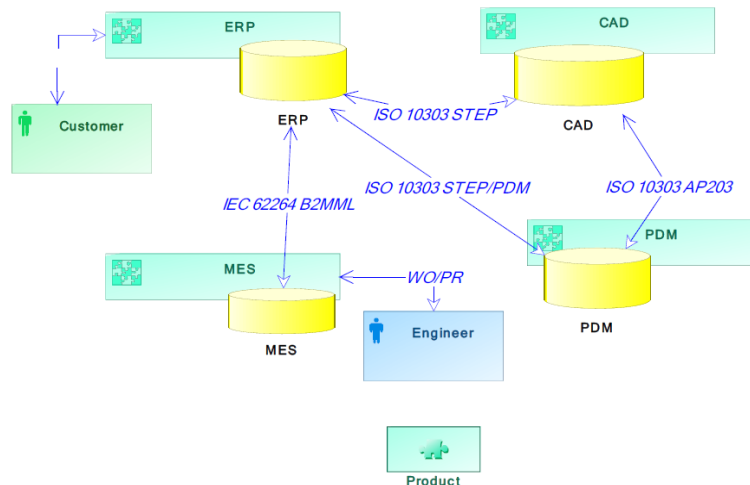


Figure 9 ISO 10303 and IEC 62264 standards within a manufacturing enterprise

A limitation in the research reported by the authors is related to the two selected standards. Indeed, if more standards need to be considered to extend the ontology, the current manual mapping process would represent a major effort [74]. Overall, the standardization of the methodological approach and the ontological model per se represent two main issues of this work. With regard to the applicability of such an approach to this research work, there is a loose correlation due to the granularity and scope of the ontological model. However, the approach is sound and relies on the widely adopted reference for manufacturing products.

The CDM-Core [75] is a publicly available ontology in the manufacturing domain in OWL2 format (available at <http://sourceforge.net/projects/cdm-core/>), designed on the basis of the DILIGENT methodology in 2016. It re-uses other ontologies

¹³ The OWL source files are available online on <http://sourceforge.net/projects/mason-onto>

publicly available, namely MASON [68] as Upper Ontology, SSN [76] domain –specific, Condition Monitoring [77] domain specific, vCard¹⁴ domain specific, Org¹⁵ domain specific, Time¹⁶ as Upper Ontology, TimeLine¹⁷ as Upper Ontology, DUL¹⁸ as Upper Ontology, SCORvoc [78], [79] domain specific. One of the objectives was to balance general (re)applicability and use case specificity, namely automotive exhaust production and metallic press maintenance. This large ontology was successfully tested in the framework of CREMA project (CREMA H2020-RIA project - Grant agreement no. 637066) for semantic annotations of process models, services, and sensor data. Furthermore, the approach presented by the authors has been thoroughly evaluated according to state-of-the-art methodologies and several given criteria. As a result, the approach and quality of the ontological model is robust, however, the *adaptability* of the CDM-Core ontology seems to be limited due to its focus on covering the use case domains defined within the CREMO project. Another limitation is given by the absence of a domain-neutral ontology (a Top Level ontology, such as BFO)

5.2 Ontologies for Petri Net Modelling and Simulation

The Petri Net Markup Language (PNML) [80] is an XML-based interchange format for Petri nets used as a starting point for a standard interchange format [81], [82] or Petri net tools, which aims to syntactically validate PN-based models. In particular, PNML provides an XML schema (meta-model) and several Petri Net Type Definition PNTD for different Petri net types.

Although XML schemas introduce a common structure to exchange information unambiguously, these lack any notion of implicit hierarchy, intentional definition of types [83], and meaning of exchanged information. PNML should be, therefore, extended with semantic web technologies in order to support automatic operation and reasoning on Petri nets [84].

In 2006, Gašević and Devedžić present a Petri net ontology with the purpose of enabling the sharing of Petri nets on the Semantic Web. This article uses a Petri net UML diagram (Figure 10) as a starting point for implementing the ontological representation [85], [86]. Same applies to the additional properties (or features), which can be attached to almost every core Petri net element (e.g., name, multiplicity, etc.) as illustrated in Figure 11.

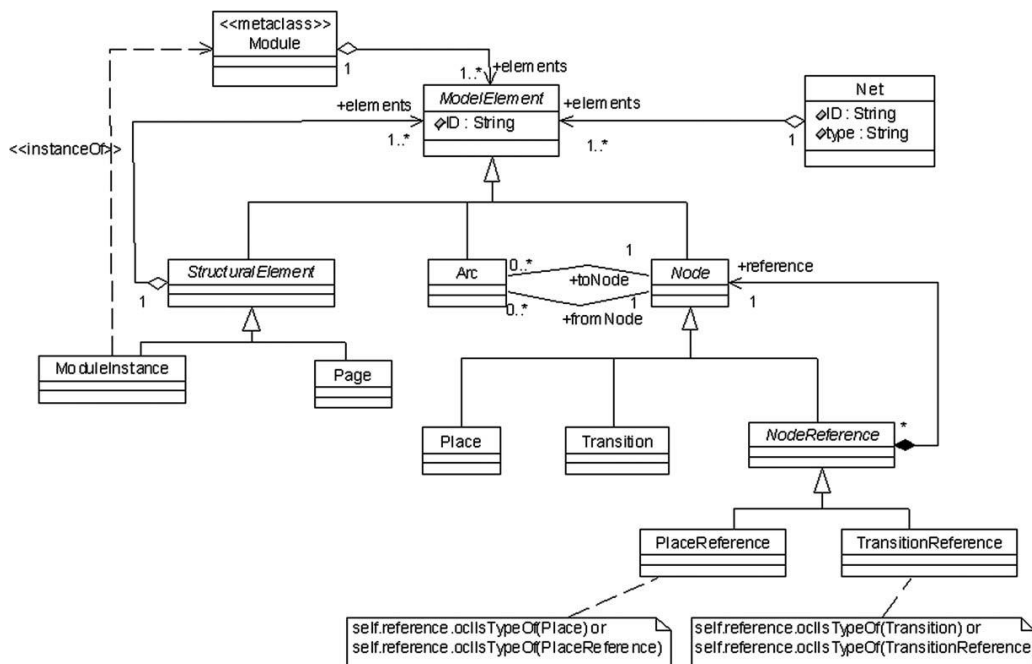


Figure 10 PN on the Semantic Web - UML Diagram of the basic structure

¹⁴ W3C Interest Group Note, vCard Ontology - for describing People and Organizations - <https://www.w3.org/TR/vcard-rdf/>

¹⁵ W3C Recommendation, The Organization Ontology - <https://www.w3.org/TR/vocab-org/>

¹⁶ W3C Recommendation, Time Ontology in OWL - <https://www.w3.org/TR/owl-time/>

¹⁷ Timeline OWL-DL Ontology - <http://motools.sourceforge.net/timeline/timeline.html#>

¹⁸ DnS Ultralite - http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite

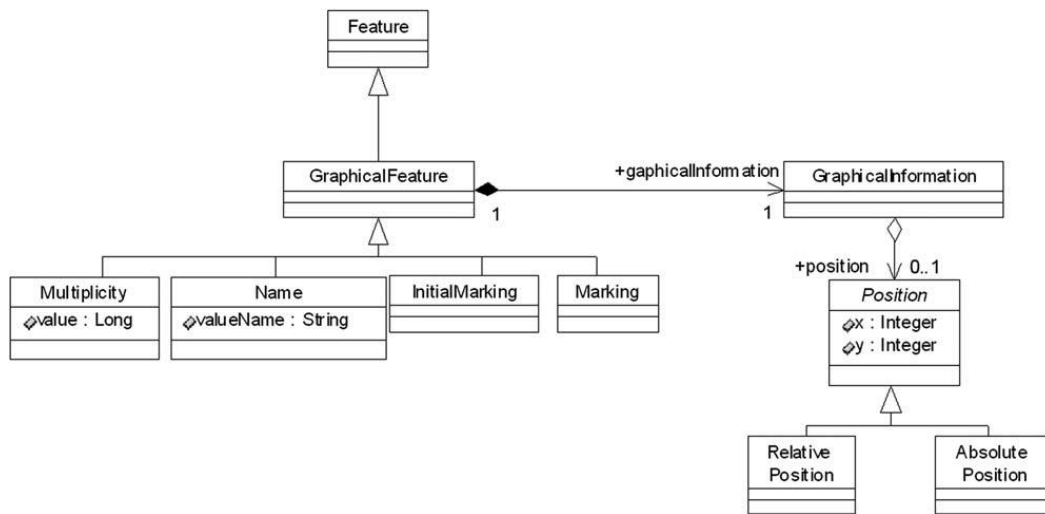


Figure 11 PN on the Semantic Web - UML Diagram of the graphical features

However, the UML representation does not let us validate semantically-enriched Petri net models, which is the reason why an OWL version of the ontological model implemented in Protégé 2000 has been also provided (Figure 12), with the mission of developing an OWL model that can be exploited for machine reasoning. Nonetheless, the model does not have any rule-based logic for model validation.

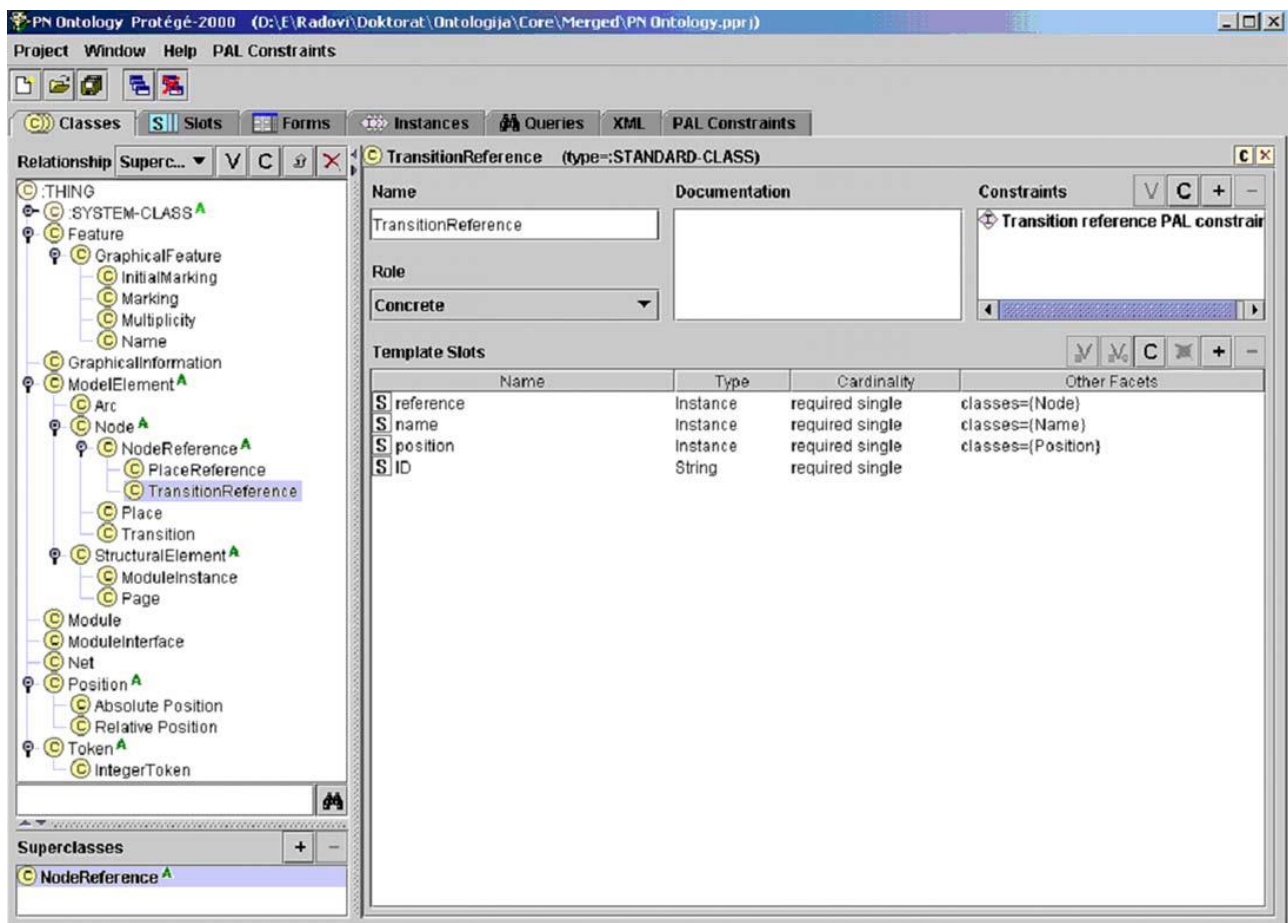


Figure 12 PN on the Semantic Web - Protégé implementation

In 2009 [84], a method to describe Petri nets with Petri Net Markup Language (PNML) in combination with OWL is discussed. Despite the more clearness is brought onto the PN semantics, no evidence of such an ontological model is available on the web. In the light

of these considerations, starting from Petri Net Core Model (Figure 13) and the PT Model (Figure 14), and using the approach presented in [87], [88], which leverages OMG’s Query/View/Transformation¹⁹ (QVT) transformation language in conjunction with the meta-models for UML and OWL 2, we built the representation hereafter presented.

Both Core and PT models are characterized by modelling structures such as *Composition*, *Multiplicity*, *Inheritance*, and *Association*. Nonetheless, following state of the art MDE approaches [89], [90] we can transform the PNML meta-models²⁰ into an OWL ontology.

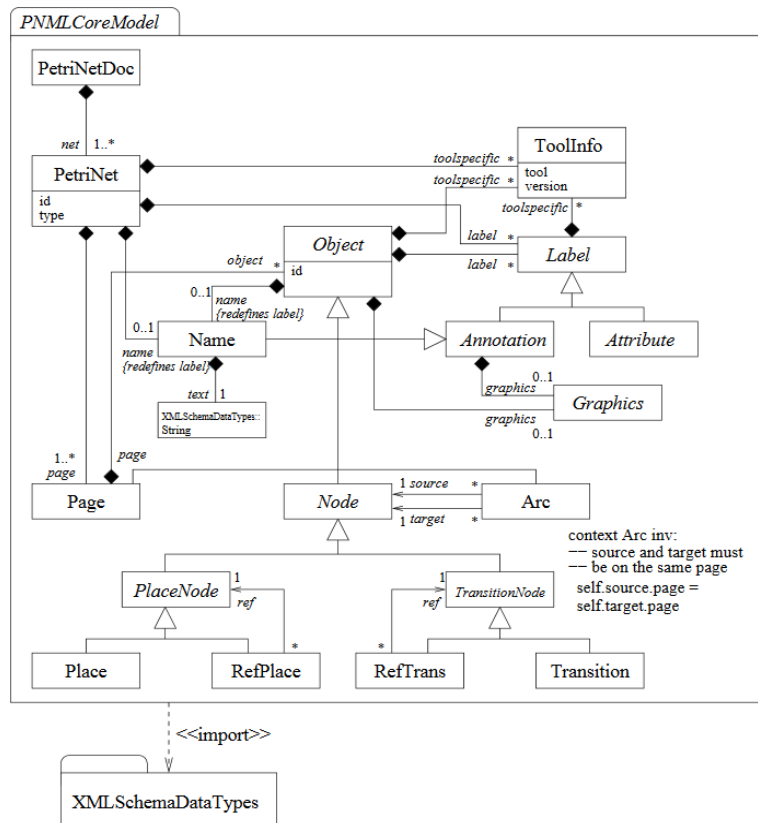


Figure 13 PNML Core Model

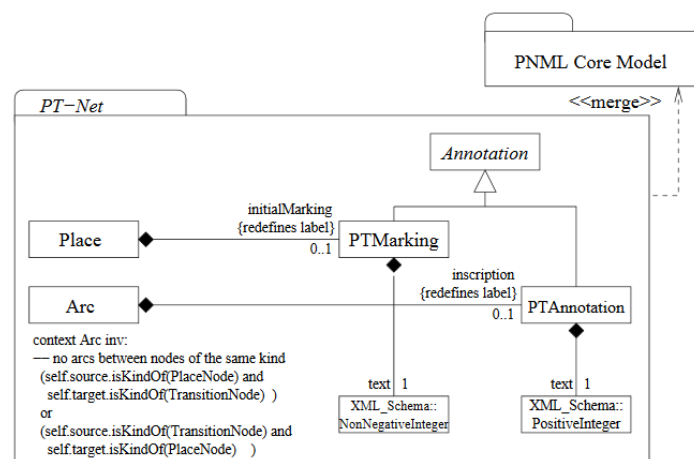


Figure 14 PNML PT Model

¹⁹ OMG MOF 2.0 QVT Specification 1.0 (2008), <http://www.omg.org/spec/QVT/1.0> (last access 7 July 2017)

²⁰ PNML Specification models - <http://www.pnml.org/grammar.php> (last access 7 July 2017)

5.3 A Semantic Framework for Modelling and Simulation of Manufacturing Systems: the proposed approach

In the light of the considerations presented above, the methodological approach here adopted is built on a set of technical principles and development activities, which characterizes all the domain specific ontologies hereafter presented:

1. Clear definition of context and scope of the representation (granularity)
2. Selection of the formats (more details in Section 3.1)
3. Analysis and re-use of the existing ontologies in the domain (likewise in CDM-Core)
4. Define any new entity starting from BFO (the most used Top Level Ontology)
5. Provision of textual (or natural language) definitions for each entity
6. Setting up of unique Identifiers & Naming Conventions for each new entity
7. Provision of a logic along with the set of domain specific entities to foster reasoning

The models presented in the previous section serve as a basis for the solutions introduced in the next one where specific issues are addressed in three different application cases. However, there is a need for further enrichment and specialization of those models, as well as the design of a framework that incorporates the all those technologies and solutions to answer the RQs of this work.

Figure 15 summarizes the proposed solution for embedding the aforementioned technologies. Such illustration introduces three elements through which the RQs will be answered in the next section:

- Data Integration and Structuring, achieved by extracting the information from the available documents and files, transforming them into structures elements having semantic tags defined according to state-of-the-art methodologies and ontological reference models, and load into triple stores, which allow the storage of information in a way that each record is structured as a set of subject, predicate, and object. This contributes to drawing conclusions around the **semantic enrichment** of industrial data (RQ1) and ontological resource **reusability** (RQ2);
- Semantics- Driven Model Transformation, achieved by leveraging SWRL rules-based reasoning, enables the translation (or transformation) of terms of one domain to terms of another specific domain (e.g. terms of a manufacturing system into terms of a Petri net resembling the latter). This contributes to answering the question around how to exploit **context-awareness** (or context-adaptiveness) for process analysis (RQ1) and semantic **interoperability** for manufacturing system models (RQ2). Moreover, it heavily contributes to the investigation of the impact on semantically-enriched manufacturing systems **modelling** (RQ3);
- Semantics-Driven Knowledge Creation, achieved by leveraging SPARQL and SWRL rules contributes to the **impact on simulation** applications for manufacturing system models (RQ3).

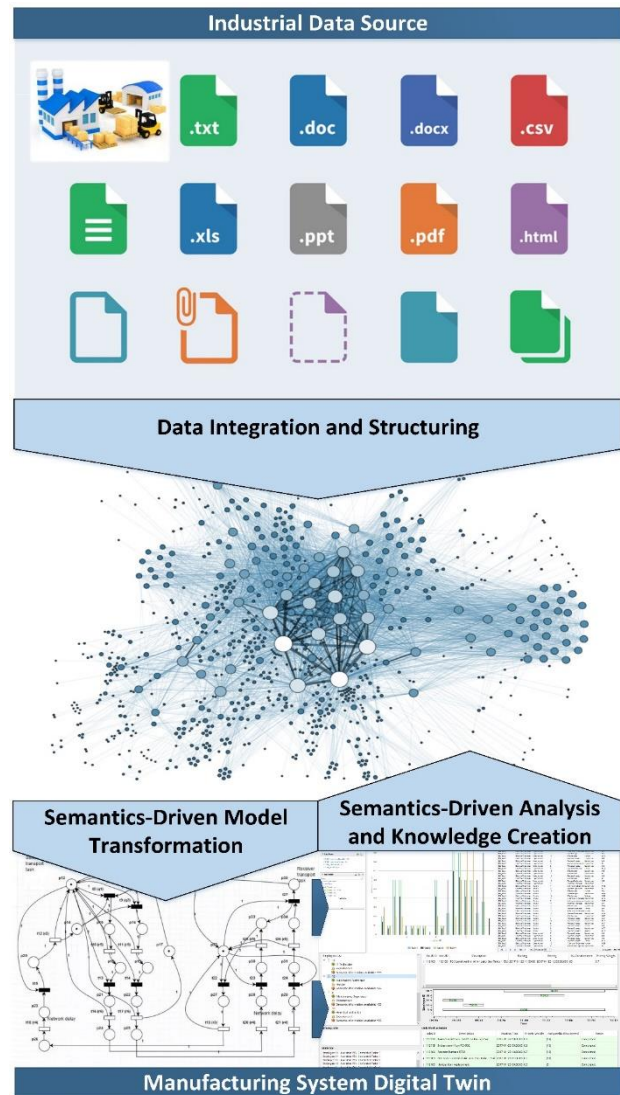


Figure 15 Overall proposed approach

Chapter 6 Modelling and Simulation of Semantically-Enriched PN-based Digital Twins

6.1 Application Case 1: Human Resource Management through Semantically-Enriched Data

Generally speaking, any industrial shop floor can be considered a highly dynamic environment, as diverse factors have to be taken into account while analysing the information flow and unexpected events are likely to occur. A rather challenging issue has consequently presented itself: optimized management. Examples of such unexpected events include machine failures, the advent of urgent jobs, due-date changes and changes in job processing time [91]. In this context, Human Resource Management (HRM) represents a cumbersome activity which spans across multiple elements, such as task design and analysis, scheduling and workforce selection and allocation, training and performance assessment among others [92], [93], and even then it may be further detailed for each person as experience level, skills, availability, and so forth. The HRM complexity can substantially increase with increasing data variability and cardinality (i.e. granularity and volume). In light of all this, automated solutions are called upon to simplify and monitoring this procedure, and therefore, to avoid solutions that take a sizable amount of time and effort, and to deal with both *static* and *dynamic* resource allocation²¹.

As for many different knowledge-intensive applications, ontologies play an important role in such case study since they provide a reusable, shareable, and formal representation of the domain knowledge which can be exploited to empower our information system with reasoning as well as semantics-driven analysis capabilities, thus transforming it into a smarter information system.

The application presented here aims to introduce a novel Human Resource (HR) optimization tool that takes into consideration the aforementioned static and dynamic datasets prior to their semantic enrichment, which is accomplished by the inclusion of semantic tags explicitly defined in an ontological model. As a result, a so-called semantically-enriched framework²² for the analysis and design of dynamically evolving shop floor operations has been developed and tested with real data acquired from a chemical process pilot plant on a real scenario. Such application case eventually not only contributes to the investigation of ontology modelling and analytics issues related to the exploitation of semantically-enriched data but paves the way towards answering **RQ1** (How to semantically enrich manufacturing system models and exploit context-awareness for process analysis?) and **RQ2** (How to use semantics to foster reusability and interoperability of manufacturing systems models?).

With this regards, two major results have been achieved here:

- Semantic enrichment of historical and real-time information from shop floor work/task scheduling offering enhanced weights on workforce allocation based on overall performance, task duration, worker affinity, etc., on industrial systems;
- Real-time task scheduling method towards HR optimization by utilizing Conditional Random Field (CRF) probabilistic models, semantically-enriched information, and semantic query and rule languages, namely SPARQL²³ and SWRL²⁴.

²¹ Namely, resources that are defined only once at the beginning of the shop-floor setup and changes are made rarely and only when required, and time-dependent resources that are evaluated every time one is required

²² Term coined within the framework of SatisFactory EU-funded Research and Innovation Action Project https://cordis.europa.eu/project/rcn/193393_en.html

²³ SPARQL Semantic Query Language (W3C) <https://www.w3.org/TR/sparql11-query/> [Accessed July 2017]

²⁴ SWRL Semantic Rule Language (W3C) <https://www.w3.org/Submission/SWRL/> [Accessed July 2017]

6.1.1 Design of a Semantic Framework for Smart HR Data Management

The solution here presented extends the work published by Zikos et. al. in 2016 [94] by employing an ontology-based approach to support – and optimize – the Human Resource (HR) assignment process built upon a so-called *comprehensive HR model*. Semantically-enriched information about tasks and workers is incorporated into the proposed task scheduling method [95], which automates the process of setting the starting time and selecting the most appropriate human resources for an arriving task [96]. As a result, the use of ontologies and ontology-based solutions for industry data modelling and smart information flow management paves the way towards the definition of a threefold goal targeting at:

- a. making explicit the connections between pieces of information;
- b. allowing machine reasoning and inference on semantically enriched data;
- c. enabling semantics-driven analysis of industry data.

Figure 16 highlights the information flow as well as the main blocks of the proposed solution (*TO BE*) [97] in comparison to the original design (*AS IS*). The HR optimization engine schedules arriving tasks efficiently by determining the starting time and the most suitable workers. To this end, it makes use of various static and dynamic parameters of tasks and workers as input. Whereas static information refers to all those parameters that do not change in the short period, dynamic information aims to describe all those parameters which basically trigger the optimization process and thus need to be semantically-enriched and meanwhile analysed each time the optimization process occurs. Further details regarding the types of data, the information extraction and their modelling requirements are given below.

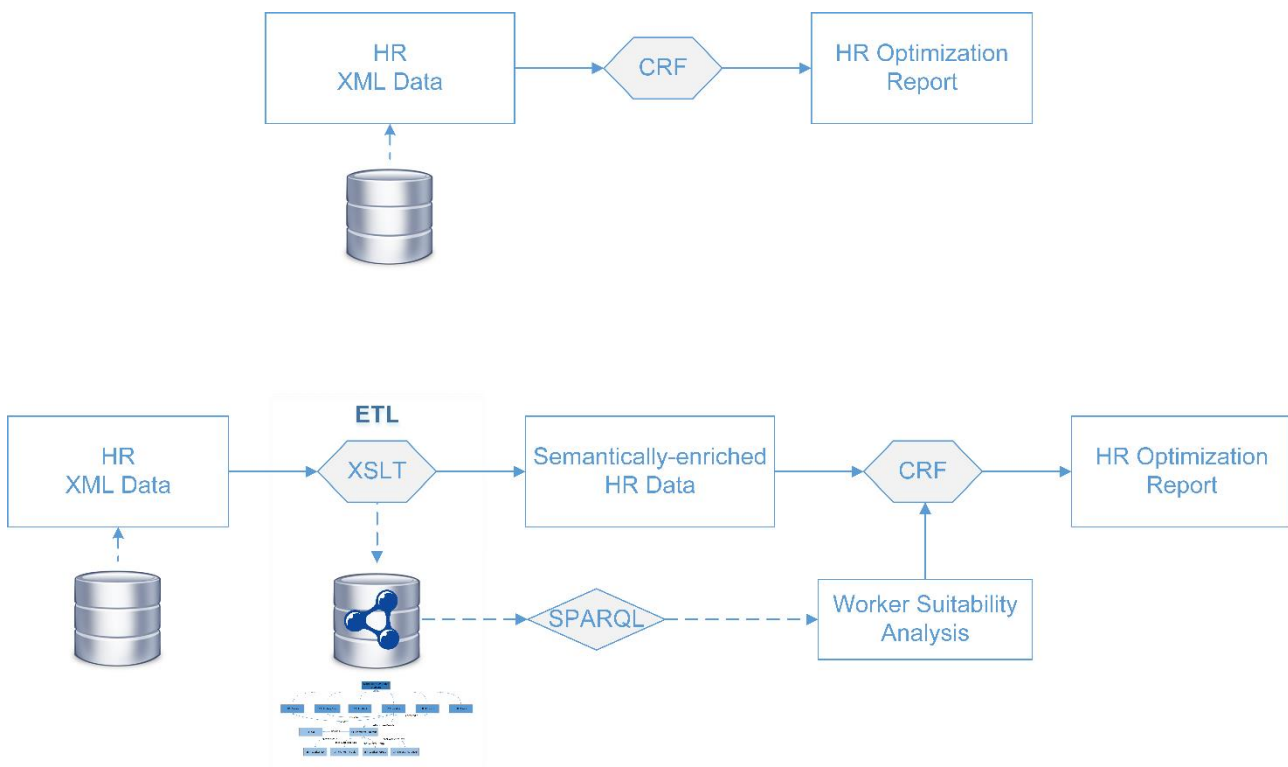


Figure 16 Proposed approach – AS IS (top side), TO BE (bottom side)

The data regarding activities, procedures and maintenance operations (either planned or unplanned ones) are based on the analysis of a real application scenario set up by the Chemical Process and Energy resource Institute (CPERI)²⁵ – in the framework of SatisFactory EU Project²⁶ – where numerous small-scale and pilot plants operate. Each plant comprises of a certain number of units where process operators work under the exclusive supervision one supervisor per unit (or process, depending on the complexity of the latter). At regular intervals, condition-based maintenance activities are performed that depend on the status of the involved process system and equipment. Also, as the processing unit handle high dynamically changing materials, e.g. experimental catalysts or feedstocks, at extreme conditions of pressure, temperature and flows, numerous unplanned maintenance operation appear that need to be handled upon request depending on the criticality. Overall, a Computerized Maintenance Management System (CMMS) handles the requests, which are addressed to the technical team that communicates with the Common Information and Data Exchange Model (CIDEM) to send appropriate information. Furthermore, the process plants are controlled by automation systems, mainly based on Supervisory Control and Data Acquisition (SCADA) architecture that send selected information from the Input/Output field to the CIDEM. The data structures used to syntactically validate and exchange data through the systems are lifted to a higher level of abstraction, and therefore, leveraged for ontology modelling. With this regard, the presented ontological model is not meant to be a replacement for any of existing information technologies, but rather an added layer that can leverage those assets for semantic interoperability.

Industrial environments, and in particular shop floors, host a vast heterogeneous network of distributed information sources that constantly produce data in various formats and intervals, introducing a rather challenging task of real-time information handling. Based on the Common Information Data Exchange Model (CIDEM) [95], a shared vocabulary of industrial knowledge through well-defined XML [98] entities, attributes, relationships, objects, etc., structured upon known standards (i.e. gbXML²⁷, ANSI/ISA-95 - B2MML²⁸) after following established methods (Chen 1976) and results from new studies [99], is adopted to ensure easy access, extended expansion capabilities and syntactic interoperability.

By covering static (Information Model - Figure 17) and dynamic (Events - Figure 18) real-time information flow from shop floor operation, the input requirements for both proactive and reactive workforce allocation are met. Characteristic examples of static information include workers, procedures, and assets of a shop floor that do not alter under normal conditions, while dynamic information is various time-dependent variables such as the time required for a procedure to actually be completed or shop floor measurements from sensors and actuators.

The HR optimization tool retrieves within the CIDEM static information about human resources, such as the trades and experience level of each worker, common procedures on the shop floor, the current work schedule and the requirements of the arriving tasks (dynamic information). When the work schedule is updated by the HR optimization tool, an event which contains the schedule is created and stored back into CIDEM so that the other components and the workers can be informed. The information stored in CIDEM is translated into RDF/XML through XSLT so as to be available to the Semantic Framework. The latter, in fact, interacts with the CIDEM to retrieve industrial data, enrich them with semantic tags, and carry out semantics driven analysis mainly based on two elements:

- **Worker Group**, the group that a worker belongs to (e.g. *Process Technician, Process Supervisor, IT Technician*, etc.), from which it is possible to infer the required skills.
- **Worker Experience**, the level of the experience of the worker belonging to a given worker group can be expressed as a discrete value (e.g. *Trainee, Novice, and Experienced*).

Worker Group and *Worker Experience* represent the core terms of the ontological model developed in the framework of SatisFactory EU project with the aim of supporting semantic interoperability of CIDEM-based data models. The model has been therefore named after it: *SatisFactory Ontology* (SFO).

²⁵ CPERI - <https://www.cperi.certh.gr/index.php?lang=en> (Accessed on 9 Oct 2018)

²⁶ SatisFactory project web site - <http://www.satisfactory-project.eu/satisfactory/> (Accessed on 9 Oct 2018)

²⁷ Green Building XML schema <http://www.gbxml.org/> [Accessed on Dec 2016]

²⁸ Business To Manufacturing Markup Language <http://www.mesa.org/en/B2MML.asp> (Accessed on Dec 2016)

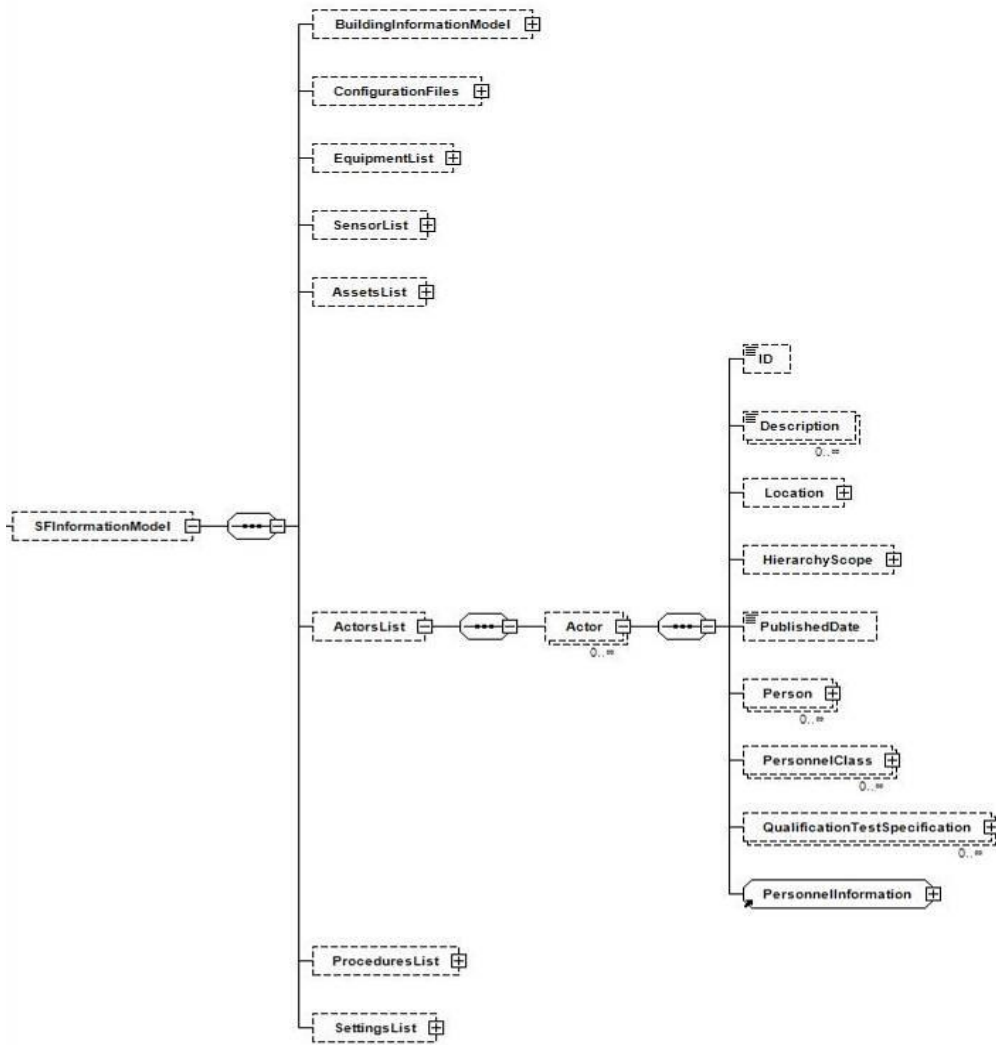


Figure 17 CIDEM XSD showing the static information of the industry (Workers) [95]

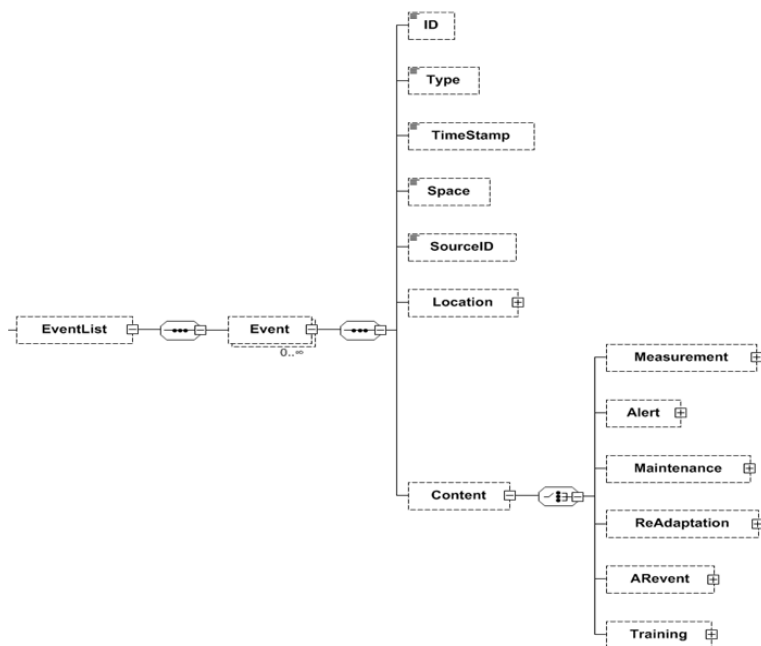


Figure 18 CIDEM XSD showing the dynamic information of the industry (Events) [95]

Regardless of the knowledge representation paradigms (frames, description logics, logic) used to formally represent knowledge modelling components (concepts, roles, etc.), and the languages used to implement the ontologies under a given knowledge representation paradigm [100], all of them share the following minimal set of components:

- **Classes** represent general elements, which are taken in a broad sense.
- **Relations** represent a type of association between elements of the ontology, usually binary relations where the first argument is known as the domain of the relation, and the second is the range.
- **Individuals** are used to representing the instances of a class or objects.
- **Formal axioms** serve to state something about *classes*, *relations*, and *individuals* that is always true, which are used to verify the consistency of the ontology – or knowledge base (KB) – itself.

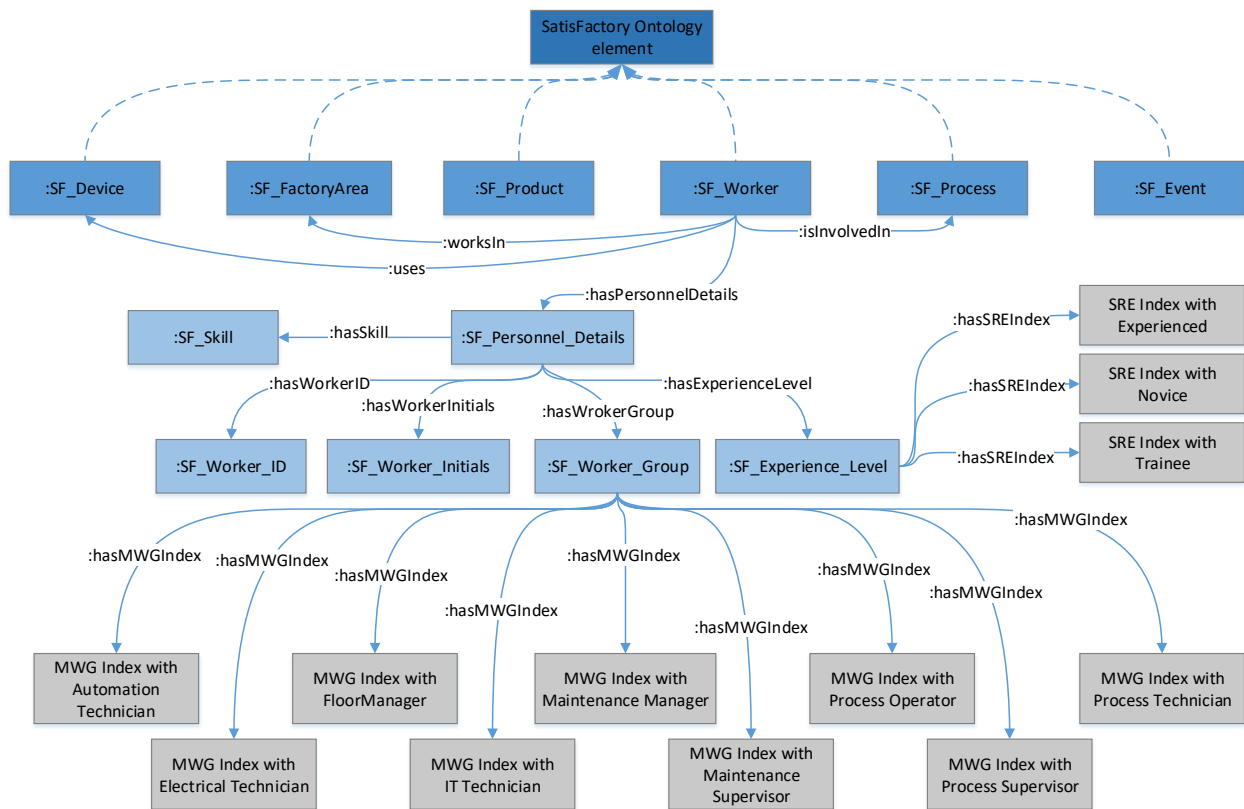


Figure 19 Ontology model excerpt from Satisfactory network of ontologies

Figure 19 shows an excerpt from the Satisfactory OWL ontology²⁹. Here, we are intending to emphasize the extension of the basic (blue) class structure with (grey) classes regarding the *Worker Group* and *Experience Level*, which introduce two pivotal elements aiming at making explicit the impact of these two elements while allocating workforce. In particular, the *Matchable Worker Group (MGW)* and *Suitable Requested Experience (SRE)* indexes aim to enhance the decision process regarding the choice of assigning a task to a worker who belongs to a worker group or has an experience level that differs from the expected one. Then, 9 (nine) worker groups have been identified, namely *Floor Manager (Group A)*, *Process Supervisor (Group B)*, *Maintenance Manager (Group C)*, *Maintenance Supervisor (Group D)*, *Process Operator (Group E)*, *Process Technician (Group F)*, *Electrical Technician (Group G)*, *Control Automation (Group H)*, *IT Technician (Group I)*. Therefore, a list of 26 (twenty-six) skills has been drawn up, describing the requirements of each specific worker group (the full list can be consulted in ANNEX). This list has been created by using the actual shop floor knowledge [101], collecting information from the managerial level of the chemical process pilot plant [102], [103], and further refined through a cyclic process lasted from the ontology design to the testing phase of the presented research work. Figure

²⁹ The complete OWL Satisfactory Ontology (v6.8) is available on GitHub <https://github.com/lano46/SFO>

20 describes, for example, the mandatory skills that two specific worker groups – Floor Manager and Process Supervisor – leverage while performing their assigned activities. This skills-based description of the worker groups will guide the definition of rules that enable inference-based validation of the worker’s profile according to his/her achieved skills:

$$\text{hasSkill some Skill and hasSkill only (Skill}_1, \dots, \text{Skill}_n) \rightarrow \text{hasWorkerGroup(Worker, [WorkerGroup])}$$

On the other hand, it allows the set-up of initial values for the *MWG* index and, at the same time, a base for its continuous adjustment and evaluation.

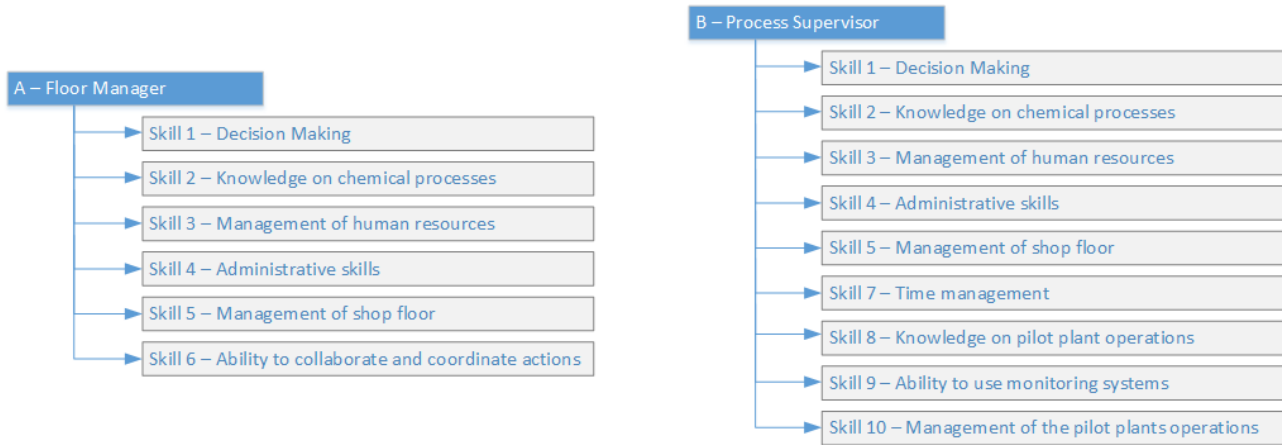


Figure 20 Examples of skills-based analysis of the Worker Groups A and B

That said, the *MWG* index values for each worker group could be extracted from a crossed analysis of the skill-based description of the worker groups. The meaning of this index can be perceived as follows, if Task X requires the worker Group H, although the only available worker groups are B and G, then such groups will match the task assignment in the following way:

- Worker Group B has *MGW*=16.7% for Task X
- Worker Group G has *MGW*=50.0% for Task X

This is because worker Group B has 1 out of 6 H skills (16.7%), while group G has 3 out of 6 H skills (50.0%). This initial statement has then been adapted according to the perception and knowledge of the experts working at the chemical plant where the semantically-enriched framework has been deployed and tested.

Table 5 shows all the possible values, however, we may generalize such a calculation as follows:

$$MWG_{(X|Y)} = \text{card}(S_X \cap S_Y) / \text{card}(S_Y)$$

Equation 1 Matchability index for X (available) and Y (required) Worker Groups

, where S_X and S_Y are the sets of skills of the available (X) and required (Y) worker groups. Therefore, $(S_X \cap S_Y)$ represents the set of skills that X and Y worker groups have in common while $\text{card}(\dots)$ denotes the cardinality of the set in brackets.

Much the same applies to the Suitability index for Requested Experience (SRE), which aims to make explicit the perception of the pilot plant responsible regarding the suitability of a specific level of experience while performing one particular task. Obviously, this does not take into consideration skills or explicit capabilities, in fact, it is rather related to the level of difficulty that can be attributed to each task. For example, Task 7 can be performed only by an experienced worker, while Task 4 is 100% suitable for a novice, 70% for a trainee, and 20% for an experienced worker. Table 6 shows the initial values that have been set by the chemical plant responsible by leveraging experience and knowledge of the chemical plant operations, therefore, it should be perceived as a living asset that will grow and continuously readapt while more knowledge will be gathered from the shop floor.

Table 5 MGW index values

		hasMatchabilityIndex_with								
		A	B	C	D	E	F	G	H	I
Worker Group	A	1.0	0.8	0.8	0.6	0.4	0.2	0.2	0.4	0.3
	B	0.6	1.0	0.5	0.3	0.9	0.4	0.1	0.2	0.1
	C	0.7	0.6	1.0	0.9	0.3	0.6	0.6	0.5	0.2
	D	0.7	0.6	0.9	1.0	0.3	0.6	0.6	0.5	0.2
	E	0.0	0.7	0.1	0.0	1.0	0.2	0.1	0.1	0.1
	F	0.1	0.3	0.3	0.2	0.7	1.0	0.6	0.5	0.2
	G	0.1	0.2	0.2	0.2	0.4	0.8	1.0	1.0	0.2
	H	0.1	0.2	0.2	0.2	0.5	0.9	0.7	1.0	1.0
	I	0.1	0.1	0.2	0.1	0.3	0.6	0.6	0.6	1.0

Table 6 SRE index values

Task ID	Experience Level			Task ID	Experience Level		
	Experienced	Novice	Trainee		Experienced	Novice	Trainee
1	1.0	0.4	0	12	1.0	0.5	0
2	0.7	1.0	0.3	13	0.3	1.0	0.4
3	1.0	0.4	0	14	0.5	1.0	0.2
4	0.2	1.0	0.7	15	1.0	0.6	0.2
5	1.0	0.5	0	16	1.0	0.7	0
6	1.0	0.2	0	17	1.0	0	0
7	1.0	0	0	18	1.0	0.5	0
8	1.0	0.3	0	19	1.0	0.7	0.3
9	1.0	0.4	0	20	1.0	0	0
10	1.0	0.7	0.4	21	0	1.0	1.0
11	1.0	0.6	0	22	1.0	1.0	0.6

As mentioned above, the shop floor data flow serialized and stored within the CIDEM in XML format is then transformed to RDF (Resource Description Framework) [104], which is a standard developed by the W3C to help create metadata for describing web resources, enabling their sharing through the Semantic Web [105]. Its data model consists of three object types: resources, properties, and statements. For the actual transformations, XSLT is used (Figure 21), which is a W3C Recommendation for defining XML data transformation and presentation.

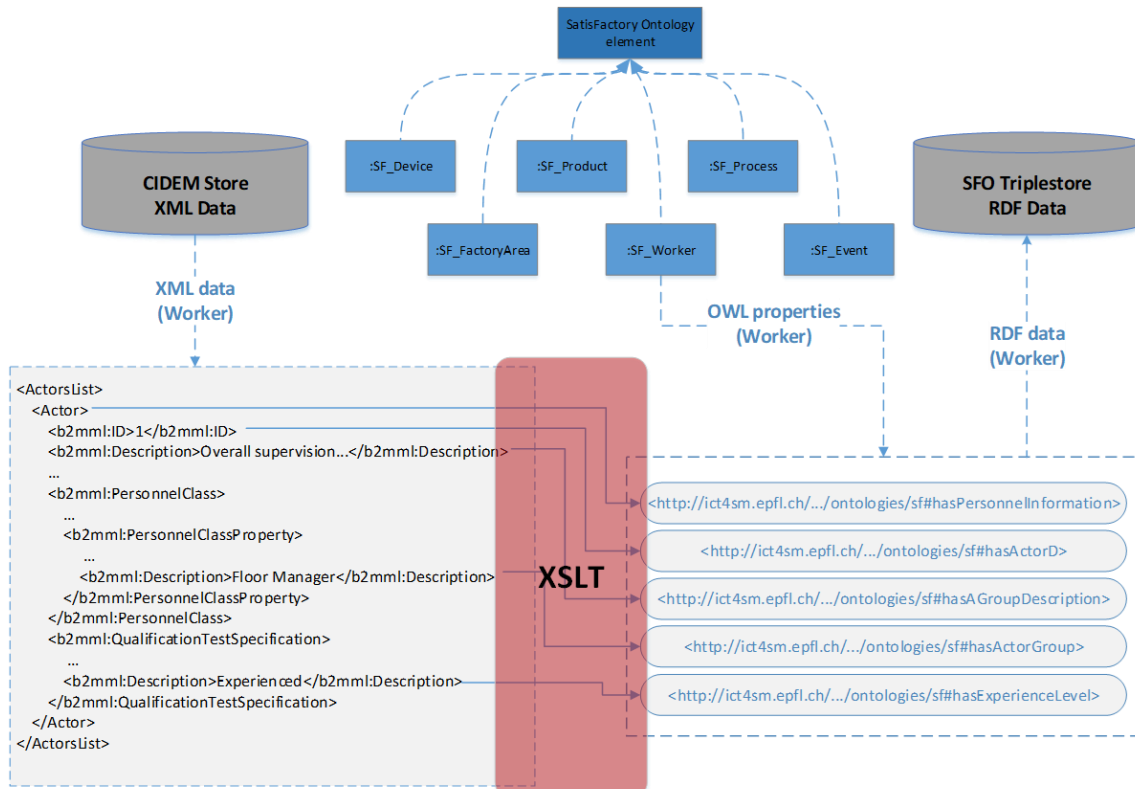


Figure 21 RDFizing shop floor data through XSLT

The combination of OWL structures and RDF data allows the use of advanced semantic querying using SPARQL (SPARQL Protocol and RDF Query Language) [106]. SPARQL, in fact, can be used to express queries across diverse data sources, whether the data is stored as RDF (or viewed as RDF via middleware). SPARQL contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions. SPARQL also supports extensible value testing and constraining queries by source RDF graph, which is a language used to express queries across diverse RDF data sources³⁰. The syntax of SPARQL queries is similar to SQL (Figure 22) and the results can be results sets or RDF graphs.

Besides being collected, semantically-enriched, and stored within a RDF triplestore, the shop floor data can be queried to support advanced assessments of human resources, i.e. the *Worker Suitability Analysis (WSA)*, which exploits both *MGW* and *SRE* indexes to provide a quantitative analysis of the suitability of a worker prior his assignment to one specific task. Roughly speaking, the SPARQL query engine carries out the aforementioned *WSA* according to the following requirements (Figure 23):

- *RDF datasets*, providing through the clause **WHERE** the basic graph patterns (or triple patterns) to match against the data graph.
- *Semantic Data selection*, which identifies the variables to appear in the query results.

Each solution gives one way in which the selected variables can be bound to RDF terms so that the query pattern matches the data. In particular, the *WSA* results highlight the unexpected suitability of some workers for tasks that are not designed for their work group. Such analysis extends the pool of workers to be taken into consideration while assigning a certain task.

PREFIX (Namespace Prefixes) e.g. PREFIX ont: <http://example.com/ontology/>
SELECT (Result Set) e.g. SELECT ?ExperienceLevel
FROM (Data Set) e.g. FROM <http://example.com/ontology/workers.rdf>
WHERE (Query Triple Patterns) e.g. WHERE {?worker ont:hasExperienceLevel ?ExperienceLevel}
ORDER BY, DISTINCT, etc. (Modifiers) e.g. ORDER BY ?ExperienceLevel

Figure 22 Query Structure

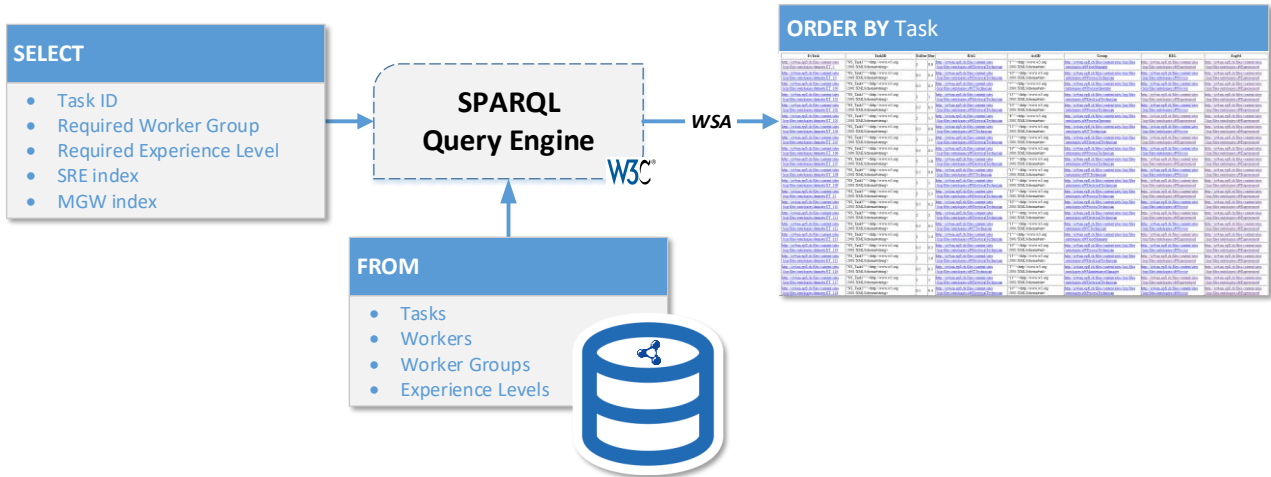


Figure 23 Worker Suitability Analysis - SPARQL query engine

Next section describes the way such query results support the HR optimization process.

³⁰ SPARQL Query Language for RDF <https://www.w3.org/TR/rdf-sparql-query/> (Accessed on 25 Feb 2019)

6.1.2 Walkthrough of the Human Resource Optimization and Task Scheduling Tool Implementation

This application case is driven by the needs and requirements of an industrial lab where more than 30 process plant configurations exist. The formulation of the models and the parameters that are considered by the developed methods and algorithms are in line with the actual data from the shop floor into consideration. Also, the HR optimization is based on the analysis and the needs as described by the nature of the operations that are performed by the groups within the technical team. In this context, both the subsequent data analysis and the obtained results are gained from the actual environment and the respective information as communicated to the CIDEM from the CMMS and the SCADA of each plant. Furthermore, the static parameters that are used are a subset of the parameters that characterize the technical team (Worker) and the activities that take place at the respective plants (Task). For simplicity reasons only information that is related to the HR optimization are considered in this work, although the rest of the structure is fully functional and described the entire infrastructure of the shop floor. Finally, the experimental results use a subset of activities that are requested by the Process Operators or are initiated by a planned maintenance program. Overall the HR optimization addresses the need to appropriately allocate the time of the technical team according to the requested expertise, availability and criticality of each Task. Moreover, due to the fact that the requested Tasks are more than the technical team can handle in quantity, it is necessary to consider the priority of the Task as well.

The objective of the real-time HR optimization engine is to schedule arriving tasks efficiently by determining the starting time and the most suitable workers. To this end, it makes use of various static and dynamic parameters of tasks and workers as input (Table 7 and Table 8).

Table 7 Static input parameters

STATIC PARAMETERS	
Task	Worker
Task type ID	Unique ID
Number & trades of workers	Trade
Estimated Duration	Experience level
Criticality	Shift times (start/end)

Table 8 Dynamic input parameters

DYNAMIC PARAMETERS	
Task	Worker
Priority weight	Current task
Starting Time	Remaining workload

When scheduling an arriving task in real-time, the algorithm determines the starting time of the task and assigns it to specific workers based on multiple criteria. By utilizing as input the information presented above along with the shop floor's current daily work schedule, it outputs an updated work schedule. A solution derived for an arriving task includes the scheduled starting time and the ID of worker(s) selected to perform the task. In order to schedule a new task, the algorithm ranks the workers according to suitability for a specific starting time of the new task and evaluates the different possible solutions. The solution of the minimum cost is selected and the new task is added to the work schedule, which is updated. The suitability of each human resource (worker) is evaluated individually. The rules derived from the enterprise's policies are combined with the output from a probabilistic model, semantically-enriched information, and other metrics which are computed dynamically. The use of probabilistic models, which are built using historical data from past task assignments, allows the use of the estimations produced about the suitability of a worker.

Both hard and soft constraints are considered when making decisions about task assignments. A weighted factor is given to each soft constraint parameter to define its significance. Rules that must not be violated are derived from the company's policies and preferences and incorporated into the hard constraints, which are checked first.

One of the main factors affecting the assignment of workers is derived from the output of a conditional random field (CRF) probabilistic model [107], [108], which is the probabilistic model type selected. CRFs are a type of discriminative undirected probabilistic graphical model and are applied in many fields (e.g. machine learning) where sequential data are available. A CRF is able

to take context into account as its output at a given time step depends on a sequence of past observations. Each observation consists of a set of features, where each separate feature is a value that can affect the output. The linear chain CRF type that is utilized is an alternative to the related Hidden Markov Models (HMMs) [109]. The main advantage of a CRF when compared to an HMM is the ability to include more complex non-independent features of the observations. A linear chain CRF representation is depicted in Figure 24. Y_t represents the hidden state at time t , whereas X_t represents the visibly associated observation at time t .

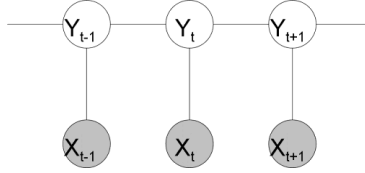


Figure 24: Representation of a linear chain CRF

The probability of producing a state sequence y given an observation sequence x can be computed by the equation below,

$$p(y|x) = \frac{1}{Z(x)} \exp \left\{ \sum_{k=1}^K \lambda_k f_k(y_t, y_{t-1}, x_t) \right\}$$

Equation 2 Probability of producing a state sequence y given an observation sequence x

, where f_k is a feature function; λ_k is the learned weight associated with feature f_k which is determined from training data, and $Z(x)$ is the following normalization function.

$$Z(x) = \sum_y \exp \left\{ \sum_{k=1}^K \lambda_k f_k(y_t, y_{t-1}, x_t) \right\}$$

Equation 3 Normalization function given the feature function f_k ; and weights λ_k

The normalization function $Z(x)$ is the summation over Y , which denotes the set of all possible state sequences, and it is used in order to get a feasible probability. A feature function f_k expresses a characteristic of the empirical distribution that needs to be held in the model distribution. For instance, it can be equal to 1 if $y_t = S1$, $y_{t-1} = S2$ and $x_t = O1$ and equal to 0 in all other cases. From Equation 2, it is therefore evident that the state sequence is modelled as a normalized product of feature functions.

In the HR management context, a CRF model is created for each worker. The model runs online and at each time step it takes as input a feature vector (observation) and infers the hidden state. The estimation produced at each time step is the model's hidden state and, in this case, it can be either 0 which means that the corresponding worker is not recommended for the task or 1 which means that the corresponding worker is suitable for the task. Furthermore, the model's class membership probability of each possible state is also returned. The class membership probability is a numerical value in the range [0-1]. The observation constructed at each time step comprises the following features:

- The type ID of the arriving task to be assigned.
- The worker's conformance to the trade required by the task.
- The worker's remaining workload level.

A training phase is required in advance in order to create each CRF model. A sequence of observations along with the actual ground truth label for each observation must be provided as input for learning the model parameters for each worker. The training data per model are created concurrently for all workers by using historical assignment events in the form: $\langle \text{Timestamp}, \text{TaskTypeID}, \text{WorkerID} \rangle$.

Figure 25 illustrates the steps of the algorithm that is followed when a new task has to be added to the work schedule.

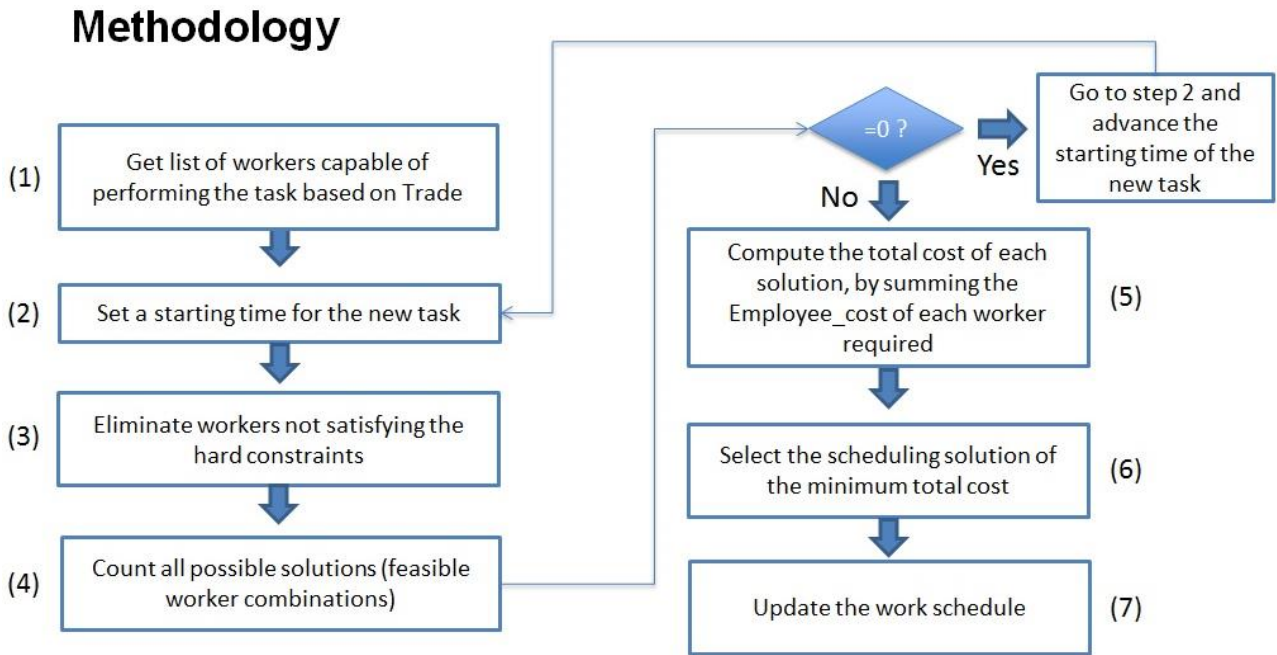


Figure 25 Steps of the task scheduling algorithm

The first step is to filter out the workers who are incapable of performing the task based on the worker trades required. A starting time for the task is then set. Initially, the starting time is set to 1 minute ahead of the current time to allow an incoming task of high priority to start as soon as possible. In the next step, the hard constraints are checked for eliminating the workers that do not satisfy them. For instance, such a constraint applied is related to the task weight parameter. The task weight is a numerical value which is computed based on the priority and requirements of the task. The description of the method that is followed for calculating the task weight is beyond the scope of this study.

The weight of the new task and the weight of a scheduled overlapping task (if any) are compared to each other. In the case of

$$\text{NewTaskWeight} > \text{ScheduledTaskWeight}$$

, or for a scheduled running task

$$(\text{NewTaskWeight} - \text{RunningTaskWeight}) / \text{RunningTaskWeight} > 20\%$$

, then the particular worker is included in the candidate list. Otherwise, the worker is eliminated. The feasible combinations of workers according to the human resource requirements of the arriving task are set. In the case of no set of workers being found at this step, the algorithm returns to step 2 and advances the candidate starting time of the task that has to be scheduled to the expected ending time of the next task in the schedule. The total cost of each solution is computed by summing the individual costs of all the workers required by the task. The selection cost of a worker, which should not be confused with a financial cost, is regarded as a real number in [0-1], and therefore the “lower the better” approach is adopted.

$$\text{Sel_workers_cost} = \text{cost}(E_1) + \text{cost}(E_2) + \dots + \text{cost}(E_n)$$

Equation 4 Selection cost components

, where $\text{cost}(E_i)$ is the selection cost of worker E_i and is given by:

$$\text{cost}(E_i) = W \cdot \text{ECapCost} + (1-W) \cdot \text{EAdaptCost}$$

 Equation 5 selection cost of worker E_i

, where ECapCost denotes the worker’s capability cost and EAdaptCost denotes the worker’s adaptation cost. The cost of selecting a particular worker depends on these two main factors: the capability cost and the adaptation cost. The former considers the suitability

of the worker in terms of experience on the particular task and instances of past assignments. The latter considers the effect of the new task on the already scheduled tasks in the task list in case the worker is selected. W is the assigned weight [0...1], which is configurable, and was set to 0.5 for the experiments of this study. The weight can be set to any value depending on the aspect one would like to put an emphasis on, or it can be changed dynamically. The worker's capability cost is computed using the following equation:

$$E_{CapCost} = ((1 - P_{model}) + (1 - SS)) / 2$$

Equation 6 Worker's capability cost

, where P_{model} is the estimated probability of selecting the worker, which is produced by the worker's CRF classifier. SS denotes the suitability score that is computed by the WSA for a particular task and worker.

The adaptation cost ($E_{AdaptCost}$) takes into account the amount of overlap with an already scheduled task expressed as a percentage, the priority weight difference between them, and the interruption factor in case a task is running and has to be suspended. The latter introduces additional costs when a task with short estimated time remaining has to be suspended in order to perform the new task. The adaptation cost is considered in order to limit the number of frequent and overwhelming changes in workers' task lists. Frequent changes usually cause frustration for workers. When a worker is available for the whole time period, during which the arriving task is planned to be performed, his adaptation cost is equal to 0, as previously assigned tasks are not affected.

Figure 26 shows a screenshot of the HR optimization tool User Interface (UI), which presents information about the workers, the assigned tasks per workers, and the current status of all tasks. At the initial step, static information about workers is loaded. Semantic data is queried on demand and stored for each worker prior to starting the operation of the HR optimization engine. However, it must be pointed out that Figure 26 does not reflect the 2 case scenarios presented hereafter, which are thoroughly supported by other screenshots.

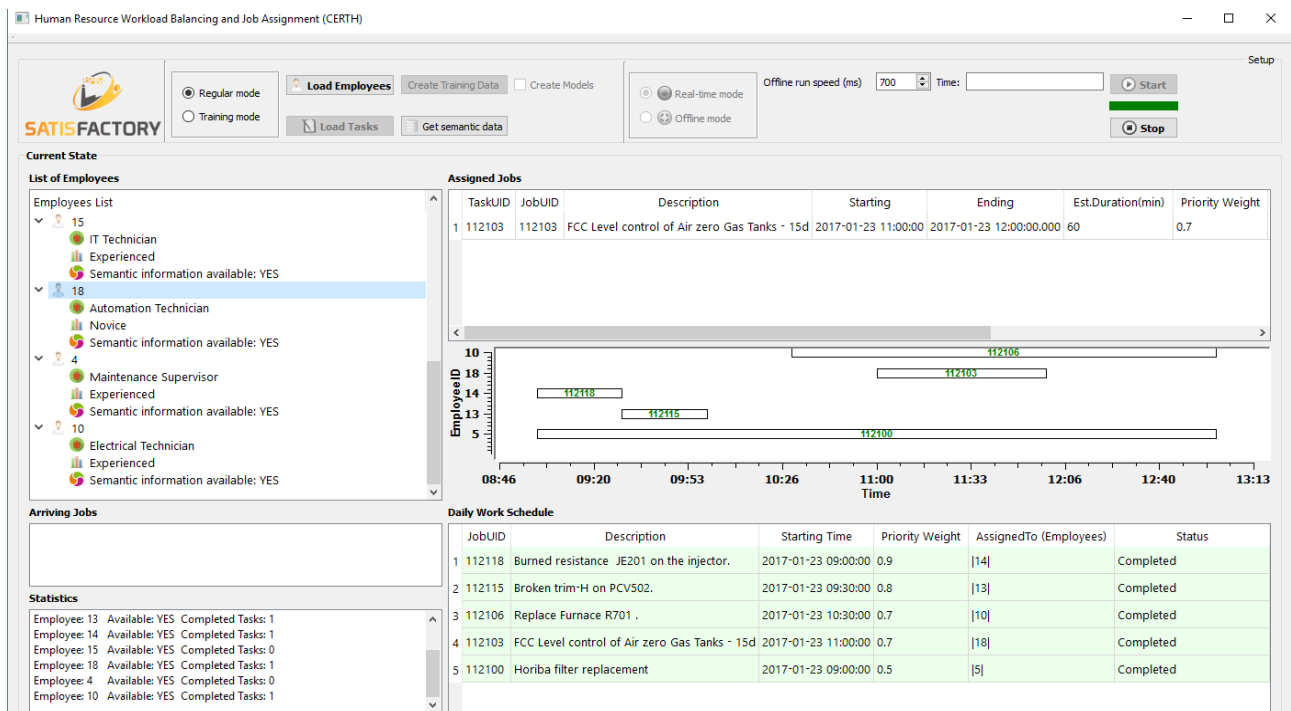


Figure 26 UI of the HR optimization tool

Real data about tasks and workers were acquired from a shop floor where chemical processes take place, in order to evaluate the operation of the task scheduling and HR assignment method presented when utilizing semantic data. Different types of technicians are included, such as Automation and Electrical technicians, along with operators. The algorithm was run in offline mode under different initial states. The scenarios that are demonstrated in this study involve arriving maintenance tasks of higher priority weight compared to the already scheduled tasks and therefore the arriving tasks can be scheduled to start immediately.

The first scenario demonstrates the operation of the HR engine in the case when a high priority task (*Task_4*; computed weight=0.65), which requires one Electrical technician, arrives. As semantic information is not available in this scenario, there are only two Electrical Technicians (ID_13, ID_10) candidate for carrying out this task. The task list per each technician when the new task arrives is depicted in Figure 27 (a) below.

The new task (labelled as *Task_4*) that has to be added to the work schedule arrives at 09:15 and has an estimated duration of about 4 hours. Both the Electrical Technicians are available when the task arrives, however, the addition of the new task results in overlapping with *Task_2* of worker ID_10 and with *Task_1* of worker ID_13. The engine assigned a new task (*Task_4*) to worker ID_10, as it is shown in Figure 27 (b). Even though the capability cost of worker ID_13 is lower than worker ID_10 (0 versus 0.2), the adaptation cost of worker ID_10 is much lower (0.04 versus 0.31), resulting in lower total selection cost (0.12 versus 0.155). *Task_2* was automatically re-scheduled to start later, after the completion of the new task. This scenarios, therefore, show the standard functioning of the HR engine, which does not fully leverage the evaluation of the ECapCost as it is equivalent to P_{model} , when semantic data is not available ($SS = 0$).

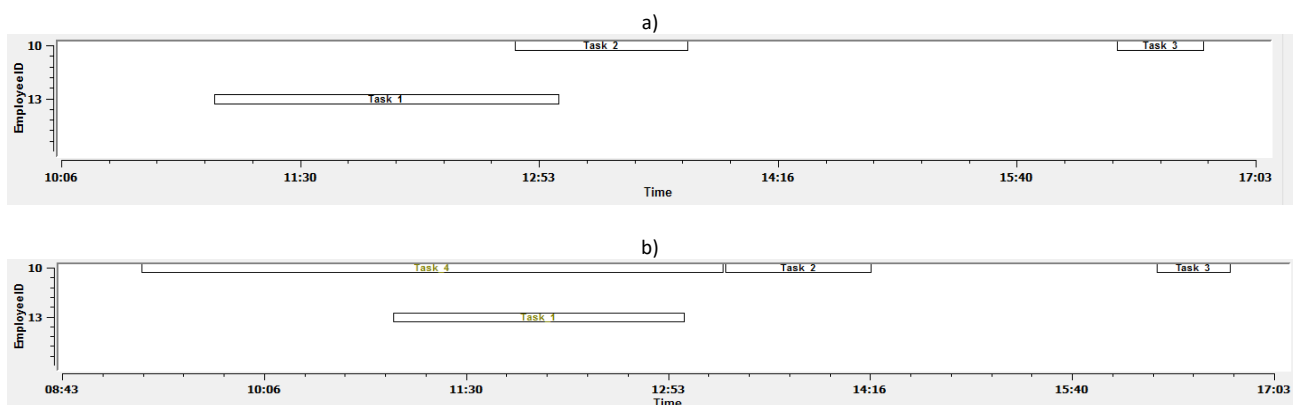


Figure 27 Task schedule before (a) and after (b) the assignment of a new task

The second scenario describes a situation that involves a different class of technicians, the Automation Technicians. In this case, the output of the engine was tested in two separate cases. In the first case, the semantically-enriched data (in short semantic data or semantic information) was not available and therefore the capability cost is deduced only from the CRF probabilistic models (as in the previous scenario), while in the second case semantic data was available. A new maintenance task of high priority (*Task_5*), which requires an Electrical Technician, is created at 11:35 and its estimated duration is one hour. This time there are only two available Automation Technicians (ID_14, ID_18) that result being candidates for executing this task due also to the skill-based analysis of the working groups, which extend the pool of suitable workers.

Table 9 Available workers for Task_5

ID	Expected Worker Group	Expected Experience Level
14	Automation Technician	Experienced
18	Automation Technician	Novice

Their task plan at the time of the arrival of the new task is shown in Figure 28 (a) below. It can be observed that when the new task arrives, both technicians are busy: technician ID_14 is performing *Task_1*, while technician ID_18 is performing *Task_2*. Due to the fact that the new task is of significantly higher priority, it has to be started as soon as possible which means that the running task will be suspended.

Work Schedule prior to semantic enrichment

In the first case, the HR engine was run without taking into account the semantically-enriched information when computing the capability cost of each worker. Figure 28 (b) depicts the updated task list of the workers and we observe that the technician ID_18 was selected to perform the new task (*Task_5*). The adaptation costs computed for the two workers were almost the same (0.496 for worker ID_18 versus 0.502 for worker ID_14). Furthermore, regarding the capability cost computed based on the probabilistic model of each worker, worker ID_18 was estimated as slightly more capable of performing the task with an estimated cost of 0.01

as opposed to 0.06 for worker ID_14. Thus, the final selection cost for worker ID_18 was 0.253, while for worker ID_14, it was 0.281, resulting in the assignment of the task to worker ID_18. The suspended task *Task_2* was scheduled to be performed after *Task_3* in order to avoid possible overlap.

Semantics-driven analysis of a worker

The WSA aims to perform an analysis of the workers' suitability for a specific task. As mentioned above, here the focus is on the worker profiles, which is enriched with further semantic data, such as the MWG (*matchability* index for worker group), SRE (*suitability* index for requested experience). In particular, the WSA performed on *Task_5* returns the results presented in Table 10. The evaluation of the WSA score (aka Suitability Score or SS) for a task *T* that requires the worker group *Y* whereas the available worker(s) belong to the worker group *X* is carried out by a SPARQL query that goes through the linked data and combines the effect of the MWG and SRE indexes, hence, taking into account cross-information regarding the Worker Group (WG) and more granular characterization of the Experience Level of the worker under analysis.

$$SS_{(T,X,Y)} = \omega_{WG} \times MWG_{(X|Y)} \times \omega_{EL} \times SRE_{(T)}$$

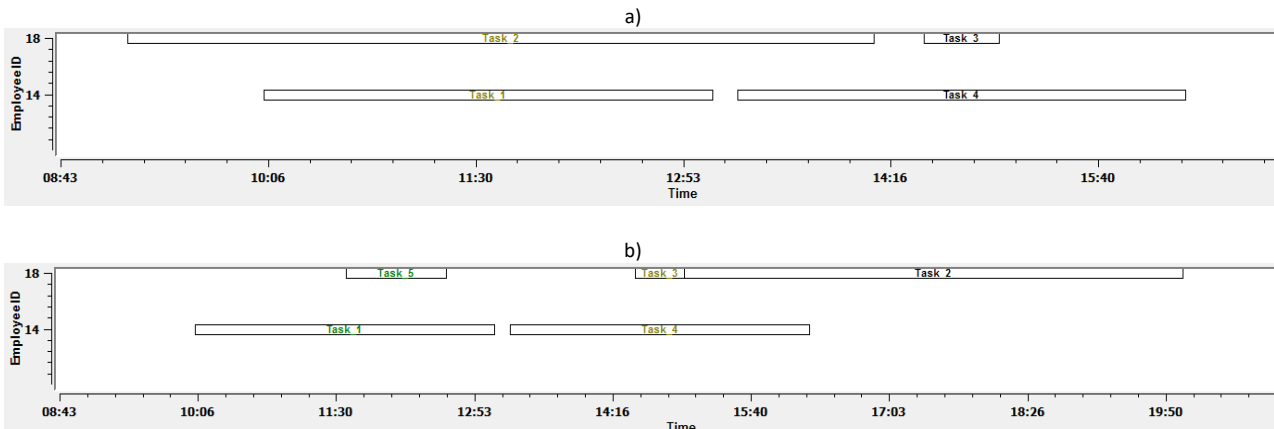
Equation 7 Suitability Score

Table 10 WSA results on Task_5 (designed for an Experienced Electrical Technician)

ID	Expected Worker Group	Expected Experience Level	SS score
14	Automation Technician	Experienced	1.0
18	Automation Technician	Novice	0.5

After applying the semantic information

The output of the HR engine produced in the case when semantically-enriched information was taken into account for the scenario presented above, is illustrated in Figure 28 (c). It can be observed that the assignment decision is different in this case, as the technician ID_14 was selected to perform the arriving task. More particularly, the suitability score produced from the semantic analysis for technician ID_18 was 0.5 (SRE= 0.5; MWG = 1), while the score for technician ID_14 was 1 (SRE= 1; MWG = 1). As a result, the capability cost of technician ID_14 was significantly lower than ID_18 (0.03 versus 0.255), and by extension, the total cost of selecting technician ID_14 was 0.266 while the corresponding cost for technician ID_18 was 0.3755. The suspended *Task_1* of technician ID_14 was scheduled to be continued after *Task_4*. The addition of semantics to the information flow feeding the decision process favoured the selection of the technician who is able to perform the specific task more efficiently and in a shorter time due to the combined evaluation of his skills and level of experience. This type of – historical – information is not included in the probabilistic models that were created. The estimations produced by the probabilistic models indicate the suitability of each worker to perform the arriving task, according to the task type and worker's current state (remaining workload level) and based on historical task assignment events. However, with the utilization of the semantically-enriched information provided by the Semantic Framework, the determination of suitability of each worker for performing a task can be further enhanced. This is due to the fact that the suitability score computed by the proposed framework takes into account the skills as well as the performance of each worker.



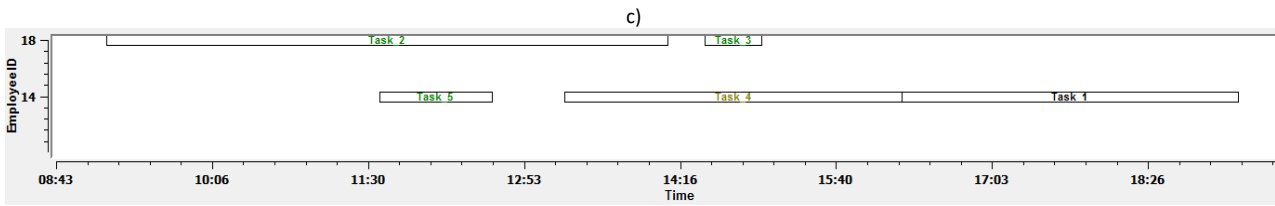


Figure 28: (a) Task list before the new task assignment (2nd scenario), (b) Output results without semantic information and (c) Output results with semantic information

6.1.3 Discussions around the application case

The aim of this application case is to investigate to what extent the development and use of a semantics enriched framework may enhance the analysis and assignment of dynamically evolving shop floor operation. The exploitation of semantic models hence the employment of knowledge engineering methodologies, such as ontology modelling, to achieve this goal is presented. In particular, a network of ontologies designed for this application case has proved to be effective in terms of domain elements representation and easiness of model (re)use and exploitation.

The proposed solution has the potential to be fully integrated within the deployment environment and to provide feedback to the CMMS, thus, offering an optimized approach for the overview of the already scheduled and newly arrived requests for process maintenance activities. In this framework, technical managers mainly gain from the employment of such a tool since they can use it as a means to monitor online the current status of the shop floor activities and obtain metrics related to the lack of system servicing. Further, the HR optimization tool can be used by the process managers to obtain an online and clear indication related to the status of their process plants and schedule appropriately their operations. Both technical and process managers can, therefore, have a view of the HR allocation that concerns them and make knowledgeable decisions for their daily or longer-term operations. Finally, the proposed HR optimization tool has proved to have the potential to improve productivity by reducing the response time required for making HR allocation decisions when new tasks have to be added to the work schedule during the working hours. Quality gains can be also acquired since multiple factors are taken into account by the engine when making allocation decisions, based on the policies and preferences that are applied at the factory.

6.2 Application Case 2: Reliability Assessment of an Automated Assembly Station

This section presents two case studies aiming to demonstrate the benefits of using ontologies to model and simulate industrial systems, such as an assembly station in the production line. In particular, two key aspects are addressed by the proposed ontology-based solution and hereafter discussed:

- Ontology-driven creation and semantic enrichment of the manufacturing system’s modelling elements, or rather, modelling primitives of a specific modelling language, such as Petri Net [110].
- Semantics-driven analysis of a digital twin’s simulation outcomes aimed at enhancing the upstream modelling phase and the downstream decision making process [111].

The creation (or instantiation) of OWL elements resembling Petri Net-based manufacturing system’s modelling primitives, which fundamentally constitute the PN-based digital copy (or digital twin) of the latter, is investigated and validated through the analysis of one case study extracted from the literature and thereafter partially adapted to the current framework. This serves as a testing environment on which all the features brought in by the proposed solution are verified. The overall modelling and simulation approach is therefore extended and applied to a second real case study on which strengths and limitations are further tested and discussed.

The application case’s findings contribute to the existing knowledge by answering – almost exhaustibly – all three Research Questions. Those findings, indeed, span aspects such as the semantic enrichment of manufacturing system models, the reusability of modelling primitives, the impact of semantically enriched manufacturing system models in modelling and simulation (M&S) applications.

6.2.1 Ontology-driven creation of OWL elements for PN-based manufacturing system models: design and preliminary validation tests

The case study here presented deals with the reliability analysis of a PN-based model of an assembly station on which each of its elements (or components) may fail and be replaced with a given occurrence rate. An overview of the main concepts, system elements and common practices for the reliability assessment of a manufacturing system – and, in particular, of an assembly station, – is provided throughout this section. The ontological model, which drives the instantiation (or creation) of the modelling primitives that in turn constitute the PN model resembling the assembly station, however, represents the core of the whole application case and it is meticulously presented below.

Design of ontological models for reliability assessment of an assembly station

The ontological model called Reliability Assessment for Assembly Stations Ontology (RAASO) has been designed with the purpose of representing all the elements that play a role in the modelling and simulation results analysis of an assembly station introduced in [112]. This ontology comprises of 20 terms that comply with the technical principles presented in Section 5.3 and thoroughly describe the application domain (up to the required granularity level). Table 11 shows these terms (RAASO OWL classes). where the short prefix *RAASO*: stands for “http://www.semanticweb.org/RAASO/RAASO_”.

The taxonomy of the terms (or classes) above is shown in Figure 29. BFO serve as the Upper Ontology from which the domain specific ontological model is developed (and aligned with), however, as far as the *analysis and re-use of the existing ontologies* principle is concerned³¹ BFO is not the only ontological resource that has been here used to develop the RAAS Ontology. Two more resources have been indeed imported and employed, i.e.:

- the Information Artifact Ontology (IAO) - IRI: <http://purl.obolibrary.org/obo/iao.owl>
- the Relation Ontology (RO) - IRI: <http://purl.obolibrary.org/obo/ro.owl>

³¹ More details in Sections 3.1 and 5.3.

Table 11 RAAS Ontology: URIs, Classes and Definitions

URI	Term	Definition
RAASO:00001	Assembly Station	A machine in a factory which automatically performs assembly operations on a product and comprises of several components.
RAASO:00002	Component	The smallest part of an assembly machine which can be either operational or failed
RAASO:00003	Component Function	A disposition of a component that is realized in response to a planned event (or activity)
RAASO:00004	Component State	A quality of a component which can be Operational, or Failed
RAASO:00005	Component Failure State	A quality that inheres in a physical component, which is triggered by a Failure Cause event.
RAASO:00006	Component Operational State	A quality that inheres in a physical component that is not failed
RAASO:00007	System State	A quality that inheres in a physical system, why can be either Operational or Failed
RAASO:00008	System Failure State	A quality that inheres in a physical system, which is due to the presence of at least one component in a Failure state.
RAASO:00009	System Operational State	A quality that inheres in a physical system that has no failed components
RAASO:00010	Initial Operational State	An Operational State of a System before the execution of an Assembly Activity
RAASO:00011	Terminal Operational State	An Operational State of a System after the execution of an Assembly Activity
RAASO:00012	Component Failure Rate	An Information Content Entity that statistically characterizes the mean time between failures (MTBF) of a component
RAASO:00013	Component Replace Rate	An Information Content Entity that statistically characterizes the mean time to be repaired (MTTR) for a component
RAASO:00014	Assembly Activity Execution Time	An Information Content Entity that statistically characterizes the mean time for an activity to be executed
RAASO:00015	Assembly Station Event	A process that can be either planned (e.g. Assembly Activity) or unplanned (e.g. Component Failure Cause)
RAASO:00016	Assembly Station Planned Event	An assembly station event which occurs in virtue of a planning activity.
RAASO:00017	Assembly Station Unplanned Event	An unplanned assembly station event of which occurrence can be statistically described by a Component Failure rate.
RAASO:00018	Assembly Activity	A planned Assembly Station Event that has an Initial Operational State and a Terminal Operational State
RAASO:00019	Component Replace Event	A planned Assembly Station Event which contributes to the reestablishment of a Component Operational State
RAASO:00020	Failure Cause Event	An unplanned Assembly Station Event which contributes to the Component Failure State condition

The Information Artifact Ontology (IAO) is an ontology of information entities (artifacts). An *information artifact* is, loosely, a dependent continuant or its bearer that is created as the result of one or more intentional processes. For examples, the English language, the contents of this document or a printout of it, the temperature measurements from a weather balloon³² (more details can be found on [113]). The OBO Relations Ontology (RO)³³ is a collection of OWL relations (Object Properties) intended for use across a wide variety of biological ontologies, however, it has been proven to be applicable to other domains.

The PN4R Ontology has been developed with the purpose of describing all those modelling PN elements that are required to represent and simulate a manufacturing system with the ultimate goal of assessing its reliability. The ontology consists of 28 terms, whether 8 of them are imported from a (basic) PN Ontology designed to describe the so-called static and dynamic structures of a Petri net.

³² Ontobee server - <http://www.ontobee.org/ontology/IAO> (Accessed on 7 Jan 2018)

³³ Ontobee server - <http://www.ontobee.org/ontology/RO> (Accessed on 7 Jan 2018)

Table 12 PNML³⁴ and PN4R Ontologies: URIs, Classes and Definitions

URI	Term	Description
PNML:00001	System Net (Petri Net)	An algebraic structure with two sets, one called places and the other called transitions, together with their associated relations and functions, and named after their inventor, Carl Adam Petri.
PNML:00002	Marking (of a net)	A state or marking of a Petri net (graph) is a multiset of its places. Whereas the marking of a place is a multiset of tokens associated with the place
PNML:00003	Net Structure	A directed graph comprising a set of nodes of two different kinds, called places and transitions, and their interconnection by directed edges, called arcs, such that only places can be connected to transitions, and transitions to places, but never transitions to transitions, nor places to places.
PNML:00004	Initial Marking	The set of initial place markings given with the net definition.
PNML:00005	Token	A data item associated with a place and chosen from the place's type.
PNML:00006	Arc	A directed edge of a net which may connect a place to a transition or a transition to a place. Normally represented by an arrow. Input Arc of a transition (or PT Arc) is an arc directed from a place to the transition. Output Arc of a transition (or TP Arc) is an arc directed from the transition to a place.
PNML:00007	Place	A node of a net, taken from the place kind, normally represented by an ellipse in the net graph. A place is typed. Whereas, an Input Place (of a transition) is a place connected to the transition by an input arc. While an Output Place (of a transition) is a place connected to the transition by an output arc.
PNML:00008	Transition	A node of a net, taken from the transition kind, and represented by a rectangle in the net graph.
PN4RO:00001	Failure Marking	A marking of a net which represent a state of failure of the represented system.
PN4RO:10002	Probability of a Marking	A rational value in a range [0,1] indicating the likelihood of a general marking
PN4RO:00002	Probability of Failure Marking	A rational value in a range [0,1] indicating the likelihood of a failure marking
PN4RO:00003	Component Arc	A directed edge of a net which connects a place to a transition or a transition to a place. Normally represented by an arrow and exclusively used to link PN elements representing an assembly station's component.
PN4RO:00004	PT Component Arc	A directed edge of a net which connects a place to a transition. Normally represented by an arrow and exclusively used to link PN elements representing an assembly station's component.
PN4RO:00005	TP Component Arc	A directed edge of a net which connects a transition to a place. Normally represented by an arrow and exclusively used to link PN elements representing an assembly station's component.
PN4RO:00006	System To Component Arc	A directed edge of a net which connects a transition to a place. Normally represented by an arrow exclusively used to link assembly station's components to the related assembly discrete phase.
PN4RO:00007	System Arc	A directed edge of a net which connects a place to a transition or a transition to a place. Normally represented by an arrow and exclusively used to link PN elements representing an assembly discrete phases.
PN4RO:00008	PT System Arc	A directed edge of a net which connects a place to a transition. Normally represented by an arrow and exclusively used to link PN elements representing an assembly discrete phases.
PN4RO:00009	TP System arc	A directed edge of a net which connects a transition to a place. Normally represented by an arrow and exclusively used to link PN elements representing an assembly discrete phases.
PN4RO:00010	Component To System Arc	A directed edge of a net which connects a transition to a place. Normally represented by an arrow exclusively used to link an assembly discrete phase to the related assembly station's components.
PN4RO:00011	System Place	A specialization of the simple Place used to represent phases of the assembly process.
PN4RO:00012	Component Place	A specialization of the simple Place used to represent conditions (or states) of a component.
PN4RO:00013	Failure State Component Place	A specialization of the Component Place used to represent a failure condition (or states) of a component.

³⁴ Definitions extracted/adapted from ISO/IEC 15909-1

PN4RO:00014	Working State Component Place	A specialization of the Component Place used to represent a working condition (or states) of a component
PN4RO:00015	System Transition	A specialization of the simple Transition used to represent a discrete activity of the assembly process.
PN4RO:00016	System Transition Firing Time	A feature of each System Transition, which is of timed nature.
PN4RO:00017	Component Transition	A specialization of the simple Transition used to represent a change of state of a specific component.
PN4RO:00018	Failure Component Transition	A specialization of the component Transition used to represent the failure of a specific component.
PN4RO:00019	Replace Component Transition	A specialization of the component Transition used to represent the replacement of a specific component.
PN4RO:00020	Module	A PN modelling element representing a component which can be either on its operational (working) or failure state.

OWL object properties between all the aforementioned classes have been, hence, defined (and partially illustrated in Figure 29 and Figure 30). Table 13 shows a complete list of the object properties. Here we use three different colours to indicate the properties either belonging to only to the Assembly Station domain (green), to the PN domain (blue), or spanning across these two domains (orange).

As for the OWL Classes, here the URIs are expressed in their *short* form, whereas the prefixes PNML, RAASO, and PN4RO stand for:

- PNML = “http://www.semanticweb.org/PNML/PNML_”
- PN4RO = “http://www.semanticweb.org/PN4RO/PN4RO_”
- RAASO = “http://www.semanticweb.org/RAASO/RAASO_”

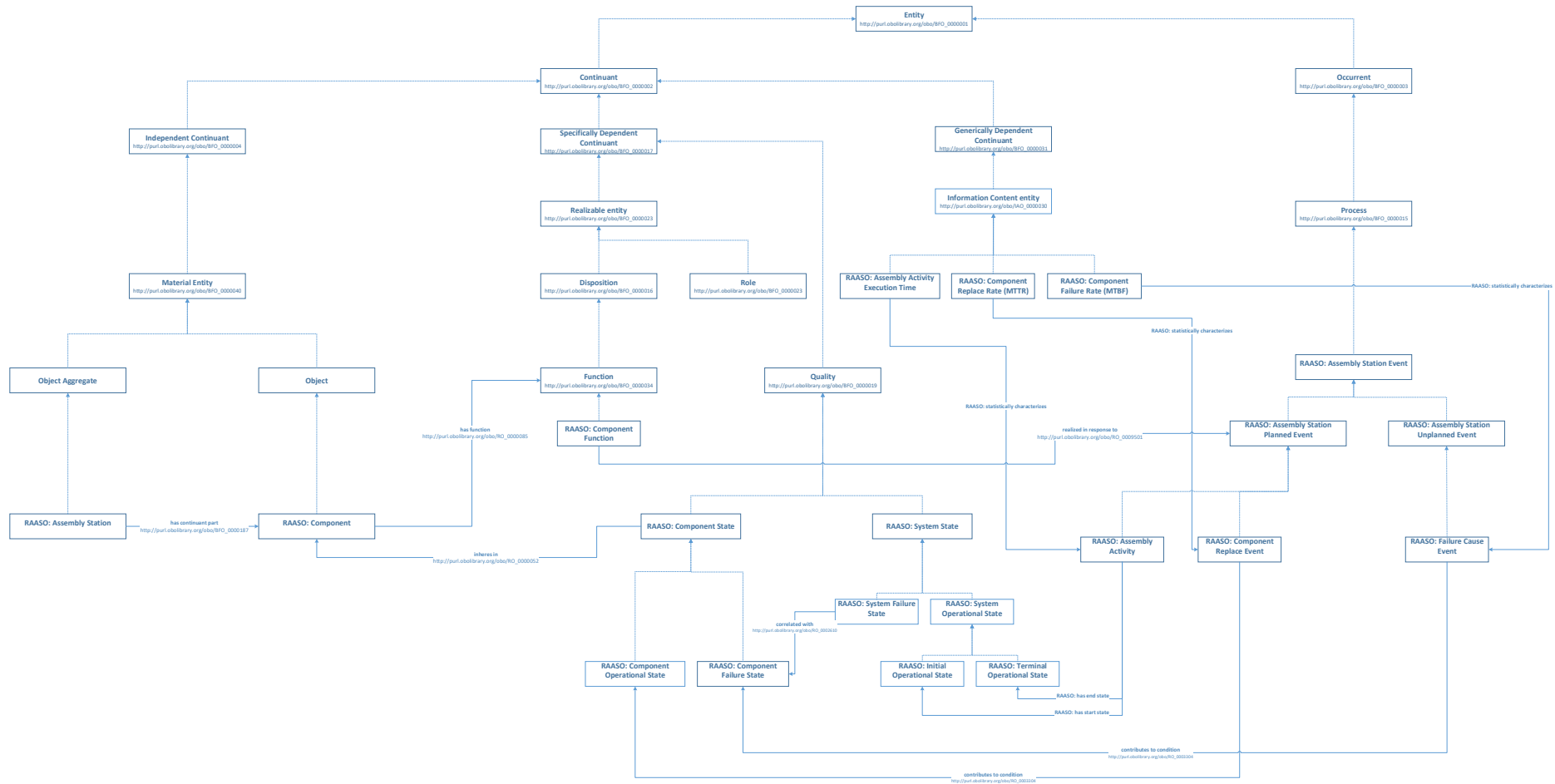


Figure 29 RAAS Ontology elements compliant to BFO

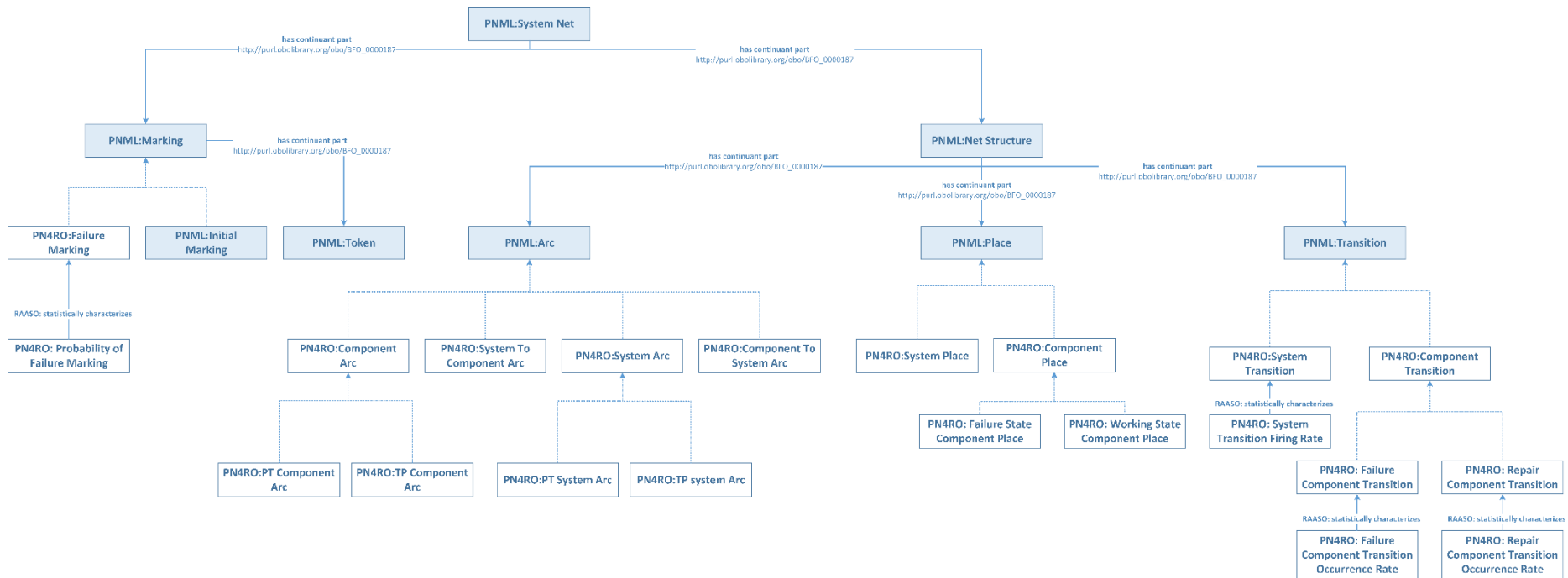


Figure 30 PN4R Ontology

Table 13 OWL object properties of the RAAS Ontology

URI	rdf:label	Domain	Range
RAASO:90001	hasEndState	Assembly Activity	Terminal Operational State
RAASO:90002	hasStartState	Assembly Activity	Initial Operational State
RAASO:90003	isInitialOperationalStateOf	Initial Operational State	Assembly Activity
RAASO:90004	isTerminalOperationalStateOf	Terminal Operational State	Assembly Activity
RAASO:90005	characterizesPNElement	-	-
RAASO:91005	characterizesSystemTransition	Assembly Activity	System Transition
RAASO:92005	characterizesSystemPlace	System Operational State	System Place
RAASO:90006	isCharacterizedBySystemElement	-	-
RAASO:91006	isCharacterizedByAssemblyActivity	System Transition	Assembly Activity
RAASO:92006	isCharacterizedBySystemOperationalState	System Place	System Operational State
RAASO:90007	failsDueToFailureEvent	Component	Failure Cause Event
RAASO:90008	requiresComponentReplacing	Failure Cause Event	Component Replace Event
RAASO:90009	isCharacterizedByFailureCauseEvent	-	Failure Cause Event
RAASO:90010	isCharacterizedByComponentReplaceEvent	-	Component Replace Event
RAASO:90011	isComposedByFCT	Module	Failure Component Transition
RAASO:90012	isComposedByFSCP	Module	Failure State Component Place
RAASO:90013	isComposedByPTCA	Module	PT Component Arc
RAASO:90014	isComposedByRCT	Module	Replace Component Transition
RAASO:90015	isComposedByTPCA	Module	TP Component Arc
RAASO:90016	isComposedByWSCP	Module	Working State Component Place
RAASO:90017	characterizesComponent	Module	Component
RAASO:90018	isInvolvedIn	Component	Assembly Activity
RAASO:90019	isFailureCauseOf	Failure Cause Event	Component
RAASO:90020	isRequiredBecauseOf	Component Replace Event	Failure Cause Event
PN4RO:90001	CSArcHasRCT	Component To System Arc	Replace Component Transition
PN4RO:90002	CSArcHasSystemPlace	Component To System Arc	System Place
PN4RO:90003	PTArcHasFCT	PT Component Arc	Failure Component Transition
PN4RO:90004	PTArcHasFSCP	PT Component Arc	Failure State Component Place
PN4RO:90005	PTArcHasRCT	PT Component Arc	Replace Component Transition
PN4RO:90006	PTArcHasWSCP	PT Component Arc	Working State Component Place
PN4RO:90007	SCArcHasFCT	System To Component Arc	Failure Component Transition
PN4RO:90008	SCArcHasSystemPlace	System To Component Arc	System Place
PN4RO:90009	SystemPlaceHasDownstreamTransition	System Place	System Transition
PN4RO:90010	SystemPlaceHasPTArc	System Place	PT System Arc
PN4RO:90011	SystemPlaceHasTPArc	System Place	TP System Arc
PN4RO:90012	SystemPlaceHasUpstreamTransition	System Place	System Transition
PN4RO:90013	SystemTransitionHasDownstreamPlace	System Transition	System Place
PN4RO:90014	SystemTransitionHasPTArc	System Transition	PT System Arc
PN4RO:90015	SystemTransitionHasTPArc	System Transition	TP System Arc
PN4RO:90016	SystemTransitionHasUpstreamPlace	System Transition	System Place
PN4RO:90017	TPArcHasFCT	TP Component Arc	Failure Component Transition
PN4RO:90018	TPArcHasFSCP	TP Component Arc	Failure State Component Place
PN4RO:90019	TPArcHasRCT	TP Component Arc	Replace Component Transition
PN4RO:90020	TPArcHasWSCP	TP Component Arc	Working State Component Place

Following the definition of OWL object properties, the ontological models have been further enriched by adding OWL data properties describing the relations between OWL classes and type of data values e.g. literal, string, double, etc. (Table 14).

Table 14 Data properties of Automated Assembly System Ontology

URI	Data property	Domain	Range
RAASO:50001	hasMTBF	FailureCauseEvent	xsd:double
RAASO:50002	hasMTTR	ComponentReplaceEvent	xsd:double
PN4RO:50001	hasLambda	FailureComponentTransition	xsd:double
PN4RO:50002	hasMu	ReplaceComponentTransition	xsd:double
RAASO:50005	hasActivityExecutionTime	AssemblyActivity	xsd:double
PN4RO:50005	hasFiringRate	Transition	xsd:double

As a result, the classes of those two domain ontologies (RAASO and PN4RO) are linked each other in a way to enable the definition of rules which leverage the Hermit Reasoner³⁵ implemented within Protégé to drive (through inference) the translation of assembly operations, components, failures, and repairs into modelling primitives of a Petri Net, thus, allowing the simulation-based reliability assessment of such an assembly process (more details in the next section). Figure 31 show an example of the interlinks between the Assembly Activities, the System Operational States (discrete stages of the assembly process in which the assembly station is operational), the System Transitions (PN transition representing an assembly activity characterized by a certain firing time), and the System Places (PN place representing either pre- or post- conditions of a certain transition).

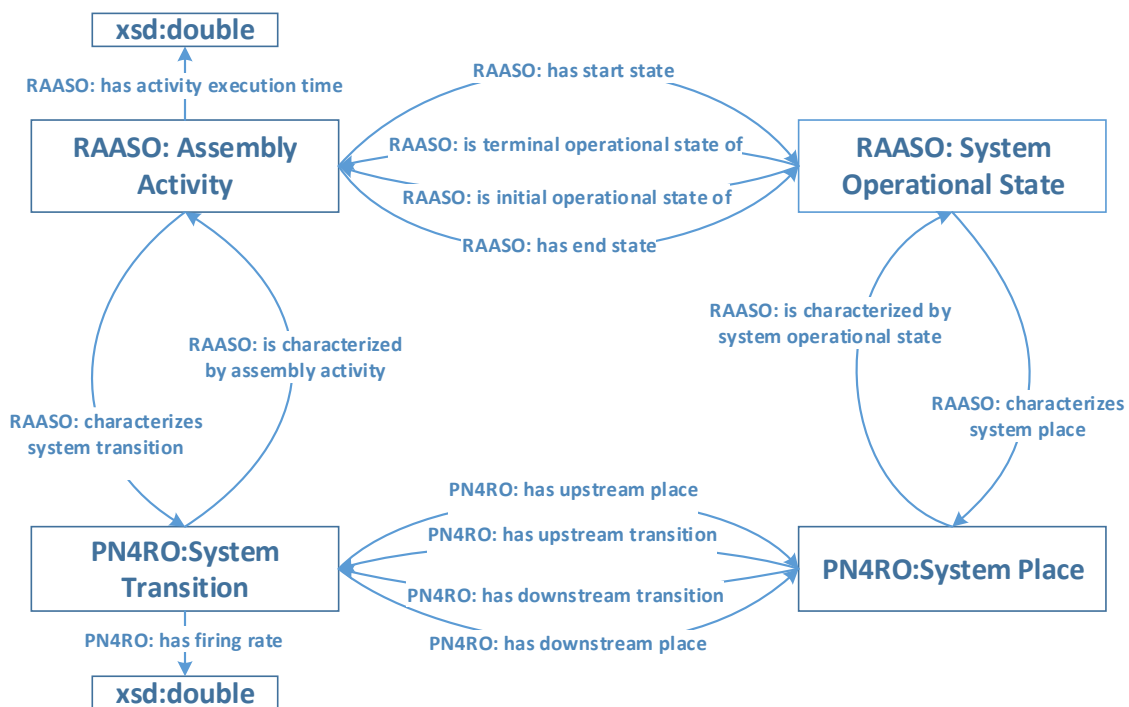


Figure 31 Example of the network of object properties

³⁵ Hermit reasoner <http://www.hermit-reasoner.com/> (Accessed on 25 Feb 2019)

Creation of the OWL individuals: from RAASO to PN4RO elements

The initial scenario is described in Table 15. The classes being initially populated are:

- [RAASO:00018] Assembly Activity
- [RAASO:00009] System Operational State
- [RAASO:00002] Component
- [RAASO:00020] Failure Cause Event
- [RAASO:00019] Component Replace Event

Table 15 Individuals on the initial scenario

Assembly Activity	System Operational State	Component	Component Failure Event	Component Replace Event
AA1	SOS1	CompFF	FailureOfFF	ReplaceFF
AA2	SOS2	CompFC	FailureOfFC	ReplaceFC
AA3	SOS3	CompFW	FailureOfFW	ReplaceFW
AA4	SOS4	CompRF	FailureOfRF	ReplaceRF
AA5	SOS5	CompUC	FailureOfUC	ReplaceUC
AA6	SOS6	CompMI	FailureOfMI	ReplaceMI
AA7	SOS7	CompBI	FailureOfBI	ReplaceBI
AA8	SOS8	CompBR	FailureOfBR	ReplaceBR
AA9	SOS9	CompBF	FailureOfBF	ReplaceBF

The sequence in which the stages of this process is carried out is shown in Figure 32. The order is relevant as it reflects the – discretized – assembly process, where each activity enables a subset of elements, which may fail.

As the elements of the Assembly Station instantiated according to the RAASO, elements of the PN4RO are therefore created by exploiting the interlinks between those two domain specific ontologies.

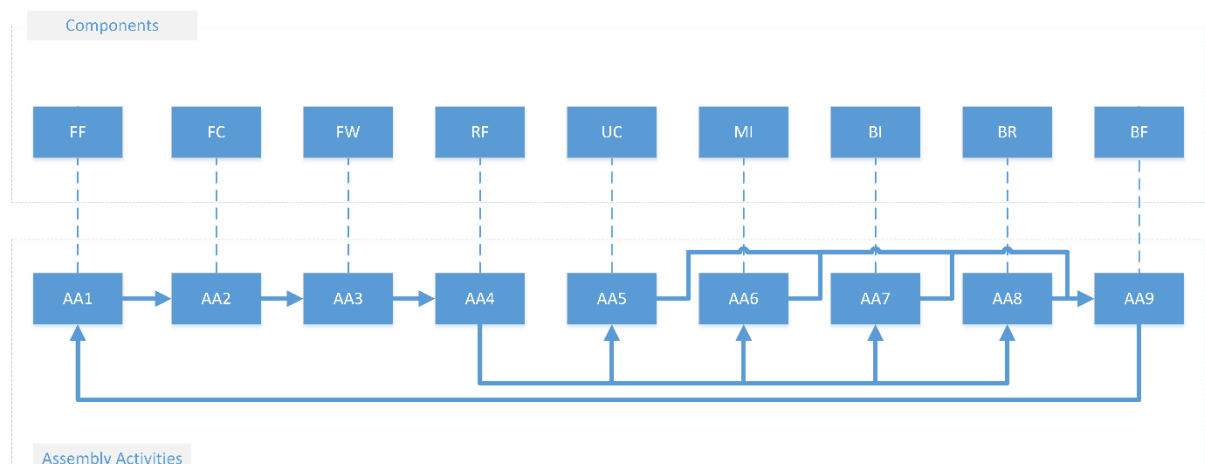


Figure 32 Sequence of Assembly Activities and Involved Components

A few clarifications regarding the statistical characterization of the PN modelling elements as well as the adaptation of the original information regarding the occurrence of failures and replacements are given below:

- Component Replace and Failure occurrence rates are statistically independent
- Switchover time from active to failed units and vice versa is negligible
- No delay time due to component replacement
- Each component has one failure mode.
- The first four activities are represented with the sequential pattern
- The fifth, sixth, seventh and eighth activities are done in parallel: to represent this behaviour we exploited the parallel and then the synchronisation patterns
- The ninth activity is linked with the first one through the repetitive pattern to model a continuative behaviour of the process

The case study addresses two non-trivial modelling issues, that is the concurrency execution (or liveness) of assembly activities and the failure events. Furthermore, the latter requires the replication of certain modelling patterns due to multiple functions - thus failure modes - inhering in the same component (more details around modelling patterns in the Annex). The data values used to simulate the mode are presented below. In particular, the initial scenario is characterized by the following values: the duration time for each assembly activity (Table 16), the failure event occurrence rates (Table 17), and the component replace event occurrence rates (Table 18).

Table 16 Assembly Activity Duration Time

Assembly Activity	Execution Time [min]
AA1	1.15385
AA2	1.15385
AA3	1.11111
AA4	1.11111
AA5	1.22449
AA6	1.22449
AA7	1.22449
AA8	1.22449
AA9	1.17647

Table 17 Mean Time Between Failures occurrence

Component Failure Event	MTBF [min]
FailureOfBF	344.82759
FailureOfBR	203.25203
FailureOfMI	819.67213
FailureOfBI	16666.66667
FailureOfFF	526.31579
FailureOfFW	813.00813
FailureOfFC	813.00813
FailureOfRF	309.59752
FailureOfUC	163.66612

Table 18 Mean Time To Replace occurrence

Component Replace Event	MTTR [min]
ReplaceBF	17.58396
ReplaceBR	17.71793
ReplaceMI	17.44592
ReplaceBI	11.89910
ReplaceFF	17.47641
ReplaceFW	17.33403
ReplaceFC	17.76199
ReplaceRF	17.60253
ReplaceUC	17.72421

Sixteen (16) steps for the creation of OWL elements (Figure 33) stemming from SWRL-based rules (more details in the Annex) that leverage the interlinks between the RAAS and the PN4R ontologies are hereafter described:

- **Creation of the Component Layer**

1. Instantiation of PN Modules, representing the behaviour of the component
2. Instantiation of PN Places, representing failure or working states of a component
3. Instantiation of PN Transitions, representing component failure events, and characterization of their (stochastic) firing time
4. Instantiation of PN Transitions, representing component replace events, and characterization of their (stochastic) firing time
5. Instantiation of PN PT Arcs, linking PN places representing the failure state of a component to PN transition representing its replace event
6. Instantiation of PN PT Arcs, linking PN places representing the working state of a component to PN transition representing its failure event
7. Instantiation of PN TP Arcs, linking PN transitions representing the failure event of a component to PN place representing its failure state
8. Instantiation of PN TP Arcs, linking PN transitions representing the replace event of a component to PN place representing its working state

- **Creation of the System Layer**

9. Instantiation of PN Transitions, representing the assembly activities
10. Instantiation of PN Places, representing the systems states before and after a certain assembly activity
11. Characterization of PN Transitions, linking them to their downstream places
12. Characterization of PN Transitions, linking them to their upstream places
13. Instantiation and characterization of PN PT Arcs, linking PN places representing the system state (discrete stage of the assembly process) to PN transition representing its next assembly activity
14. Instantiation and characterization of PN TP Arcs, linking PN transitions representing the assembly activity to PN places representing its terminal state (discrete stage of the assembly process after the aforementioned assembly activity).

- **Creation of cross-layers links**

15. Instantiation of TP Arcs, linking PN transitions representing the component replace event to the PN place representing the stage of the assembly process on which it may fails
16. Instantiation of PT Arcs, linking PN places representing a certain system state on which a component may fail and a PN transition representing the failure event of that component.

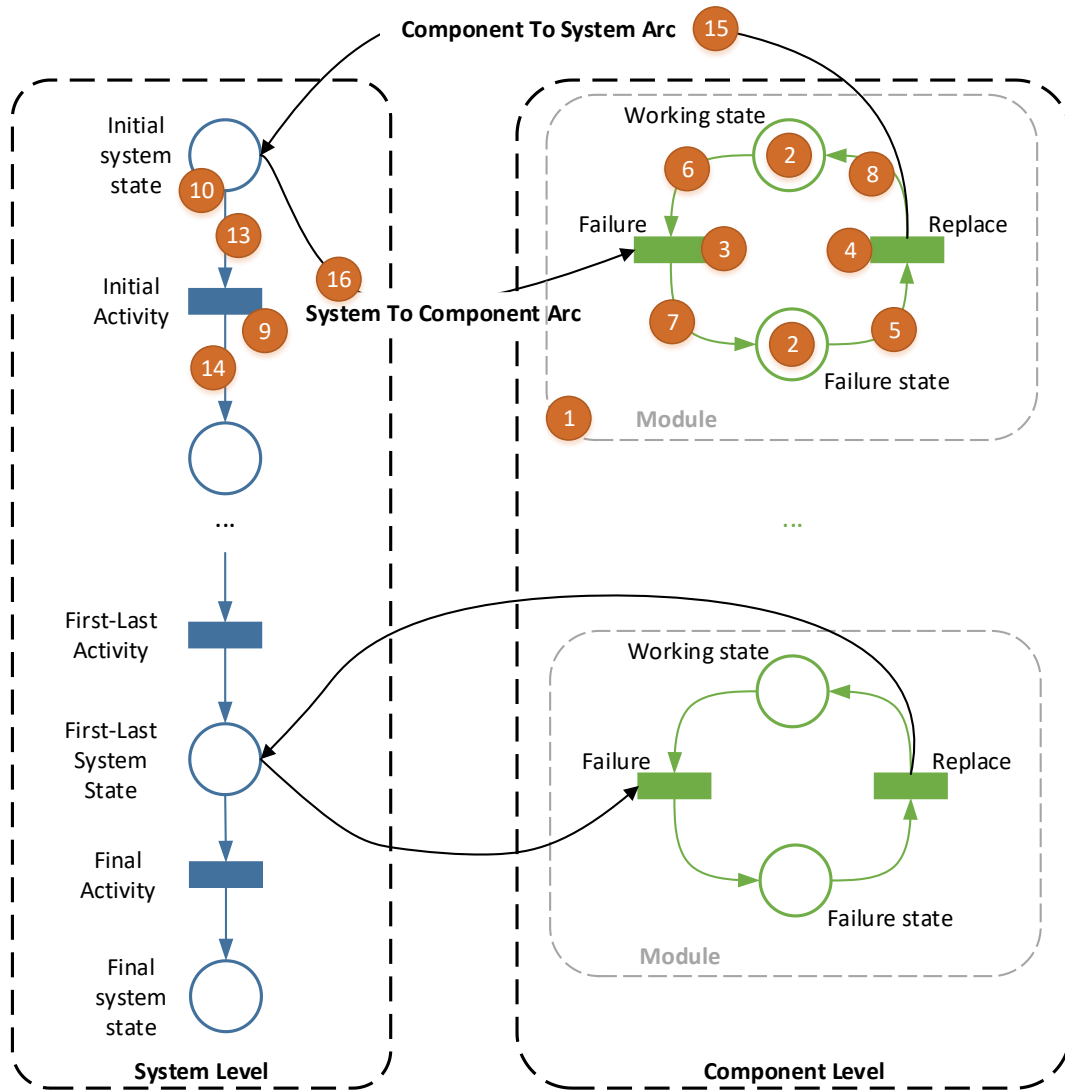


Figure 33 Stepwise description of the PN creation

Petri Net simulation

Modelling and analysis of the behaviour of dynamic systems – for example, an assembly station, is of wide interest in various fields of science and engineering. Common to realistic models of time dynamic systems are the complexity, very often prohibiting numerical or analytical evaluation. Simulation remains the only tractable evaluation methodology [114], this is why we have chosen this one to extract performance analysis from the case study under investigation. The PN model presented below represents a digital twin of the of the physical system, which can be eventually considered as a specification of states and events (planned and unplanned). Performing a simulation thus means mimicking the occurrence of events as they involve in time and recognizing their effects as represented by states. There are two kinds of simulations: continuous and discrete. In a continuous simulation, state changes occur continuously in time, while in a discrete simulation, as our case study is, the occurrence of an event is instantaneous and fixed to a selected point in time.

Because of the discrete nature of our case study, we refer to a Discrete Event Simulation, often called with the acronym DES. This can be discerned into three main typologies: discrete event simulation, continuous simulation and Monte Carlo simulation [115]. In particular, we have chosen this latter typology: Monte Carlo simulation is used to study complex systems when the analysis is difficult or impossible through analytical methods [116]. This latter, in its basic form, has the goal to compute $\mu = E[X]$ for some random variable X . It generates n independent copies of X , $(X_i, 1 \leq i \leq n)$. Then,

$\bar{X}_n \rightarrow \frac{1}{n} \sum_{i=1}^n X_i$ is an approximation (an estimation) of μ

$\bar{X}_n \rightarrow \mu$ with probability 1, as $n \rightarrow \infty$ (Strong Law of Large Numbers)

We can evaluate the accuracy of \bar{X}_n by means of the Central Limit Theorem, which allows us to build the following confidence interval [117]:

$$CI = \left(\bar{X}_n - \frac{c \alpha \sigma}{\sqrt{n}}, \bar{X}_n + \frac{c \alpha \sigma}{\sqrt{n}} \right)$$

Equation 8 Confidence interval

For complex Petri net models, simulation is a way to check the system properties: it can supplement mathematical/analytical modelling approaches in the analysis of real systems [118]. The idea is that, by using the execution algorithm to run the net, it is possible to feign the behaviour of the model. However, simulation is usually an expensive and time-consuming technique. Despite this, Petri net simulation is indeed a convenient and straightforward yet effective approach for engineers to validate the desired properties of a discrete event system [119].

Starting from the RDF instantiation of a – stochastic – Petri net model resembling the elements of an Assembly Station along with its operational and reliability parameters (i.e. Assembly Activity Duration, MTBF, MTTR), it is possible to use an external tool to load simulate the PN model. As stated earlier, on ontology does not provide any simulation tool but it could rather support the latter for enhanced – or enriched – formalization of the simulation model. Figure 34 illustrates the model format transformation process: the RDF/XML-based file of the Stochastic Petri Net (SPN) model is translated into another XML-based file that can be read and analysed by the specific PN application tool. The W3C community supports XSL Transformations³⁶, in particular, XSLT is a W3C Recommendation for defining the transformation of XML documents in other XML documents.

³⁶ W3C recommendation for XML transformations <https://www.w3.org/standards/xml/transformation> [Last access 7 Jan 2019]

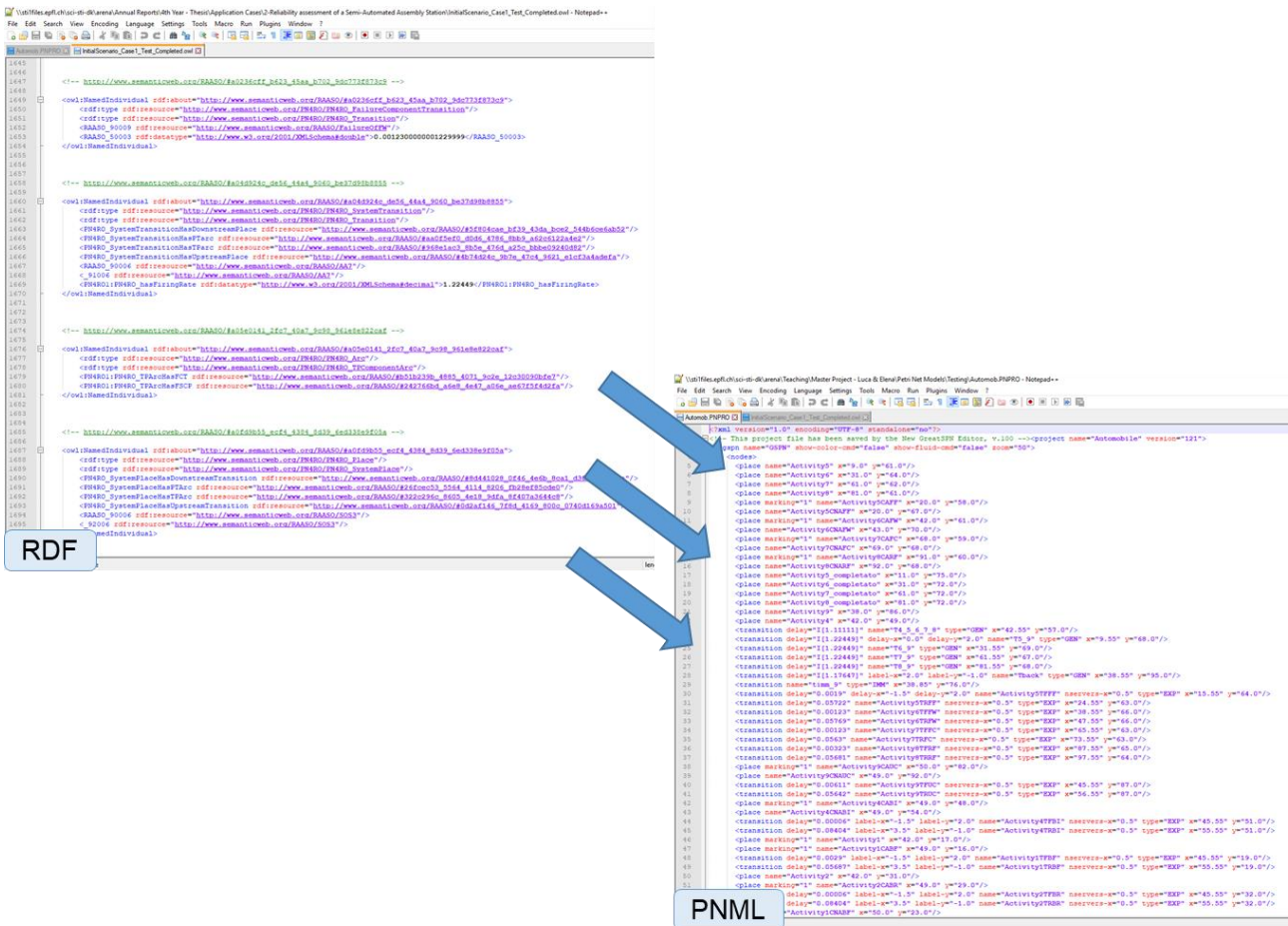


Figure 34 From RDF to PNML: (left side) PN model resulting from the SWRL rules; (right side) PN model formalized according to a tool specific XML data structure

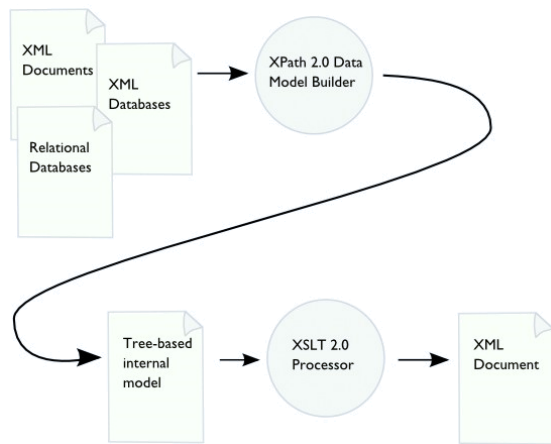


Figure 35 W3C: XSL Transformation process

The tool used to simulate the SPN model is GreatSPN [120], [121]. GreatSPN2.0 is a software package for modelling, validation, and performance evaluation of distributed systems using Generalized Stochastic Petri Nets and their coloured extension: Stochastic Well-formed Nets. The tool provides a friendly framework to experiment with timed PN-based modelling techniques.

Figure 36 illustrates a partial view of the SPN model as loaded and ready to be analysed on GreatSPN, which has been obtained through SWRL-based instantiation described in the previous section. The simulation engine of this software tool leverage the Monte Carlo Simulation technique to extract some performance indices, e.g. the average number of tokens in a place and the transition

throughputs. Given the fact that the net has been built in a way that each place can contain at most one token, this measure reflects the probability of having a token in that place (more details are given below).

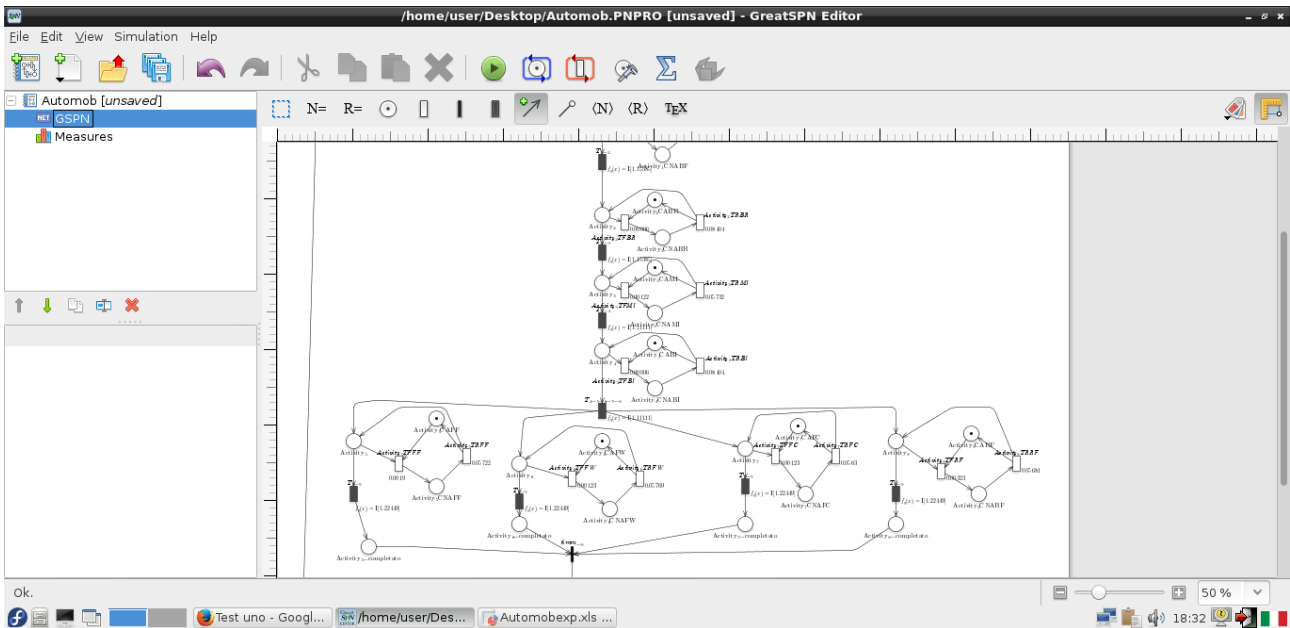


Figure 36 Screenshot of the model used for the example of testing

The simulation engine requires the setting of certain parameters, such as Confidence Interval, Approximation, batch characteristics. The following parameters have been, therefore, set in order to run the Basic GSPN (no colours) simulation by means of the Solver GreatSPN [122]:

- Confidence Interval: 95%
- Approximation: 20%
- Set batch: duration
- Set batch length to min max: 31000

Simulation results are exported in an Excel sheet, as shown in the figure below.

1	Average token count in places		
2	Place	Average token count	
3	Activity1	0.1579041170865072 <= 0.1579942816987993 <= 0.158084463110913	
4	Activity1CABF	0.9919144519311494 <= 0.9921285328526862 <= 0.9923426137742231	
5	Activity1CNABF	0.0076573862257478 <= 0.0078714671473149 <= 0.0080855480688819	
6	Activity2	0.1579005836794071 <= 0.1579908390665407 <= 0.1580810944536742	
7	Activity2CABR	0.9998768626033349 <= 0.9998966099367805 <= 0.9999163572702262	
8	Activity2CNABR	8.36427297662E-5 <= 1.033900632191E-4 <= 1.23137396672E-4	
9	Activity3	0.152050053995549 <= 0.1521368569929808 <= 0.1522236599904126	
10	Activity3CAMI	0.9967174063780934 <= 0.9968620099731021 <= 0.9970066135681108	
11	Activity3CNAMI	0.0029933864318253 <= 0.0031379900268981 <= 0.0032825936219709	
12	Activity4	0.1520475645948915 <= 0.1521344401187077 <= 0.1522213156425239	
13	Activity4CABI	0.9998942007633544 <= 0.9999117412575964 <= 0.9999292817518384	
14	Activity4CNABI	7.07182481114E-5 <= 8.82587424049E-5 <= 1.057992366983E-4	
15	Activity5	0.1675607733379116 <= 0.167656428681392 <= 0.1677520840248725	
16	Activity5CAFF	0.9941071414430588 <= 0.9942917074757515 <= 0.9944762735084441	
17	Activity5CNAFF	0.0055237264914963 <= 0.0057082925242496 <= 0.0058928585570028	
18	Activity5_completato	0.0162699068816524 <= 0.016586660174161 <= 0.0169034134666697	
19	Activity6	0.1675607648670294 <= 0.1676564263866956 <= 0.1677520879063617	
20	Activity6CAFW	0.9962429703136108 <= 0.9963969208474062 <= 0.9965508713812016	
21	Activity6CNAFW	0.0034491286188334 <= 0.003603079152595 <= 0.0037570296863566	
22	Activity6_completato	0.0183631297859089 <= 0.018691875840512 <= 0.0190206218951151	
23	Activity7	0.1675608355770063 <= 0.167656495804299 <= 0.1677521560315917	

Figure 37 Partial 'Average number of token per place' results of the example simulation of testing

The solver has the capability to compute the *average number of tokens per place* and the *transition throughput*. This is why, similarly to the approach adopted for the term *Module*, the RAAS Ontology contains also those two owl classes, since they are strictly related to the scope of such an ontology.

Table 19 Two extra terms in RAASO

URI	Term	Definitions
RAASO:00021	Average Number of Tokens Per PN Place	An Information Content Entity that statistically characterizes the probability of a place to be marked
RAASO:00022	Average Number of Tokens in FSCP	An Information Content Entity that statistically characterizes the probability of a FSCP to be marked
RAASO:00024	PN Transition Throughput	An Information Content Entity that statistically characterizes the occurrence rate of a PN transition

The net is built in a way that each place contains at most one token. This is why the probability to have a certain number of tokens in a place is equivalent to the probability of that place to be marked. Thus, giving specific meaning to each place (i.e. a specific state of the system or the component), we can obtain the probability that such state happens. By analysing only the FSCP (Failure State Component Place) places, it is possible to conduct a reliability analysis, extracting the information that a certain typology of failure for a certain component happens with a determined probability.

Much the same applies to transitions throughput. In particular, focusing on the last transition (T_{back} , which represents the activity AA9), if the event related to this transition occurs, it means that the last assembly activity is completed and a new cycle can start. Knowing the duration of the simulated time, it is therefore possible to deduce the number of simulation cycles.

In order to calculate it, the procedure to be applied is presented in the following. The transition throughput is computed by the simulator in $\left[\frac{Events}{Unit\ of\ time}\right]$, therefore, the total number of cycles can be computed as:

$$Total\ number\ of\ cycles\ [Events] = Transition\ Throughput\ \left[\frac{Events}{Unit\ of\ time}\right] * Total\ simulated\ time\ [Unit\ of\ time]$$

Equation 9 PN model simulation: Total number of cycles

In our case:

$$Total\ number\ of\ cycles = 0.13693049549648 * 31000 = 4244.845 \cong 4245$$

21	T1_2	0.136927743111919 <= 0.1369277503418728 <= 0.13692775718265	
22	T2_3	0.1369249354156971 <= 0.1369249427973872 <= 0.1369249501790773	
23	T3_4	0.1369232508595215 <= 0.1369232582706957 <= 0.1369232656818699	
24	T4_5_6_7_8	0.1369211297572379 <= 0.1369211370148621 <= 0.1369211442724862	
25	T5_9	0.1369231888165149 <= 0.1369231958808184 <= 0.1369232029451219	
26	T6_9	0.1369192582364013 <= 0.1369192653185386 <= 0.1369192724006759	
27	T7_9	0.1369163880279451 <= 0.1369163953841757 <= 0.1369164027404062	
28	T8_9	0.1369192582383639 <= 0.1369192653185387 <= 0.1369192723987135	
29	Tback	0.1369304884298321 <= 0.1369304954964807 <= 0.1369305025631294	

Tool statistics Place averages **Transition throughputs**

Figure 38 Partial 'Transition throughput' results of the example simulation of testing

Semantics-driven analysis of the simulation results

In this section, it is explained how the results of the PN simulation can be further enriched by the knowledge bases built upon the ontological models presented above. As shown in Figure 39, the two main results that can be reached through PN are the probability that a failure state component place has a token and the throughputs of the system. These can be RDFized using the OWL classes PN4RO:10002, PN4RO:00002, and then, translated into elements of the OWL classes RAASO:00022 and RAASO:00024, thus, enriching the current knowledge base with simulation results that can be used either for design and general evaluation purposes. Eventually, it is possible to SPARQL query the above-mentioned class, and aggregate the semantically-enriched simulation results to draw conclusions around other system reliability aspect, i.e. the probability of failure for a specific failure cause event, the number of items to keep the stock of each component, the overall probability of system failure or system availability.

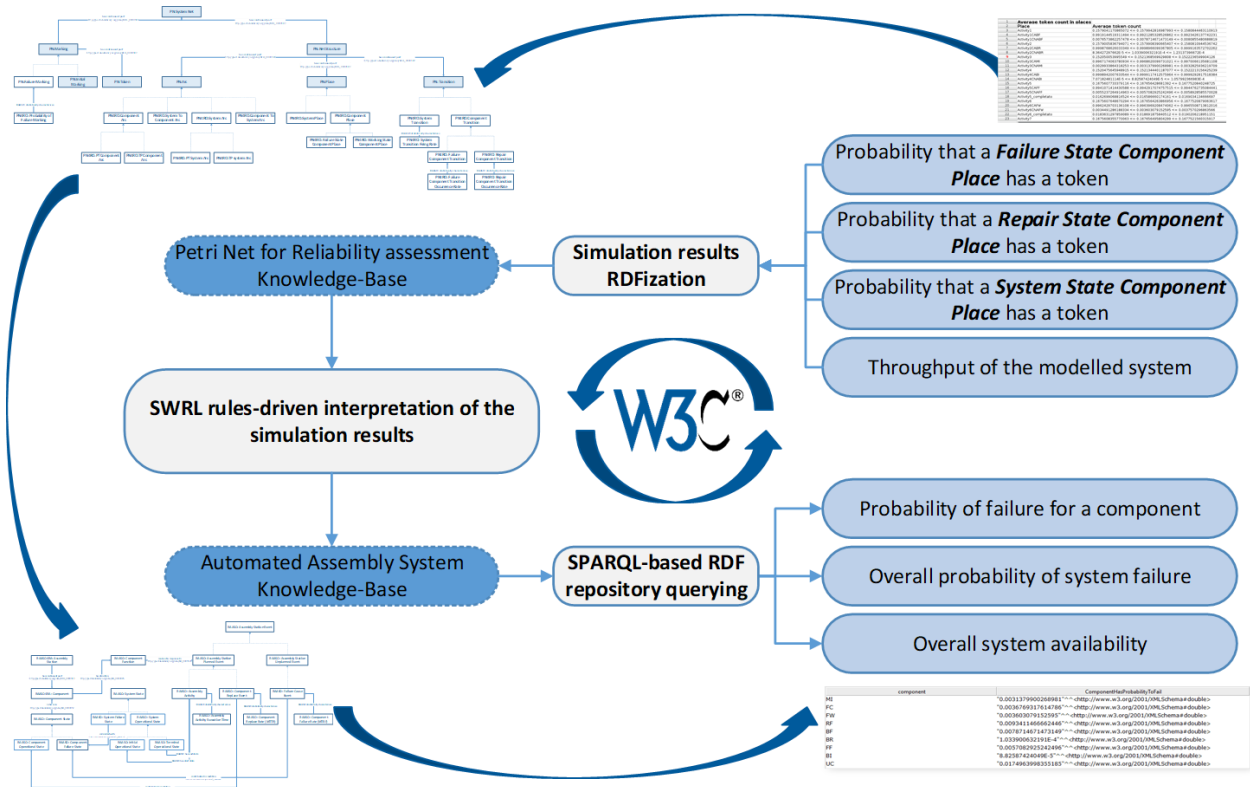


Figure 39 Semantically enriched results data analysis

Starting from the probability that a failure could take place in a *Module*, it is possible to calculate probabilities that are more meaningful.

The terminology of reference is:

- Component typology: $i = 1, 2, \dots, I$
- Cause of failure: $j = 1, 2, \dots, J$
- Failure typology: $k = 1, 2, \dots, K$
- P_{ijk} : probability that a component of typology i fails because of the cause j through a failure type k

The probability that a component of a specific type fails

It is given by the sum of the probabilities that a failure happens for a specific component.

$$\sum_j^J \sum_k^K P_{ijk}$$

Equation 10 Probability that a component of a specific type fails

In the specific example, there is no need to perform this summation since each component appears just once in the system. Anyhow, we preferred to propose the most general query, in order to be able to applicate it also in the systems where the components are replicated. In the query below, GROUP BY allows collecting the results by the specified class, which, in this case is Component.

```

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX p: <http://www.semanticweb.org/arena/ontologies/2016/8/untitled-ontology-433#>

SELECT ?component (SUM(?prob) AS ?ComponentHasProbabilityToFail)

WHERE

{ ?m p:ModuleHasProbabilityThatAFailureHappens ?prob.

?m p:ModuleCharacterizesComponent?component}

GROUP BY ?component
    
```

The result of the query, as shown by the SPARQL tab in Protégé in Figure 40, consists in the probability that each component (listed on the left) has to fail. This information could be important for two reasons. The first is that the management could find the components that have more probability of a breakdown, being able to monitor them and address to them actions such as preventive maintenance (for instance, in the example, the component “BI”). The second is that it will be used to reach more meaningful results through calculation presented in the next steps.

component	ComponentHasProbabilityToFail
MI	"0.0031379900268981"^^<http://www.w3.org/2001/XMLSchema#double>
FC	"0.0036769317614786"^^<http://www.w3.org/2001/XMLSchema#double>
FW	"0.003603079152595"^^<http://www.w3.org/2001/XMLSchema#double>
RF	"0.0093411466662446"^^<http://www.w3.org/2001/XMLSchema#double>
BF	"0.0078714671473149"^^<http://www.w3.org/2001/XMLSchema#double>
BR	"1.033900632191E-4"^^<http://www.w3.org/2001/XMLSchema#double>
FF	"0.0057082925242496"^^<http://www.w3.org/2001/XMLSchema#double>
BI	"8.82587424049E-5"^^<http://www.w3.org/2001/XMLSchema#double>
UC	"0.0174963998355185"^^<http://www.w3.org/2001/XMLSchema#double>

Figure 40 Probability of failure for each component shown through Protégé SPARQL Tab

Availability of the system

The overall probability of having a failure within the entire process is given by the sum of the overall probabilities:

$$\sum_i^I \sum_k^K \sum_j^J P_{ijk}$$

Equation 11 System availability

```

SELECT (SUM(?prob) AS ?OverallProbabilityOfSystemFailure)

WHERE

{?m p:ModuleHasProbabilityThatAFailureHappens ?prob.}
    
```

OverallProbabilityOfSystemFailure
 "0.0510269559199233"^^<http://www.w3.org/2001/XMLSchema#double>

Figure 41 System Failure probability shown through Protégé SPARQL Tab

From which it is possible to compute the availability of the system (A) as follows:

$$A = 1 - \sum_i^I \sum_k^K \sum_j^J P_{ijk} = 1 - 0.0510269559199233 = 0.9489730440800767$$

```
SELECT (1-(SUM(?prob)) AS ?SystemAvailability)
WHERE
{ ?m p:ModuleHasProbabilityThatAFailureHappens ?prob. }
```

The computation of the availability, performed in SPARQL, gives as result 94,9% (Figure 42). This value is a measure of the overall availability of the system, that the management can use as a reference, in the process of improvement.

SystemAvailability
 "0.9489730440800767"^^<http://www.w3.org/2001/XMLSchema#double>

Figure 42 System availability evaluation shown through Protégé SPARQL Tab

Number of failures of each component in the related time period

The number of failures of each component can be computed as the product between the failure component transition throughput obtained by the simulation of the Petri net model and the simulated time. In this particular case, the simulated time is equal to 31000 time units. Eventually, using the GROUP BY, it is possible to sum all the values obtained by organizing them by type of component.

```
SELECT ?component (SUM(?tp)*31000 AS ?ComponentHasNumberOfFailures)
WHERE { ?m p:ModuleCharacterizesComponent?component.
?m p:ModuleIsComposedByFCT ?fct.
?fct p:FailureComponentTransitionHasTP ?tp. }
GROUP BY ?component
```

Figure 43 shows, for each component (on the left column), the associated number of failures (on the right column), in the considered time units. This calculation is presented since the calculation of the number of items to keep in stock of each component exploit this result. It is also interesting to notice that some components will fail a number of times extremely higher with respect to others (for instance component UC with respect to component BI)

component	ComponentHasNumberOfFailures
BI	"0.2572334647527"^^<http://www.w3.org/2001/XMLSchema#double>
BR	"0.31525605078740004"^^<http://www.w3.org/2001/XMLSchema#double>
BF	"14.0395317341019"^^<http://www.w3.org/2001/XMLSchema#double>
MI	"5.6726748279552"^^<http://www.w3.org/2001/XMLSchema#double>
FF	"10.0146983495204"^^<http://www.w3.org/2001/XMLSchema#double>
FW	"6.4869251186342"^^<http://www.w3.org/2001/XMLSchema#double>
FC	"6.3960230671818"^^<http://www.w3.org/2001/XMLSchema#double>
RF	"16.6679548814549"^^<http://www.w3.org/2001/XMLSchema#double>
UC	"30.6978161845505"^^<http://www.w3.org/2001/XMLSchema#double>

Figure 43 Number of failures for each component shown through Protégé SPARQL Tab

Total repairing hours for the system divided by the category of component:

The components of this example are divided into Mechanic and Pneumatic; hence, there could be two different categories of maintainer to accomplish the different replacements. To achieve this outcome it is just needed to consider the total repairing hours for the system and apply the GROUP BY according to the typology of components.

```

SELECT ?type (SUM(?tp*31000*?mttr) AS ?HoursOfRepair)

WHERE {?type rdfs:subClassOf p:Component.

?component a ?type.

?m p:ModuleCharacterizesComponent ?component.

?m p:ModuleIsComposedByFCT ?fct.

?fct p:FailureComponentTransitionHasTP ?tp.

?m p:ModuleHasProbabilityThatAFailureHappens ?prob.

?f p:FailureRequiresReplacement ?r.

?r p:ReplacementHasMTTR ?mttr.}

GROUP BY ?type
    
```

Figure 44 shows the overall amount of hours needed to perform repairing activities for each type of component. This is particularly useful for the management that can forecast the amount of time in which each specific type of maintenance workers will be required. In this case, it is also noteworthy that the difference of the repairing times between the two typology of component is wide.

type	HoursOfRepair
Pneumatic	"711.1457020346813"^^<http://www.w3.org/2001/XMLSchema#double>
Mechanical	"2463.3116141506935"^^<http://www.w3.org/2001/XMLSchema#double>

Figure 44 Total repairing hours grouped by component category shown through Protégé SPARQL Tab

6.2.2 Walkthrough implementation for a real case scenario: Powertrain Systems

The modelling and analysis approaches presented in Section 6.2.1 are hereby extended and applied to another real application scenario shaped around the needs and requirements of an Italian company working in the automotive domain. Due to the big amount of information analysed and the higher complexity of this use case, particular emphasis is given on the data collection and filtering phase. No further explanation is given regarding the overall methodology, which has been thoroughly described in the previous section.

Data collection and filtering

In this case study, we start with the analysis of the available information. The documents provided by the company are the sequence of operations, the FMECA analysis, and the ledgers for each typology of components (mechanicals, electrics and pneumatics). The so-called “cyclogram” contains the sequence of the activities that are performed by the assembly station. For each activity, the documents indicate the component’s reference identifier and the duration of the activity. Then, the FMECA thoroughly described all the possible failure modes related to each of the components. For each failure, then, we get the following details: the possible causes, the part that must be replaced to fix the failure, the MTBF in years and the MTRR in hours. Then, in each ledger the different typologies of components (mechanicals, electrics and pneumatics) are divided according to the unit (and subunits) they belong to and each part is classified with the code A, B or C (Figure 45). Components of class A are those that once they fail, they cause the stopping of the system. Components of class B could lead to severe consequences on the production process quality but do not stop the system. Finally, when components of class C fail, they do not generate any effects on the production process.

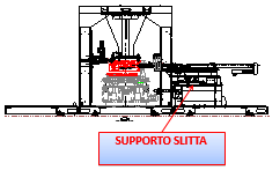



Equipment	GROUP	SUB-GROUP	Number	COMPONENT				Class A B C
				photo	name	SAP code	Location in the warehouse	
	1 SUPPORTO SLITTA WMPMA0000016040	1.1 SUPPORTO SLITTA	1.1.1		SERVOMOTORE SINCRONO SIEMENS 1FK7032-2AKT1-1CH1			A
			1.1.2		SLITTA A VITE GR.300- 0.700 MONTESIC TVP300 C0700			B
			1.1.3		CATENARIA PORTADAVI- 17 MAGLIE (Ø.782 M) 10x5 250010.075			C

Figure 45 Detail of the mechanical ledger

The first document analysed is the cyclogram. Despite the clear representations of the sequence of operations, information is rather scattered and stored in several files. After a preliminary analysis of all the related pieces of information, it was possible to link the components in the FMECA with the activities indicated in the cyclogram.

The complexity of the system under analysis now is much higher than in the previous case study. Due to the need of several modelling patterns, resembling the synchronization or parallel execution of the assembly activity, plus, the presence of multiple failure causes for each component and multiple components that may fail in each of the assembly activity, a rather high computation power is required (more details below).

Figure 46 represents the sequence of 56 activities that must be carried out by the station to complete the assembly process. This diagram constitutes the back bone of the Petri Net (system level elements) and will be further characterized by the details extracted from the FMECA and the cyclogram documents.

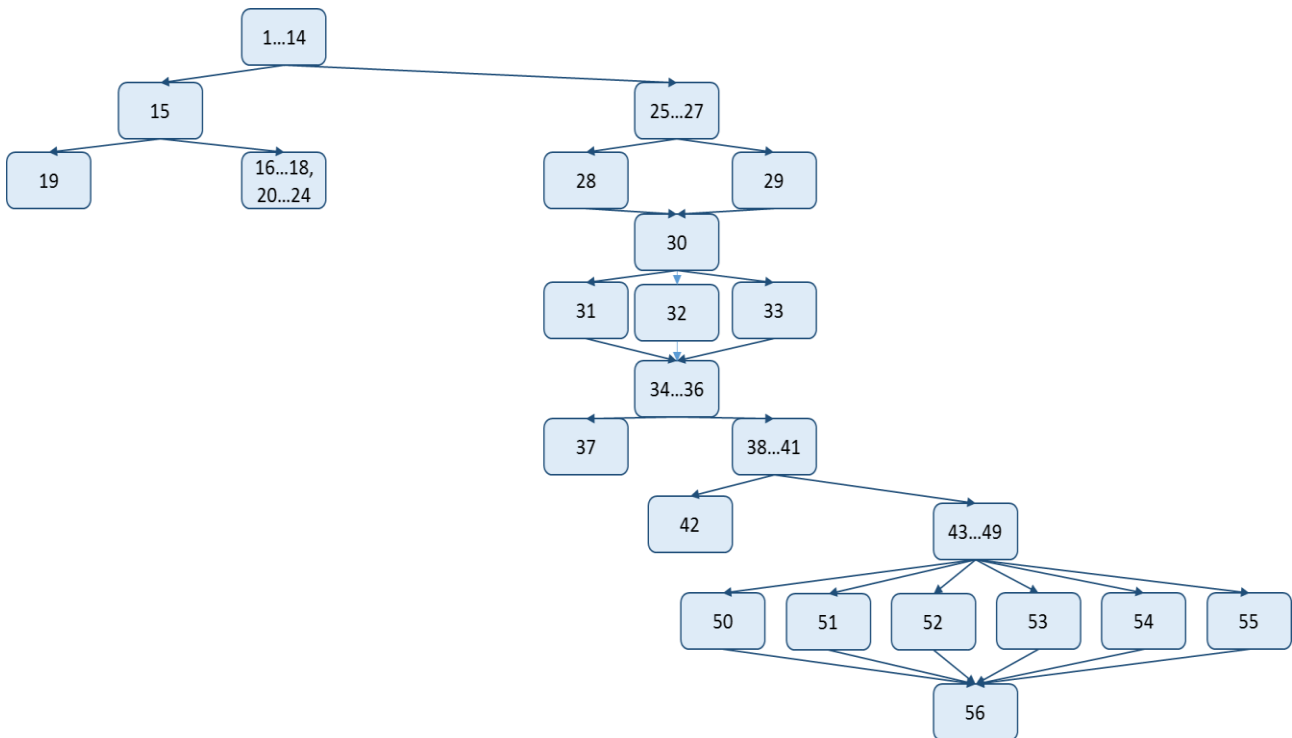


Figure 46 Activity-tree representing the logical sequence of the activities

Like in the previous case study, the behaviour of the systems can be described in terms of system states and their changes. Here we have 57 different system states that just resemble the sequence of operations. However, taking into account the 25 components, you may already have a rough estimation of the size of the state space.

In order to simulate the dynamic behaviour of a system, a state or marking in a Petri net is changed according to the transition (firing) rule described in the previous sections, which takes into consideration the execution time of the operations – reasonably – considered having a fixed and deterministic durations.

Regarding the modelling patterns, here we had to adopt 5 different patterns, namely:

- Sequential (Figure 65 in Annex B1)
- Parallel (Figure 66 in Annex B1)
- Branch-Ultimate (Figure 67 in Annex B1)
- Synchronization (Figure 68 in Annex B1)
- Repetitive (Figure 69 in Annex B1)

Those patterns together with the elements presented in the first case study put the basis for modelling and analysis of the assembly station (more details in the Annex).

Creation of the OWL individuals: from RAASO to PN4RO elements

Using the FMECA report it is possible to extract the detailed information required to instantiate the element of the RAAS OWL Ontology (Table 20). The table below shows the main elements from which the instances of the PN4RO will be subsequently created. More details regarding the latter have been already presented in the previous section, this is why here we limit the explanations up to the cardinality of each of the classes and instances of the ontological model.

Table 20 Instances of the AASO applied to the use case

Class	Number of instances and examples
Activity	56 (E.g. Pallet In Prestop, Stop Pallet In Stazione, Traslazione Pallet, Lettura Tag Magnetico, Sollevamento Pallet)
SystemState	57 (E.g. Before Pallet In Prestop, After Pallet In Prestop Before Stop Pallet In Stazione, After Stop Pallet In Stazione Before Traslazione Pallet)
CausesOfFailure Component	11 (E.g. Degrado caratteristiche meccaniche di molla, Deperimento lubrificante interno) 25 Electrical (E.g. Servomotore sincrono Siemens) 4 Mechanical (E.g. Pressa elettromeccanica Promess 10KN c.300) 8 Pneumatic (E.g. Cilindro pneumatico SMC c95sdb32-160)

Based on the procedure described in the first use case, for each instance already present in the AASO, new instances will be generated in the PN4RO. Table 21 shows the number of the individuals for each PN4RO OWL class providing an insight into the complexity of the net.

Table 21 Instances of the PNO applied to the use case

Class	Number of instances and examples
Place	71 SystemPlace (E.g. Pallet In Prestop, Stop Pallet In Stazione), 290 ComponentPlace (E.g. Pallet In Prestop CAC01G1, Pallet In Prestop CNAC01G1)
Transition	59 SystemTransition (E.g. T12, T23), 290 ComponentTransition (E.g. TFCAC01G1, TRCNAC01G1)
Arc	142 SystemArc (E.g. PTA1, TPA1), 870 ComponentArc (E.g. Pallet In Prestop PTFC01G1, Pallet In Prestop PTRC01G1, Pallet In Prestop TPFC01G1, Pallet In Prestop TPRC01G1)

In order to detect the failures possibilities related to each component and of each failure typology, the component module was replied, when necessary, to take into consideration for each phase the multiplicity of the component itself (i.e. if more than one component of the same typology intervenes in the phase, it is replicated as many times as the number of components) and the multiplicity the failure typology. The FMECA technique is firstly used by the company to collect data on system failures mode and on failure criticality; from a system availability point of view, a failure is considered critical if it causes a system breakdown. It must be considered that our analysis focused on the failure events that involve an obliged stop of the system in case of occurrence. A similar starting approach was used also by [123] to set the initial values of the model proposed by the authors.

The values of the transitions are computed in the following way. The FMECA report given by the company provided the values of MTBF in years, so it has been necessary to translate them in hours, multiplying such values for 6000 hours, which is the yearly amount of hours that the station works.

$$MTBF (years) * 6000 hours = MTBF (hours)$$

Equation 12 Mean Time Between Failures

Then it was necessary to compute the corresponding rate (indicated with λ):

$$\lambda = \frac{1}{MTBF (hours)}$$

Equation 13 Failure Rate

The same process was followed for computing the repairing/replacing rate (indicated with μ), with the only difference that the value reported inside the FMECA was already given in hours:

$$\mu = \frac{1}{MTTR (hours)}$$

Equation 14 Replacing Rate

The simulation has been performed in accordance with the following simulation parameters:

- Confidence Interval: 95%
- Approximation: 20%
- Set batch: duration
- Set batch length to min max: 175200

However, due to the difference between the MTTF and the duration time of the assembly operations, the simulation engine used to analyse the dynamic behaviour of the system could not converge to a reliable solution, which in the “worst case” indicated an overall system reliability of about 99.98% that, on the one end, describes a scenario plausible for the company. On the other hand, instead, we certainly attribute such a positive evaluation to the need of a Rare Event Simulation engine, which is missing in the current version of the simulation tool.

6.2.3 Discussions around the application case

The company providing the application case presented in the previous sections could certainly take advantage of the proposed solution as the results achieved are enriched, raise awareness and clearness compared to the conventional performance analysis carried out on their assembly stations. Following are a comprehensive list of all the attainable results. No further details are hereafter provided as they can be found in Sections 6.2.1 and 6.2.2.

Regarding the dependability of the overall system and its components, through the proposed solution it is possible to provide the following analysis:

- The probability that a failure happens in a specific unit
- The probability that a component of a specific type fails
- The probability that a failure happens for a specific cause
- The overall probability of failure of the station (or its availability)
- The number of failures of each component in a specific time frame
- The total repairing hours for a station in a specific time frame
- The total repairing hours per station divided by the category of component
- The number of items to keep in stock for each component

Increasing awareness regarding the performance parameters above is vital for a company that produces automatic assembly lines as it also increases proactive maintenance actions in case of which the sporadic – even unlikely – occurrence of failures. All this results in a more reliable manufacturing system, which ultimately implies a reduction of revenue losses.

6.3 Application Case 3: Automated HAZOP Analysis of a Chemical Plant

Safety in industrial environments is important as it safeguards human life, especially in high-risk areas such as chemical and oil and gases industries, where a fatal mistake or unpredicted event could be catastrophic. Industrial safety, therefore, aims at reducing risks to industrial processes, human health, and last but not least environment.

Maintaining a safe and healthy working environment is not only an important human resources issue, but it is also the law. In this regard, the Seveso-III-Directive (full title: Directive 2012/18/EU of the European Parliament and of the Council of 4 July 2012 on the control of major-accident hazards involving dangerous substances, amending and subsequently repealing Council Directive 96/82/EC Text with EEA relevance) is a European Union directive aimed at the prevention of major accidents involving dangerous substances. However, as accidents may nevertheless occur, it also aims at limiting the impact and consequences of such accidents³⁷.

Reliability methodologies can be roughly divided into four groups: (i) *Guidelines* describe the process of reliability analysis, general steps, reports, actions (e.g. FMEA, Root Cause, and HAZOP), which use (ii) *System-level* models describe the reliability aspects of the product on a system level (e.g. RDB, FTA, and ETA) that use (iii) *Component-level* metrics define the reliability characteristics of a single component (e.g. MTTF, MTTR, and Failure rate) and (iv) *Mathematical models* are used for numerical (probabilistic) reliability analysis (e.g. DTMC, CTMC, and SPN). *Guidelines* can be considered as the most top level methodologies used for reliability assessment because they describe how the reliability analysis should be carried out. Identification and analysis of risks, which potentially impact on the reliability of a system, has been proved to be very useful for the risk-driven design phase of industrial plants [124], [125], however, this may require being continuously revalued. In this context, although being time-consuming and requiring significant human and economic resources, the HAZOP study is considered the most dependable methodology for reliability assessment and risk identification in chemical and petrochemical plant.

The Hazard and Operability study is a standard hazard analysis technique used in the preliminary safety assessment of the new system or modifications to existing ones. The HAZOP study is a detailed examination, by a group of specialist, of components within a system, to determine what would happen if that component was to operate outside its normal design mode or under abnormal conditions. Although it is time-consuming and requires significant human and economic resources, the HAZOP study is up to date considered the most dependable tool for reliability assessment in chemical plants.

Nowadays the new methodologies for reliability assessment are rather extensions of the old ones or combinations of some of them. Following this approach, we thoroughly present a tool combining the HAZOP methodology with Coloured Petri nets in a way to simulate the propagation of failures throughout an industrial plant. Therefore, as an attempt to respond the RQ1 and RQ3, the proposed solution extends the works [126], [127] by investigating more use cases and, above all, by employing Semantic Web Technologies, such as OWL Ontology, which embeds the system with context-awareness capabilities. As a result, the exploration of all the possible system failures is proven to be more effective and less costly.

6.3.1 The HAZOP methodology

The HAZOP technique was developed by ICI (Imperial Chemical Industries, a British chemical company) in the UK in the 1960s, however, even today it is widely adopted in diverse systems engineering applications for risk identification and reliability assessment.

A HAZOP study is a detailed process carried out by a dedicated team to identify risks and operability problems. HAZOP studies deal with the identification of potential deviations from the design intent, examination of their possible causes and assessments of their consequences. This is a structured and systematic examination of a planned or existing process or operation with the aim of identifying and evaluate problems that may represent risks to personnel or equipment, or prevent efficient operation, whereas:

- A *Hazard* is any existing or potential condition that can lead to injury, illness, or death to people; damage to or loss of a system, equipment, or property; or damage to the environment
- *Operability* problems are referred as to causes of operational disturbances and process deviation likely to lead to non-conforming products and/or violation of environmental, health or safety regulations.

³⁷ <http://ec.europa.eu/environment/seveso/legislation.htm>

HAZOP is a qualitative methodology which embraces deductive aspects (search for causes) and inductive aspects (consequences analysis) with the objective of identifying the initiating events of undesired accident sequences [128]. The stepwise procedure can be summarized as follows:

1. Decompose the system model into functionally independent process units (reaction unit, storage unit, pumping unit, etc.); for each process unit, identify the various operation modes (start-up, regime, shutdown, maintenance, etc.).
2. For each process unit and operation mode, identify the potential deviation from the nominal process behaviour. In order to do this, one should:
 - a. Specify all the unit incoming and outgoing fluxes (energy, mass, control signals, etc.) and the characteristic process variables (temperature, flow rate, pressure, concentration, etc.);
 - b. Write down the various functions that the unit is supposed to fulfil (heating, cooling, pumping, filtering, etc.);
 - c. Apply keywords (low, high, no, reverse, etc.) to the previously identified process variables and unit functions, so as to generate deviations from the nominal process regime.
3. For each process deviation (qualitatively) identify its possible causes and consequences. For the consequences, include effects also on other units: this allows HAZOP to account also for domino effects among different units.

At the outset of the study, a conceptual model of the system or operation should be built using all available, relevant material such as P&IDs, operating procedures, material data sheets and, if any, reports of earlier hazard studies. The method applies to processes for which design information is available. This commonly includes:

- Process flow diagrams,
- Piping and instrumentation diagrams (P&IDs),
- Layout diagrams,
- Material safety data sheets,
- Provisional operating instructions,
- Heat and material balances,
- Equipment data sheets start-up and emergency shut-down procedures.

The size of sections should be appropriate to the complexity of the system and the magnitude of the hazards it might pose. The HAZOP team then determines what are the possible significant *Deviations* from each intent, feasible *Causes* and likely *Consequences*. It can then be decided (at the HAZOP, or by subsequent analysis) whether existing, designed safeguards are sufficient, or whether additional actions are necessary to reduce risk to an acceptable level.

In order to identify deviations, the team applies a set of *guidewords* to each section of the process. To prompt discussion, or to ensure completeness, it may also be helpful to explicitly consider appropriate *parameters* which apply to the design intent. For example, Flow, Temperature, Pressure, Composition, Phase, Level, Relief, Instrumentation, Reaction, Maintenance, etc. The current standard (IEC 61882) notes that guidewords should be chosen which are appropriate to the study and neither too specific (limiting ideas and discussion) nor too general (allowing loss of focus). Once a meaningful deviation has been identified, the team responsible for the HAZOP study then seeks a cause. If there are likely to be several causes it is very helpful to have a short brainstorming session to identify as many causes as possible, remembering that causes may be related to human factors as well as to systems elements.

Although only realistic causes (and consequences) need to be discussed in detail, a judgment on this cannot be made without taking account of the nature and seriousness of the consequences. Acceptable risk involves a trade-off between frequency and severity so it is impractical to completely separate the discussion of cause and consequences in a HAZOP analysis. The term 'realistic' implies a consideration of the likely frequency of a cause. If only minor consequences arise, then low frequency causes may be ignored. In fact, a risk assessment is made based on a combination of the frequency of the event and the seriousness of the consequences. The consequences of each cause must be carefully analysed to see whether they take the system outside the intended range of operation. It is essential to fully identify all of the consequences, both immediate and delayed, and both inside and outside the section under analysis. It often helps to analyse how the consequences develop over a period of time, noting when alarms and trips operate and when and how the operators are alerted. This allows a realistic judgment on the likelihood and influence of operator intervention.

Where an effect occurs outside the section or stage being analysed, the HAZOP team leader must decide whether to include the consequences in the immediate analysis or to note the potential problem and defer the analysis to a later, more suitable point in the overall HAZOP study.

Figure 47 summarizes the stepwise HAZOP study, however, Risk ranking, Agreement on actions to be taken and Monitoring are out of the scope of this work. Finally, a list of potential advantages and disadvantages is hereafter presented [129]:

Advantages

- Systematic and comprehensive technique. A detailed plan for performing the technique is available which systematically applies guidewords and parameters to all the pipes and vessels in the process.
- Examines the consequences of failure. Thought should be given by the assessment team to the consequences of the deviations identified. This aids in the production of recommendations for methods to minimise or mitigate the hazard.

Disadvantages

- Time consuming and expensive. Most plants contain a large number of pipes and vessels each of which needs to be examined by the application of the various guidewords and parameters.
- Requires detailed design drawing to perform the full study. To fully perform the study, the process has to be designed to such a level that all the pipes and vessels are detailed with their operating conditions and control instrumentation.
- Additional guide words are required for unusual hazards. For specific dangers that will not be covered by the general guide words, further words (such as radiation for the nuclear industry) will need to be applied.
- Requires experienced practitioners. Experienced team members are required to identify all possible causes and consequences of the deviations, as well as producing realistic recommendations.
- Focuses on one-event causes of deviation only. Only the hazards associated with single deviations can be studied. Hazards that are caused by two or more separate deviations cannot be identified by the technique.

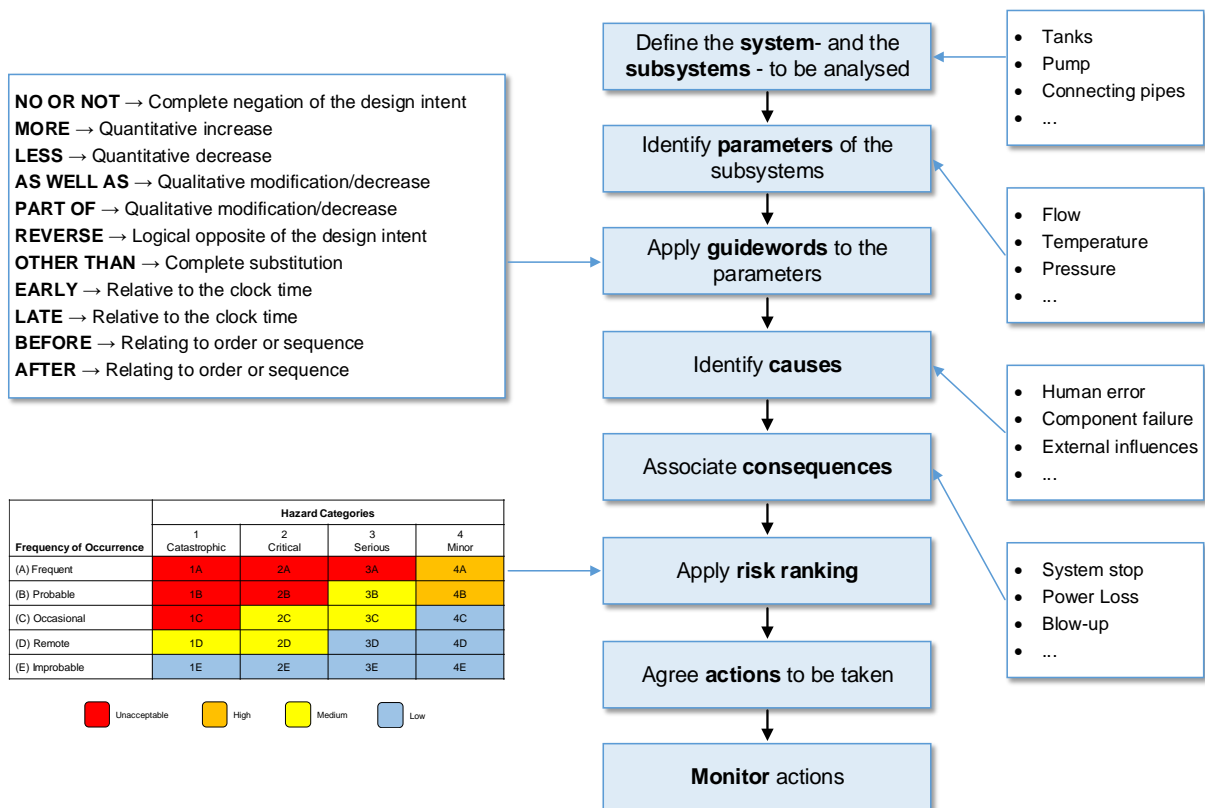


Figure 47 HAZOP study in a nutshell

6.3.2 Coloured Petri Nets

Formal definition

Non-hierarchical Coloured Petri nets (CPNs) belong to the area of discrete event system methodology [130]. A CPN is well known for its capability in modelling discrete event systems.

The structure of a Petri net is a bipartite directed graph describing the structure of a discrete event system, while the dynamics of the system is described by the execution of the Petri net. A Petri net is coloured if the tokens are distinguishable.

According to CPNs [131], a Coloured Petri Net model is a nine-tuple

$$CPN = (\Sigma, P, T, A, N, C, G, E, IN)$$

satisfying the following requirements:

1. Σ is a finite set of non-empty types, called colour sets
2. P is a finite set of places
3. T is a finite set of transitions
4. A is a finite set of arcs such that $P \cap T = P \cap A = T \cap A = \emptyset$
5. $N: A \rightarrow P \times T \cup T \times P$ is a node function
6. $C: P \rightarrow \Sigma$ is a colour function
7. G is a guard function. It is defined from T into expressions such that

$$\forall t \in T: [Type(G(t)) = Bool \wedge Type(Var(G(t))) \subseteq \Sigma]$$

8. E is an arc function. It is defined from A into expressions such that

$$\forall a \in A: [Type(E(a)) = C(p(s))_{MS} \wedge Type(Var(E(a))) \subseteq \Sigma]$$

where $p(a)$ is the place of $N(a)$ and C_{MS} denotes the set of all multi-sets over C

9. IN is an initialization function. It is defined from P into expressions such that

$$\forall p \in P: [Type(IN(p)) = C(p(s))_{MS} \wedge Type(Var(IN(p))) \subseteq \Sigma]$$

where:

$Type(expr)$ denotes the type of an expression,

$Var(expr)$ denotes the set of variables in an expression,

$C(p)_{MS}$ denotes a multi-set over $C(p)$.

A binding of a transition t is a function b defined on $Var(t)$, such that:

- (i) $\forall v \in Var(t) : b(v) \in Type(v)$,
- (ii) $G(t)(b)$ denotes the evaluation of the guard expression $G(t)$ in the binding b .

For each CPN-based model we, therefore, define the following elements:

- I. A **marking** is a function M that maps each place $p \in P$ into a multiset of tokens $M(p) \in \mathcal{C}(p)_{MS}$.
- II. The **initial marking** M_0 is defined by $M_0(p) = I(p)(\cdot)$ for all $p \in P$.
- III. The **variables of a transition** t are denoted $Var(t) \subseteq V$ and consist of the free variables appearing in the guard of t and in the arc expressions of arcs connected to t .
- IV. A **binding** of a transition t is a function b that maps each variable $v \in Var(t)$ into a value $b(v) \in Type[v]$. The set of all bindings for a transition t is denoted $B(t)$.
- V. A **binding element** is a pair (t, b) such that $t \in T$ and $b \in B(t)$. The set of all binding elements $BE(t)$ for a transition t is defined by $BE(t) = \{(t, b) \mid b \in B(t)\}$. The set of all binding elements in a CPN model is denoted BE .
- VI. A **step** $Y \in BE_{MS}$ is a non-empty, finite multiset of binding elements.

CPN Tools

CPN Tools is a tool for the editing, simulation, state space analysis, and performance analysis of CPN models. CPN Tools supports untimed and timed hierarchical CPN models. A licence for CPN Tools can be obtained free of charge via the CPN Tools Web pages. In this paragraph, we provide a very brief introduction to CPN Tools. The CPN Tools Web pages contain an elaborate set of manuals on how to use the tool.

The user of CPN Tools works directly with the graphical representation of the CPN model. The graphical user interface (GUI) of CPN Tools has no conventional menu bars and pull-down menus, but is based on interaction techniques, such as tool palettes and marking menus. Figure 48 provides a screenshot of CPN Tools. The rectangular area to the left is an index. It includes the Tool box, which is available for the user to manipulate the declarations and modules that constitute the CPN model.

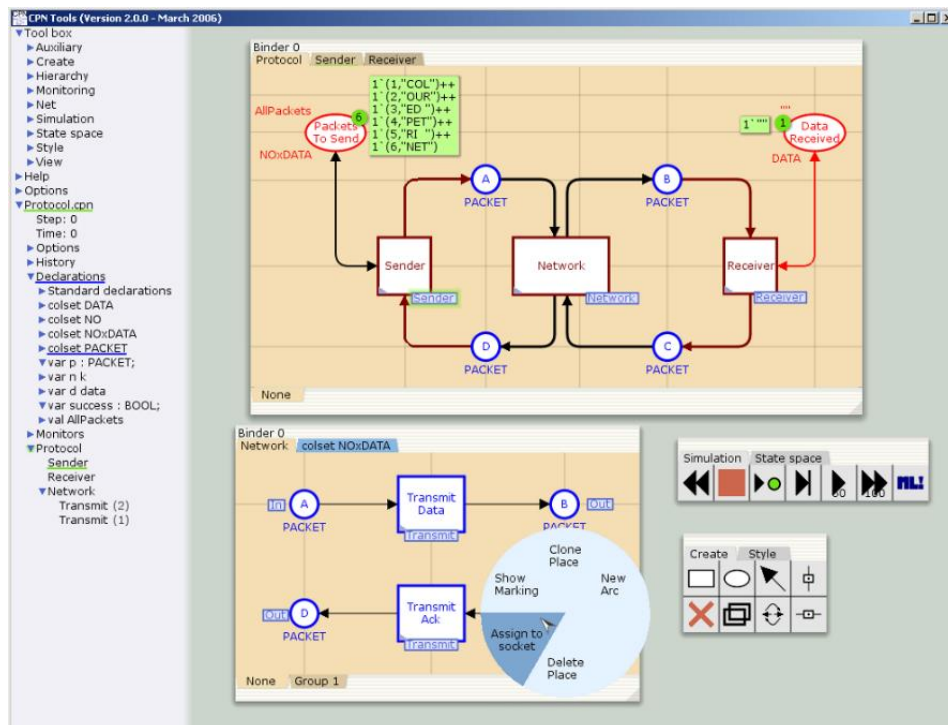


Figure 48 CPN Tools screen shot (www.cs.au.dk/CPNTools)

The Tool box includes tools for creating, copying, and cloning the basic elements of CP-nets. It also contains a wide selection of tools to manipulate the graphical layout and the appearance of the objects in the CPN model. The latter set of tools is very important in order to be able to create readable and graphically appealing CPN models. The remaining part of the screen is the workspace, which in this case contains four binders (the rectangular windows) and a circular pop-up menu.

There are two kinds of binders. One kind contains the elements of the CPN model, i.e., the modules and declarations. The other kind contains the tools which the user applies to construct and manipulate CPN models. Items can be dragged from the index to the binders, and from one binder to another binder of the same kind. It is possible to position the same item in two different binders, for example, to view a module using two different zoom factors.

CPN Tools performs syntax and type checking, and error messages are provided to the user in a contextual manner next to the object causing the error. The syntax check and code generation are incremental and are performed in parallel with editing.

CPN Tools supports two types of simulation: interactive and automatic. In an interactive simulation, the user is in complete control and determines the individual steps in the simulation, by selecting between the enabled events in the current state. CPN Tools shows the effect of executing a selected step in the graphical representation of the CPN model. In an automatic simulation, the user specifies the number of steps that are to be executed and/or sets a number of stop criteria and breakpoints. The simulator then automatically executes the model without user interaction by making random choices between the enabled events in the states encountered. Only the resulting state is shown in the GUI. A simulation report can be saved, containing a specification of the steps that occurred during an automatic simulation.

Context-adaptive Petri Nets

Context-adaptive Petri nets (CAPNs) enhance Coloured Petri nets with ontology-based structures that aim at including context information within the CPN-based model. Although not widely used or accepted, a few approaches dealing with context-awareness and PNs have been presented in the last few years. Here we adopt the approach presented by Serral and colleagues in 2015, which allow seamless integration of context data with CPNs, hence, using an ontology to model such context data [132].

The following context hence references can be introduced in CAPN:

- The data property value of a specific individual.
- The instance(s) related through an object property of a specific individual.
- List of individuals of a certain class.

These context references can be used in various constructs within a CPN, which is referred to as “context dependent variation points”, e.g. Transition guards, Transition timing, Transition functions, Arc expressions, Initial marking.

In order to allow the modelling of context-adaptive behaviour, CAPN enables context references in the expressions (*expr*, see CPN Formal Definition) used in the identified context dependent variation points (more details in the next Section). And in so doing, CAPN deals with context data and behaviour as separate concerns: the context is properly defined in an ontology – context – model (more details about the ontology in the last section), while the CPNs can refer to the context represented within the latter and deal with the (modelled system’s) behaviour in a context-adaptive way.

6.3.3 Context-adaptive Coloured Petri Nets for HAZOP assessment

Propagation of process deviations

The identification of the initiating events of undesired events (such as component failures) is crucial in the HAZOP study and is carried out by a person in a team of experts. The mental (stepwise) process according to which the expert seeks for a process deviation - either in a section (or node) of the plant or a single component - is exemplified in Figure 49.

The qualitative information (e.g. No, Low, High) related to a process variable (e.g. Temperature, Pressure, Level) is defined to resemble the discrete quantitative levels that can be possibly reached the latter. For example, high temperature could represent a temperature value 20% higher than the expected one; likewise, low temperature might represent a temperature value 20% less than the expected one).

As a deviation is selected for further investigation, the next question is: “What are the possible causes and consequences having internal or external effects?”. With this regard, we distinguish two different types of causes and consequences:

- Internal Cause - Any fault of a component resulting in a deviation on one of its process parameters.
- External Cause - The propagation of a process deviation from an upstream component or an unexpected external event affecting the component’s process parameters.
- Internal Consequence – As a result of an internal or external consequence, the component behaviour can deviate from the expected one or even fail.
- External Consequence – The component behaviour’s deviation can propagate through the downstream components.

It might be already clear how time consuming can this activity be. This is why, there is a need for reducing this time by automating the mental process followed to detect the fault paths, starting from the selection of the component, exploration of all the possible deviations, and identification of the related causes and consequences.

The propagation of a fault through a plant can be split into three elements: the initiation of a fault in a unit, which is unhealthy/faulty, the passage of the fault through units, which are otherwise healthy, and the termination of the fault in a unit, which is thereby rendered unhealthy [133]. In the same way, a failure event (Internal or External Cause) resulting in a process deviation within a component (Internal Consequence) affects the component per se and it can also become an External Consequence for a downstream component until such a failure propagation reaches the end of the system and cannot affect the direct next component anymore.

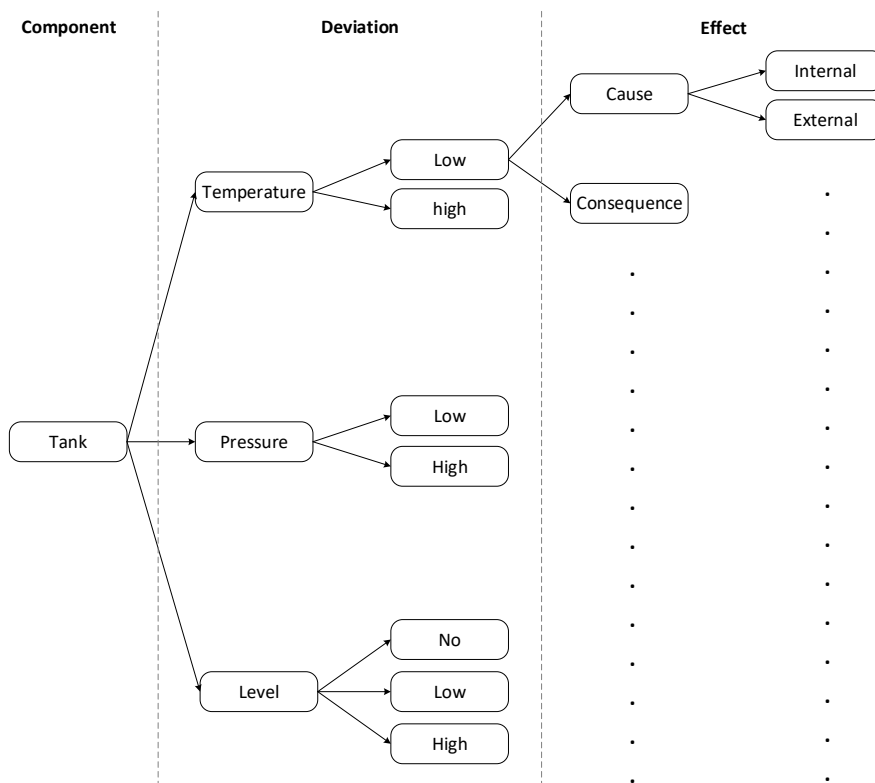


Figure 49 From Component to Effect through Deviations

System behaviour modelling with CPN

The CPN model is built starting from the conceptual design presented above. Figure 50 shows an excerpt of the necessary declarations for the CPN model of the plant and designed to resemble the system’s behaviour. It is worth noting that *colour sets* such as *PARAMETER*, *GUIDEWORD*, *DEVIATION* together with the function called *DevFun*, make possible the generation of all the possible deviations deriving from the sound combination of HAZOP parameters and guidewords (a complete list of deviation is presented in Annex).


```

▼ colset PARAMETER = with MORE|LESS|NO|REVERSE|ASWELLAS|PARTOF|OTHERTHAN;
▼ var p : PARAMETER;
▼ colset GUIDEWORDED = with FLOW|PRESSURE|TEMPERATURE|LEVEL|REACTION; (**there're more**)
▼ var g : GUIDEWORDED;
▼ colset DEVIATION = string;
▼ var d : DEVIATION;
▼ fun DevFun (parameter, guideword) =
  case (parameter, guideword) of
    (**meaningful deviations **)
    (MORE, FLOW) => "High Flow" | (MORE, PRESSURE) => "High Pressure" |
    (MORE, TEMPERATURE) => "High Temperature" | (MORE, LEVEL) => "High Level" |
    (MORE, REACTION) => "Fast Reaction" |
    (LESS, FLOW) => "Low Flow" | (LESS, PRESSURE) => "Low Pressure" |
    (LESS, TEMPERATURE) => "Low Temperature" | (LESS, LEVEL) => "Low Level" |
    (LESS, REACTION) => "Slow Reaction" |
    (NO, FLOW) => "No Flow" | (NO, PRESSURE) => "Vacuum" |
    (NO, LEVEL) => "No Level" | (NO, REACTION) => "No Reaction" |
    (REVERSE, FLOW) => "Reverse Flow" |
    (ASWELLAS, FLOW) => "Deviating Concentration" | (ASWELLAS, PRESSURE) => "Delta-p" |
    (ASWELLAS, LEVEL) => "Different Level" |
    (PARTOF, FLOW) => "Contamination" |
    (OTHERTHAN, FLOW) => "Deviating Material" | (OTHERTHAN, REACTION) => "Unwanted Reaction";
    
```

Figure 50 CPN model declarations

The reaction section of the plant comprises six (6) components, regarding to which one or more process variable play a role for the study of the system behaviour (Table 22).

Table 22 Components and Process Variables

Components	Process Variables					
	Flow	Pressure	Temperature	Level	Composition	Reaction
Reactor - Column	X	X	X	X		X
Heat Exchanger	X		X		X	
Valve	X					
Tank	X	X	X	X		
Pump	X	X				
Filter	X		X		X	

Therefore, introducing some correspondences with the basic modelling elements of CPN, i.e. places and transitions, we can define these as follows:

- **Place** (circle) – it can be used for modelling the process variables related to the component, or just including in the model additional information *post* or *pre* conditions for the transitions.
- **Transition** (square) - it can be used for modelling the dynamic aspects (for example, process parameter or component status changes) of each component or the elementary units, which made up the component.

Now the question is: "How do the propagations can be represented in CPN?". At this stage, it is necessary to define how a process variable deviation might influence another one. Seven (7) scenarios have been hence defined, which eventually characterize six (6) different CPN modelling structures (Figure 51):

1. (more X → more Y) An increase in X results in an increase in Y
2. (less X → less Y) A decrease in X results in a decrease in Y
3. (more X → less Y) An increase in X results in a decrease in Y
4. (less X → more Y) A decrease in X results in an increase in Y
5. (X <> N → more Y) Any deviation in X results in an increase in Y
6. (X <> N → less Y) Any deviation in X results in a decrease in Y
7. (X=0 → Y=0) An absence of X lead to an absence of Y

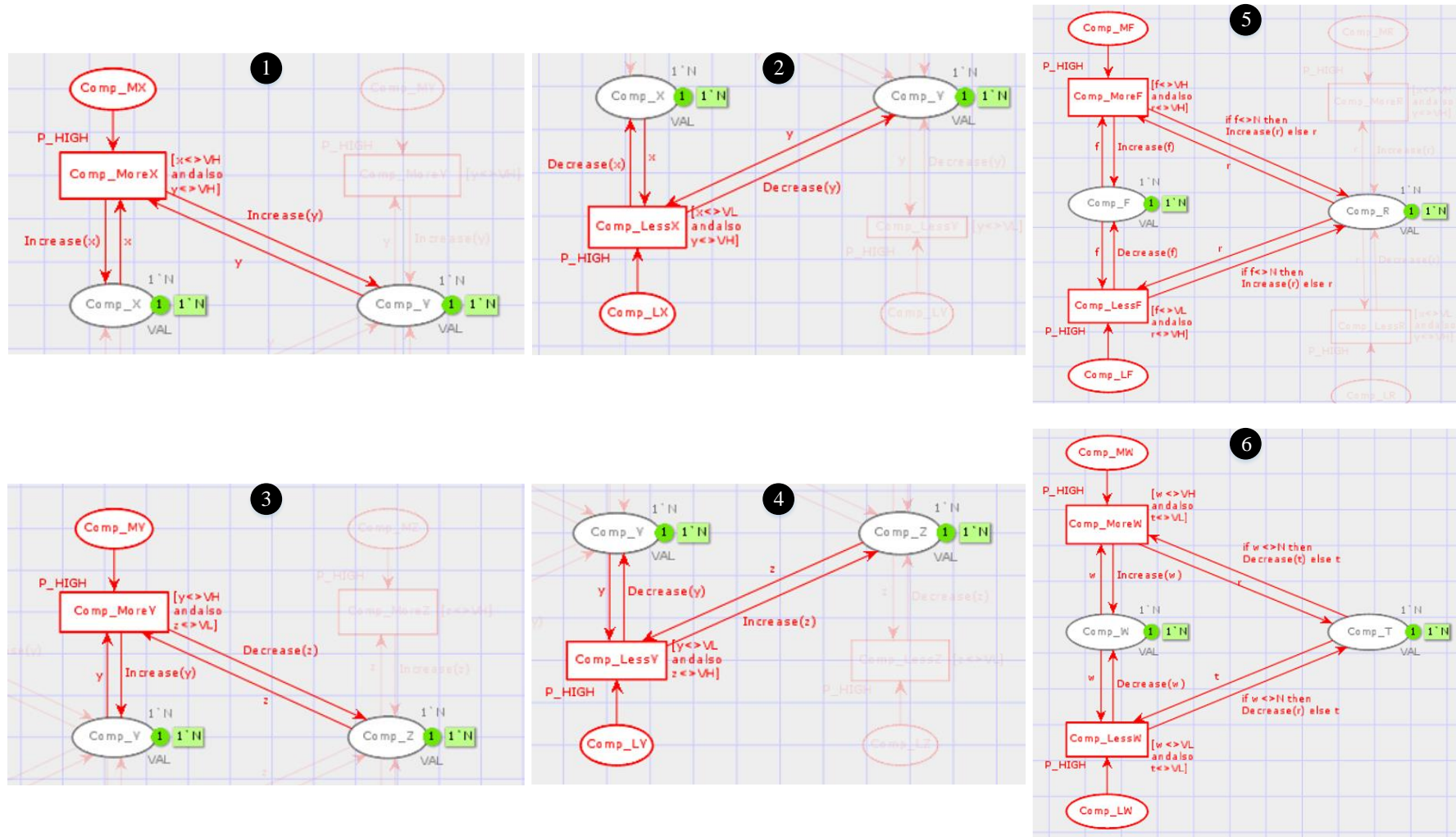


Figure 51 CPN models for the six deviation scenarios

The last scenario is usually modelled by a dedicated transition that set as *Z* the *VAL* token within a place.

Each process variable can therefore deviate according to the related discrete levels. Five qualitative levels have been defined: *Very Low, Low, No, High, Very High*. The following colour set represents those values:

$$\text{colset VAL} = \text{with Z | VL | L | N | H | VH};$$

When an external or internal cause occurs, a token is moved within places such as *Comp_MX* or *Comp_LX*. This is an enabling condition for the transition linked to this place through a *PT Arc* (Place to Transition Arc), which will eventually fire with the highest priority, due to the *P_HIGH* priority level assigned to it. The priority levels have been defined in order to strictly rule the firing sequence. For example, no more causes are evaluated until the propagation is not finished (*termination of the fault in a unit*), which is in line the HAZOP principles of single cause investigation.

The guard (e.g. $[x <> VH \text{ and also } y <> VH]$) is necessary for bounding transitions, meaning that such event related to the transition can happen till a certain value reachable by the token exchanged between the transition and the linked place.

The initial marking of process variable places has always an initial marking (*1 N*) as each component is assumed to be in a *Normal State* at the beginning.

When the transition *Comp_MoreX* fires the variable *x* is increased. Similarly, when the transition *Comp_LessX* fires the variable *x* is decreased.

To define the state of some components, such as valves and pumps, other colorsets were created:

- *colset PSTATE= with W|F; (*Working, Failed*)*
- *colset VSTATE= with C|FO|FC; (*Controlled, FailOpen, FailClose*)*
- *colset RSTATE= with S|NS. (*Product Standard, Not Standard*)*

Places and the related transitions have the same colour. This is in order to make the interpretation of the net easier. A detailed list of different *colors* used in this model along with a short description of their relation to the HAZOP study (in brackets) is following provided:

- **Green Places** (Failure Causes), support a STRING colorset and contain internal and external causes using a textual token, which can start a parameter deviation.
- **Red Places** (Failure Consequences), support a UNIT colorset, so the value of the token can be set only at “1” and they are initialized empty.
- **Grey Places** (Process Parameter Deviations), contain a VAL token, set at “Normal” level (colset VAL = with Z|VL|L|N|H|VH).
- **Purple Places** (Failure Consequences), support a STRING colorset and used to express the consequences as a string message to include in the HAZOP assessment report.
- **Gold Places** (Failure Consequences), initialized empty, allow the passage of a UNIT token with No-Flow information.

The transitions firing priorities play a crucial role while simulating the net. Those parameters, in fact, affect the behaviour of the components, hence, the sequence of steps that substantially characterize the propagation of the faults within the system model. This model provides for the use of four priority levels, defined by a label and a numeric value: the smallest the value the higher the priority.

- **Normal priority** ($P_NORMAL = 400$), used with green transitions to fire all the transitions containing internal and external causes. In the start condition, these causes have all the same probability of occurrence;
- **High priority** ($P_HIGH = 100$), associated to red transitions, it is the highest level of priority and it is the first firing after a start event so that the component state or the process variable level can change immediately; It is also associated with gold transitions, modelled to simulate the *No Flow* conditions in every component;

- **Medium priority** ($P_{MED} = 200$), used with purple transitions, so it is associated with the firing of the consequence transition;
- **Reset priority** ($300 < P_{RESET} < 399$), used with grey transitions, it allows the net reset when some specific conditions, or stop conditions, are reached. Unlike in [127], in this work, several *Reset priority levels* defined between 300 and 399, have been defined to cover the components spectrum and resemble the normal substance flow.

When the net is in the start condition, it can evolve by firing a random green transition; then a red one will be enabled and all the red transitions bounded downstream will fire.

At the end of the red transitions flow, there will be a consequent text message from a purple place. If the net reaches a stop condition, for example, one of the variables reaches VH or Z, the grey transitions will be enabled and the net will be reset to the start condition (or initial net marking).

Unlike the study presented in [127] and [126], the industrial plant under analysis in this work (Figure 52) consists of six different types of component: Valve (Figure 53), Reactor (Figure 54), Filter (Figure 55), Pump (Figure 56), Vessel (Figure 57), and Heat Exchanger (Figure 58).

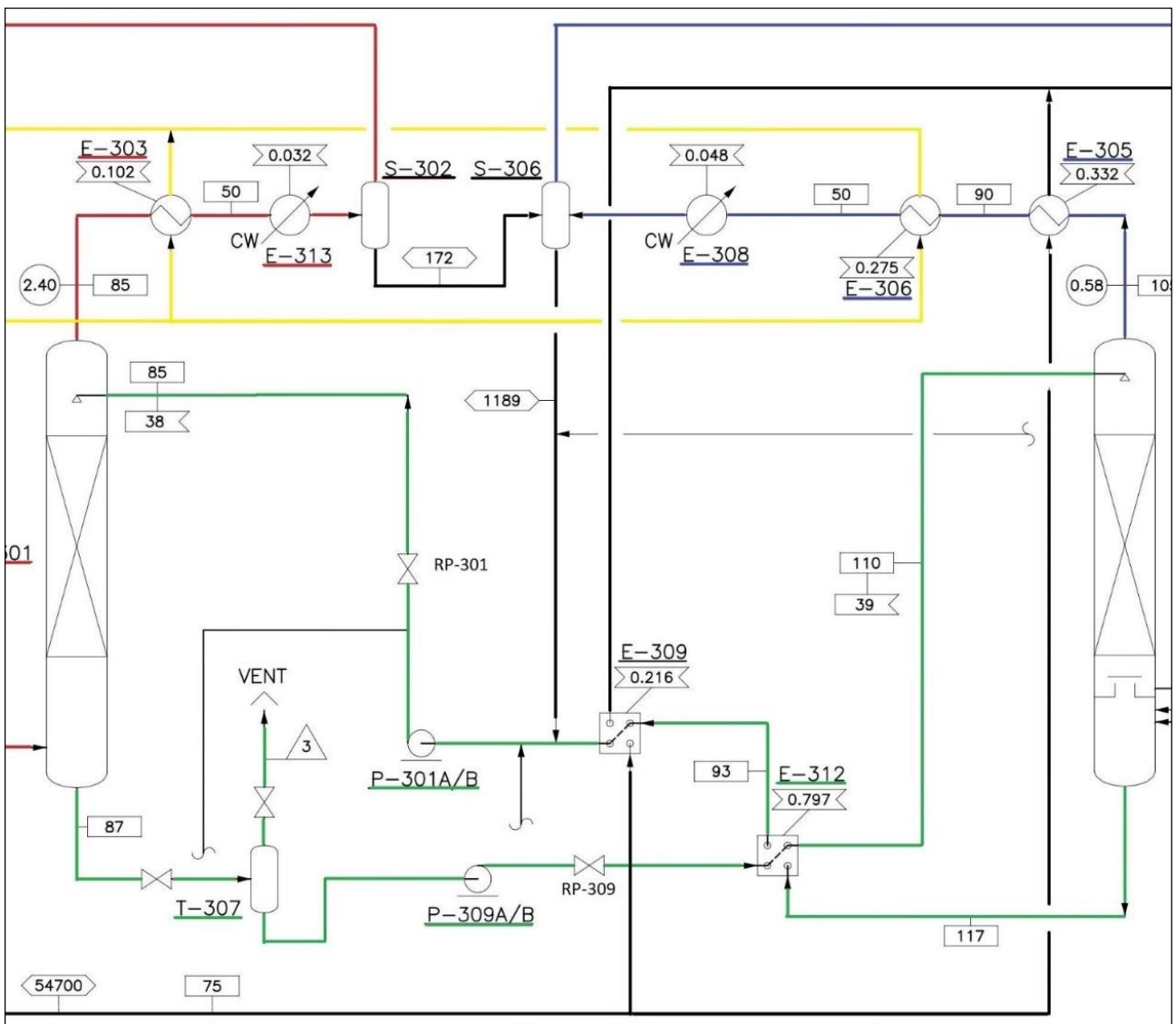


Figure 52 Extract of nodes partition of the plants

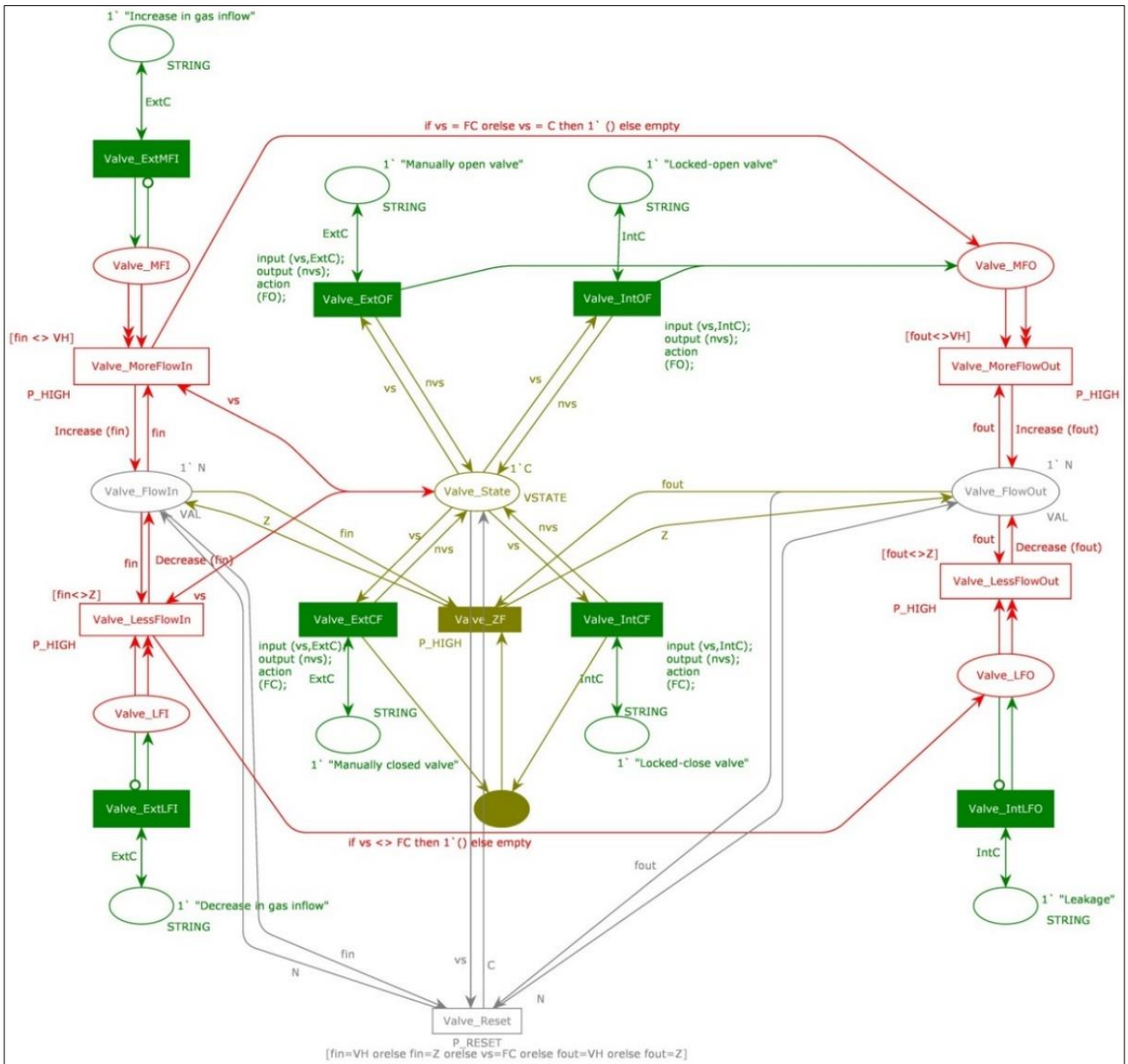


Figure 53 CPN Model of the Valve

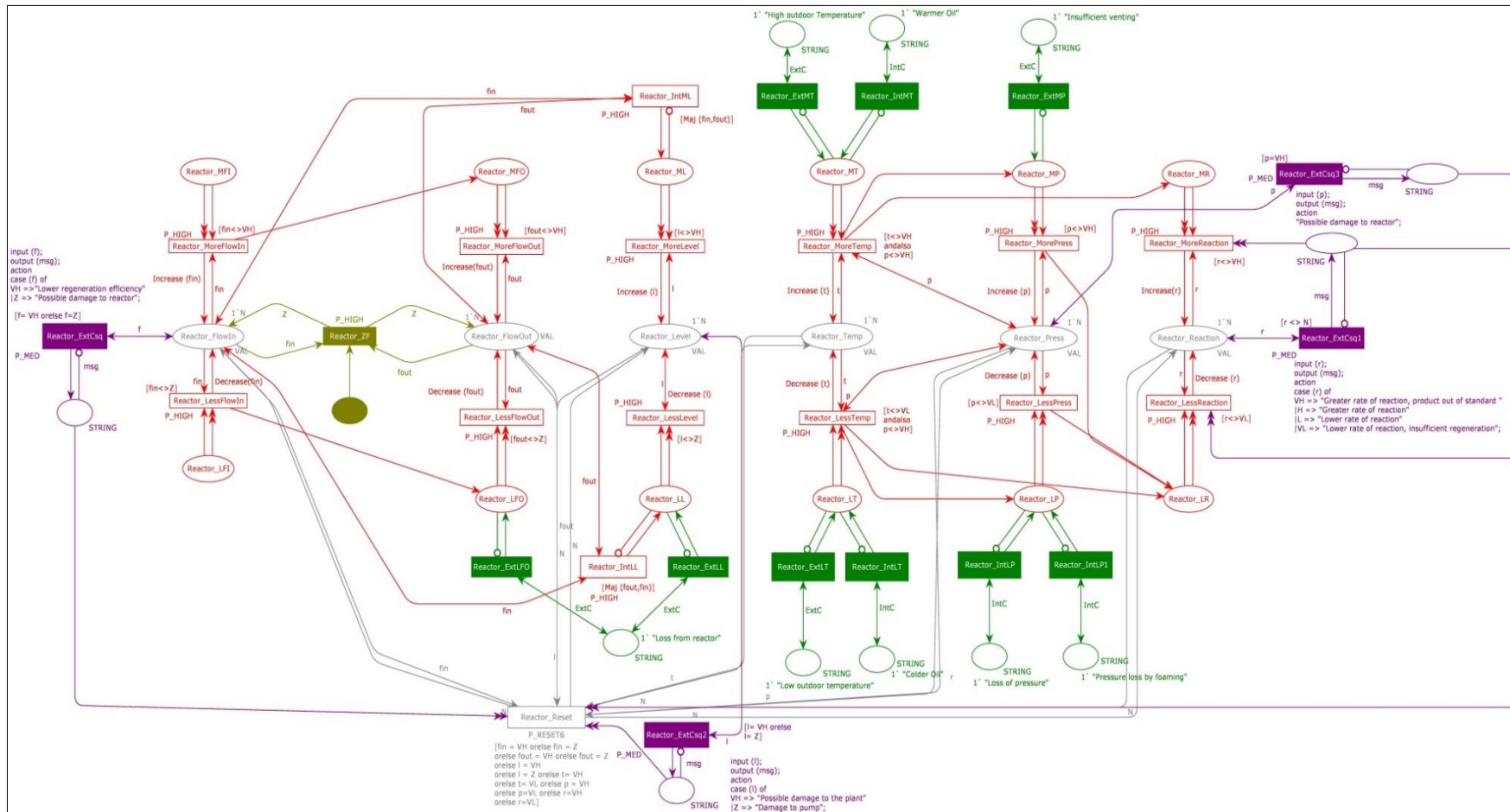


Figure 54 CPN Model of the Reactor

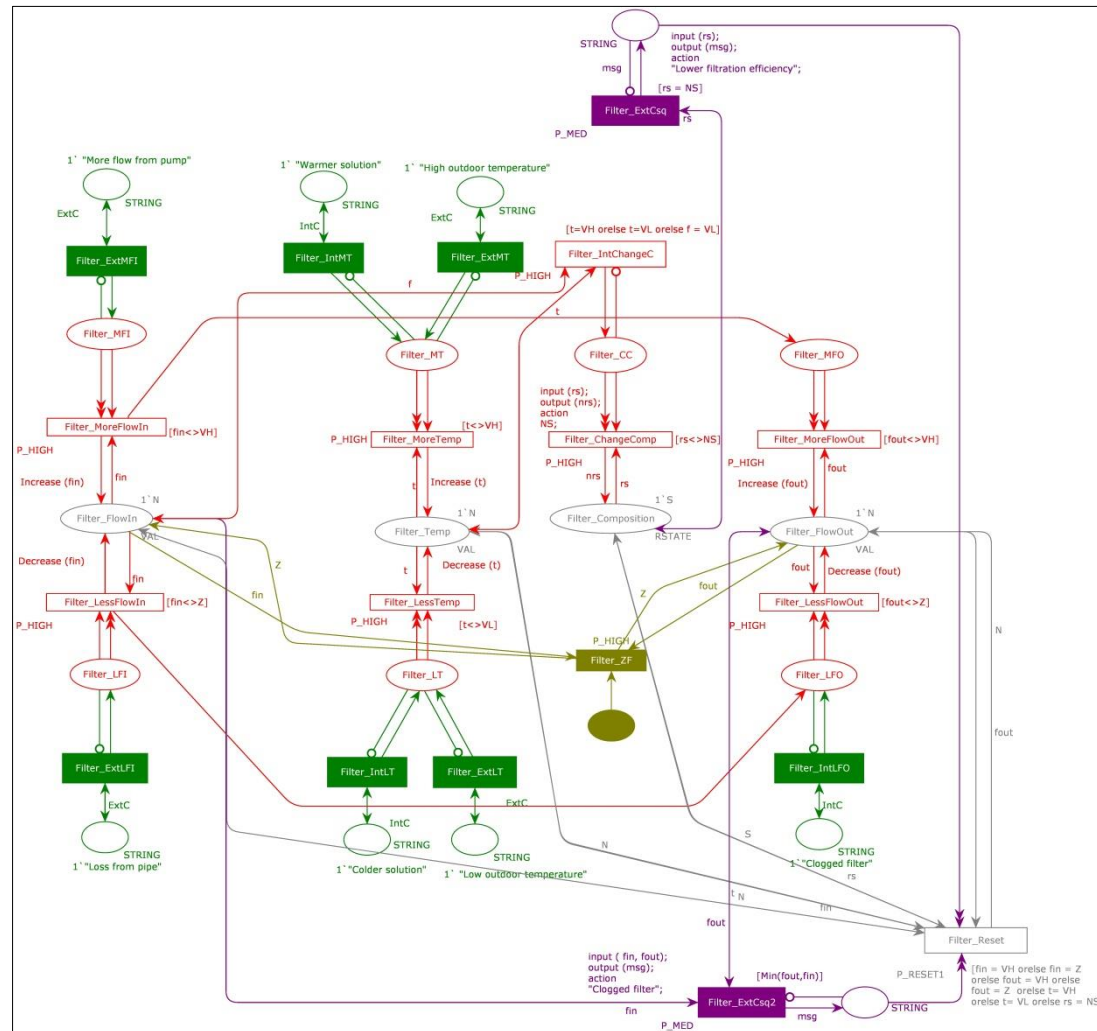


Figure 55 CPN model of the Filter

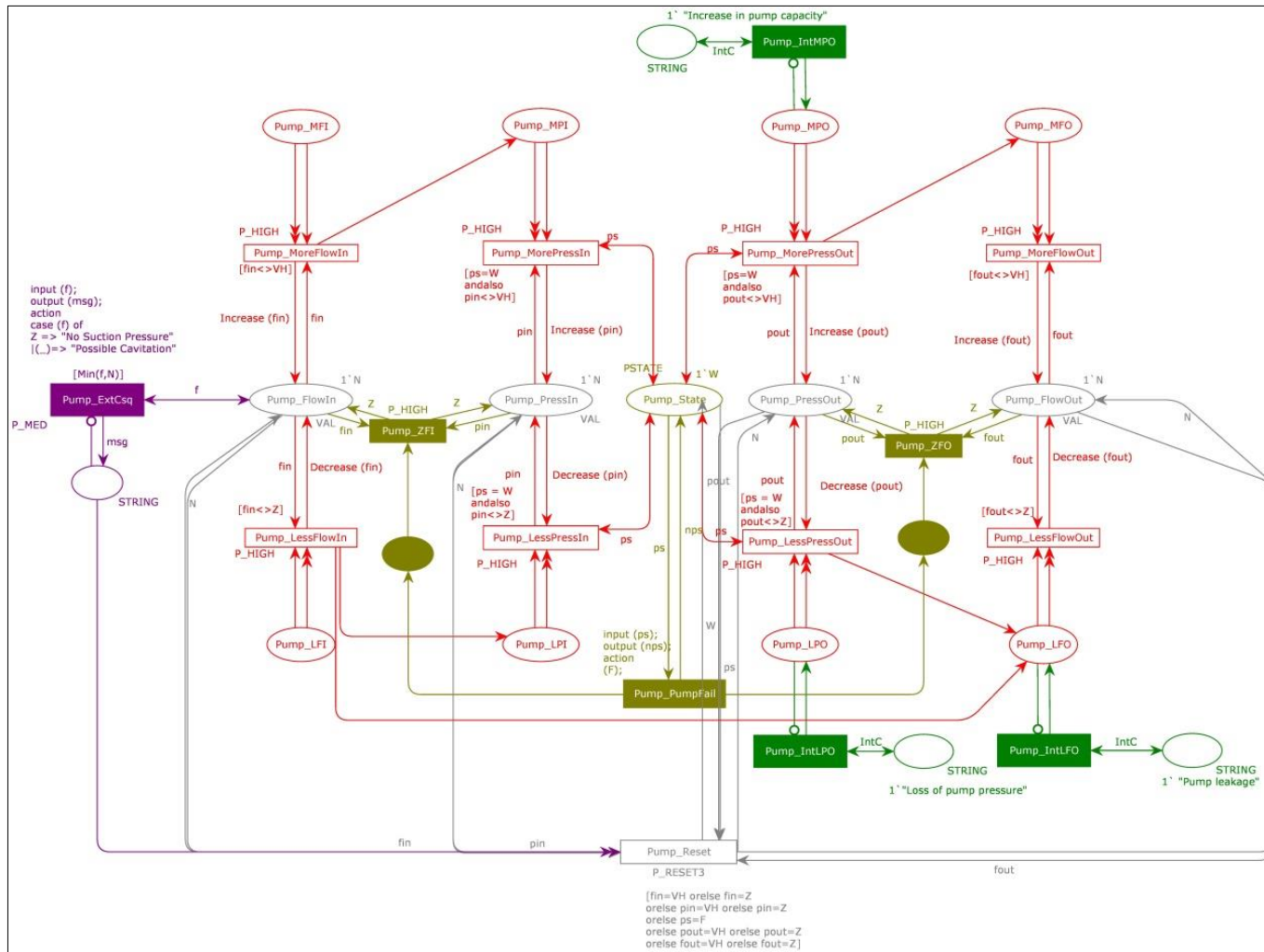


Figure 56 CPN model of the Pump

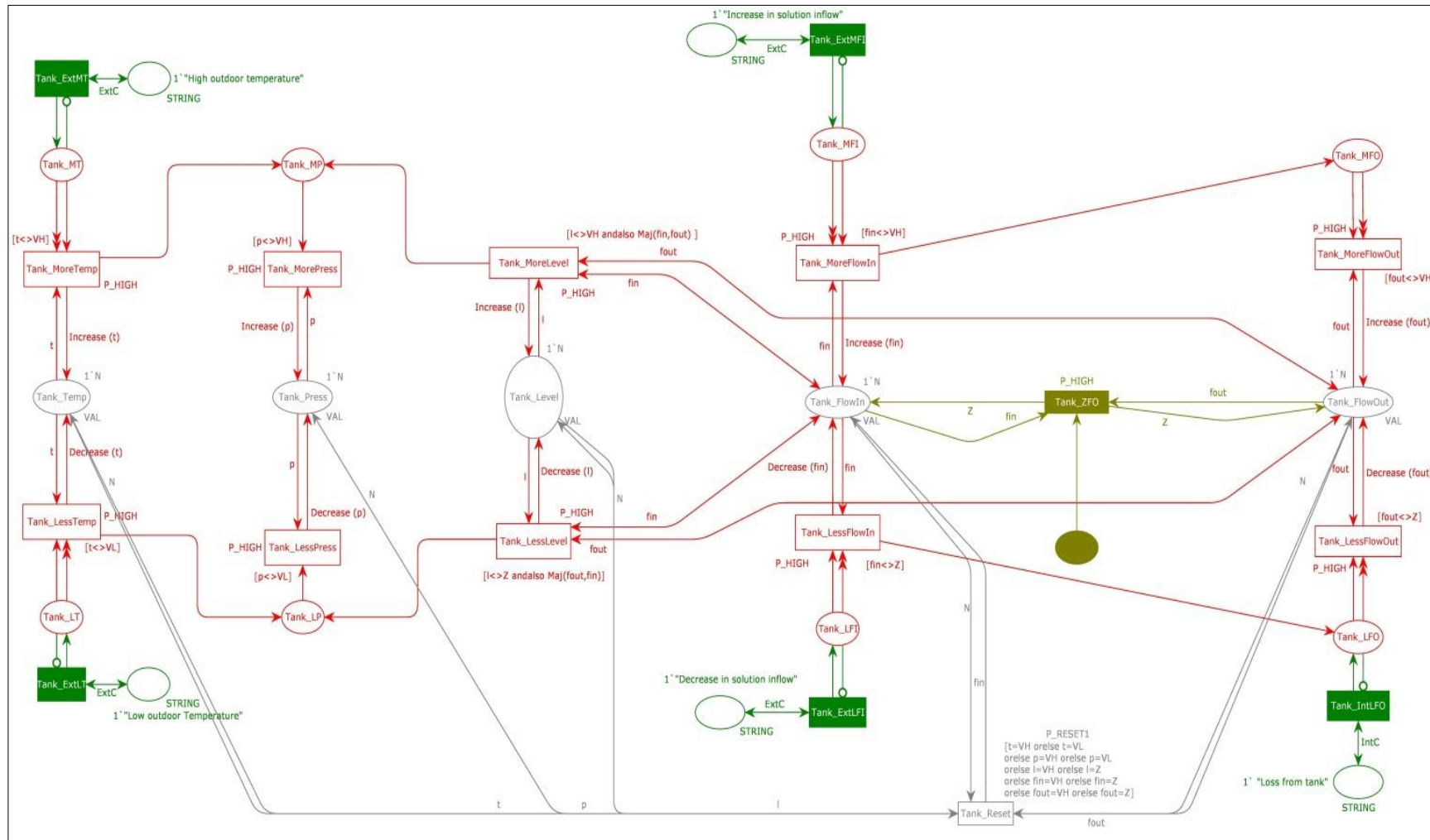


Figure 57 CPN model of the Vessel

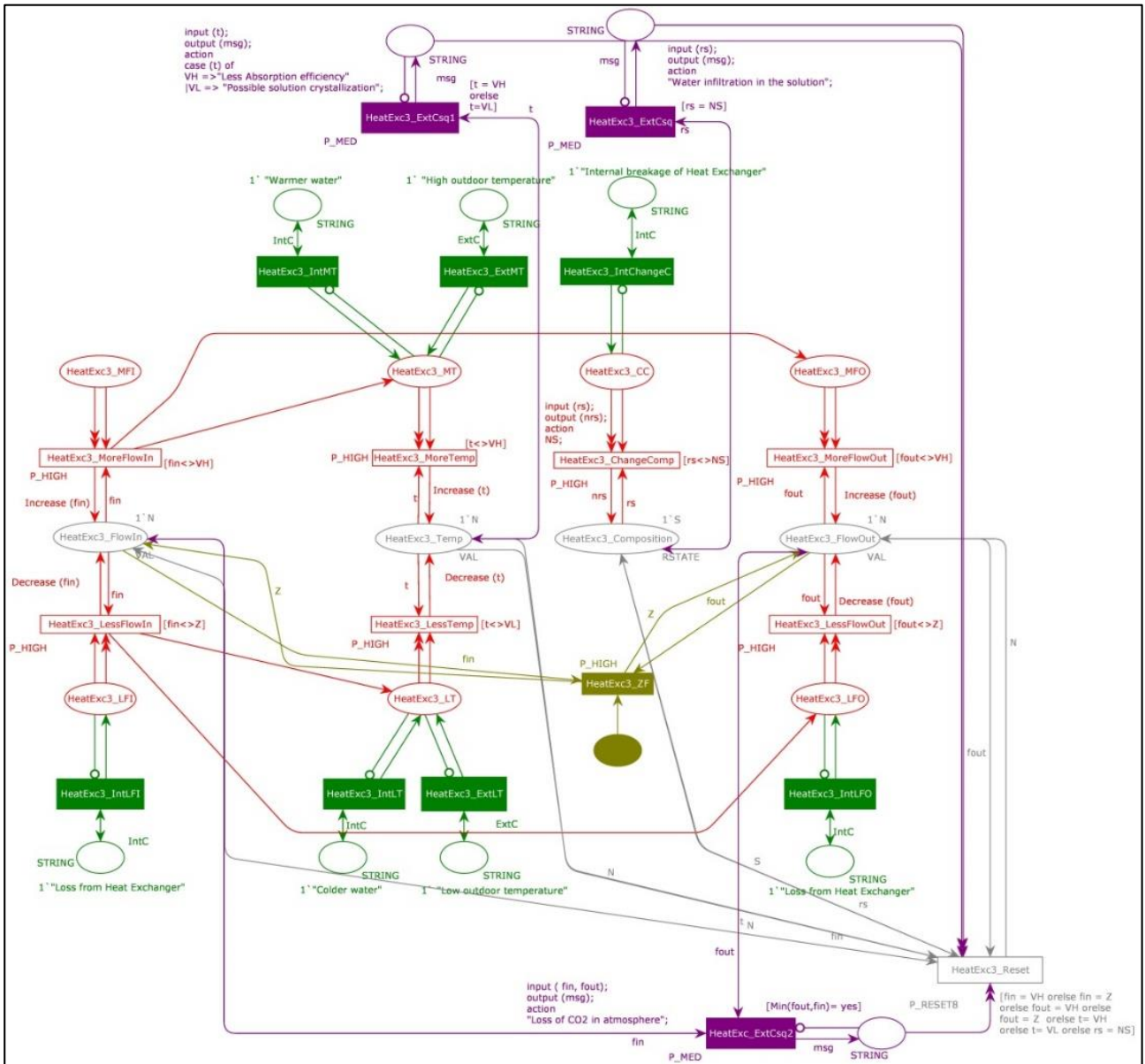


Figure 58 CPN Model of the Heat-Exchanger

Enhancing the CPN model with an OWL Ontology

With the aim supporting context “awareness”, thus, transforming the current CPN models into Context-adaptive Petri nets (CAPNs), and ontological model called CPN-HZP has been designed. The CPN-HZP Ontology (prefix: CPNHZP³⁸) is a BFO³⁹ compliant ontology that represents the context under which the HAZOP analysis is carried out by the CPN tools. In this framework, we therefore described the main element of the HAZOP context as follows: A *Process Variable Deviation Cause Event* [CPNHZP:001] *Contributes to Condition* [RO⁴⁰:0003304] *Process Variable Deviation from Expected State* [CPNHZP:002], which *inheres in* [RO:0000052] *HAZOP Unit* [CPNHZP:003] and eventually *results in* [CPNHZP:004] a *Process Variable Deviation Consequence Event* [CPNHZP:005] (the OWL ontology in Figure 59).

³⁸ CPNHZP URI - <http://semanticweb.org/cpnhzp.owl>

³⁹ Basic Formal Ontology IRI - <http://purl.obolibrary.org/obo/bfo.owl>

⁴⁰ Relations Ontology IRI - <http://purl.obolibrary.org/obo/ro.owl>

Table 23 introduces all the subclasses of the class *Process Variable Deviation Cause Event* [CPNHZP:001], such as *Filter_ExtMT* [CPNHZP:119] that refers to the external events causing an increase of temperature in the filter, e.g. *High Outdoor Temp.*

Table 24 introduces all the instances of the class *Process Variable Deviation from Expected State* [CPNHZP:002], e.g. *MT* [CPNHZP:207] that refers to *More Temperature*.

Then, the list of *HAZOP Unit* [CPNHZP:003] is shown in Table 25, whereas each unit under analysis will be instantiated in related class, e.g. the specific filter that we analyse in the context of this HAZOP study will be an instance of the class *Filter* [CPNHZP:303], which is a sub-class of the class *HAZOP Unit* [CPNHZP:003].

As a result, each piece of information will be linked to one another in a way to provide the system with a logic that can be leveraged for further machine reasoning semantics-driven analysis (Figure 60).

Table 23 Process Variable Deviation Cause Event [CPNHZP:001]

OWL Class	Short URI	HAZOP Unit Type	Origin	Type of deviation
Valve_ExtMFI	CPNHZP:101	Valve	External Cause	More Flow In
Valve_ExtLFI	CPNHZP:102	Valve	External Cause	Less Flow In
Valve_ExtOF	CPNHZP:103	Valve	External Cause	Open-Failed
Valve_IntOF	CPNHZP:104	Valve	Internal Cause	Open-Failed
Valve_ExtCF	CPNHZP:105	Valve	External Cause	Closed-Failed
Valve_IntCF	CPNHZP:106	Valve	Internal Cause	Closed-Failed
Valve_IntLFO	CPNHZP:107	Valve	Internal Cause	Less Flow Out
Reactor_ExtLFO	CPNHZP:108	Reactor	External Cause	Less Flow Out
Reactor_ExtLL	CPNHZP:109	Reactor	External Cause	Less Level
Reactor_ExtMT	CPNHZP:110	Reactor	External Cause	More Temperature
Reactor_IntMT	CPNHZP:111	Reactor	Internal Cause	More Temperature
Reactor_ExtLT	CPNHZP:112	Reactor	External Cause	Less Temperature
Reactor_IntLT	CPNHZP:113	Reactor	Internal Cause	Less Temperature
Reactor_ExtMT	CPNHZP:114	Reactor	External Cause	More Pressure
Reactor_ExtLT	CPNHZP:115	Reactor	External Cause	Less Pressure
Reactor_IntLT	CPNHZP:116	Reactor	Internal Cause	Less Pressure
Filter_ExtMFI	CPNHZP:117	Filter	External Cause	More Flow In
Filter_ExtLFI	CPNHZP:118	Filter	External Cause	Less Flow In
Filter_ExtMT	CPNHZP:119	Filter	External Cause	More Temperature
Filter_IntMT	CPNHZP:120	Filter	Internal Cause	More Temperature
Filter_ExtLT	CPNHZP:121	Filter	External Cause	Less Temperature
Filter_IntLT	CPNHZP:122	Filter	Internal Cause	Less Temperature
Filter_IntLFO	CPNHZP:123	Filter	Internal Cause	Less Flow Out
Pump_IntMPO	CPNHZP:124	Pump	Internal Cause	More Pressure Out
Pump_IntLPO	CPNHZP:125	Pump	Internal Cause	Low Pressure Out
Pump_IntLFO	CPNHZP:126	Pump	Internal Cause	Low Flow Out
Tank_ExtMT	CPNHZP:127	Tank	External Cause	More Temperature
Tank_ExtLT	CPNHZP:128	Tank	External Cause	Less Temperature
Tank_ExtMFI	CPNHZP:129	Tank	External Cause	More Flow In
Tank_ExtLFI	CPNHZP:130	Tank	External Cause	Less Flow In
Tank_IntLFO	CPNHZP:131	Tank	Internal Cause	Less Flow Out
HeatExc_IntLFI	CPNHZP:132	Heat Exchanger	Internal Cause	Less Flow In
HeatExc_IntMT	CPNHZP:133	Heat Exchanger	Internal Cause	More Temperature
HeatExc_ExtMT	CPNHZP:134	Heat Exchanger	External Cause	More Temperature
HeatExc_IntLT	CPNHZP:135	Heat Exchanger	Internal Cause	Less Temperature
HeatExc_ExtLT	CPNHZP:136	Heat Exchanger	External Cause	Less Temperature
HeatExc_IntCC	CPNHZP:137	Heat Exchanger	Internal Cause	Composition Change
HeatExc_IntLFO	CPNHZP:138	Heat Exchanger	Internal Cause	Less Flow In

Table 24 Process Variable Deviation from Expected State [CPNHZP:002]

OWL instance	Short URI	Type of Deviation
MFI	CPNHZP:201	More Flow In
LFI	CPNHZP:202	Less Flow In
OF	CPNHZP:203	Open-Failed
CF	CPNHZP:204	Closed-Failed
LFI	CPNHZP:205	Less Flow Out
LL	CPNHZP:206	Less Level
MT	CPNHZP:207	More Temperature
LT	CPNHZP:208	Less Temperature
MP	CPNHZP:209	More Pressure
LP	CPNHZP:210	Less Pressure
MPO	CPNHZP:211	More Pressure Out
LPO	CPNHZP:212	Low Pressure Out
LFO	CPNHZP:213	Low Flow Out
CC	CPNHZP:214	Composition Change

Table 25 HAZOP Unit [CPNHZP:003]

OWL Class	Short URI	HAZOP Unit
Valve	CPNHZP:301	Valve
Reactor	CPNHZP:302	Reactor
Filter	CPNHZP:303	Filter
Pump	CPNHZP:304	Pump
Tank	CPNHZP:305	Tank
HeatExc	CPNHZP:306	Heat Exchanger

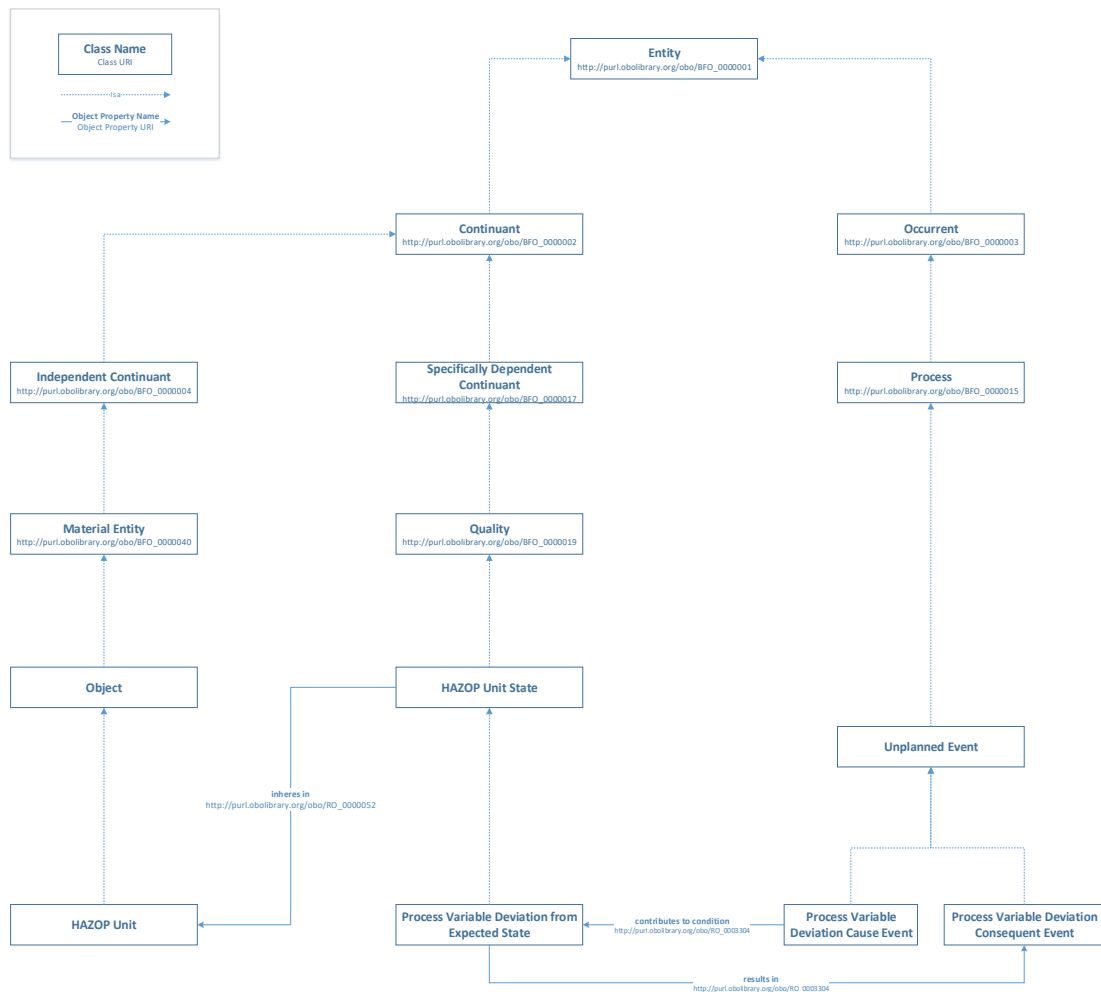


Figure 59 CPN-HZP Ontology

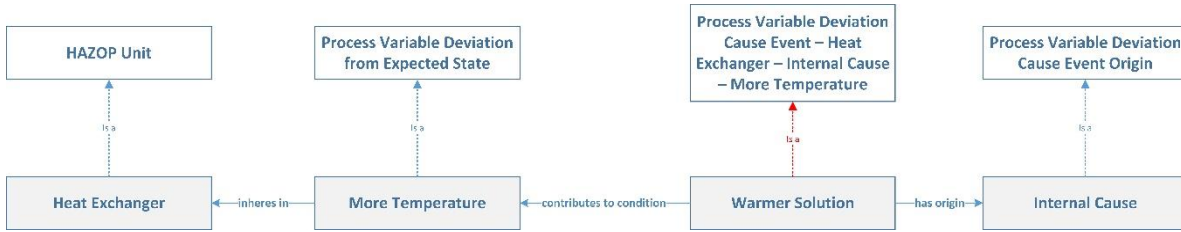


Figure 60 Machine reasoning-based OWL class assignment to “Warmer Solution”

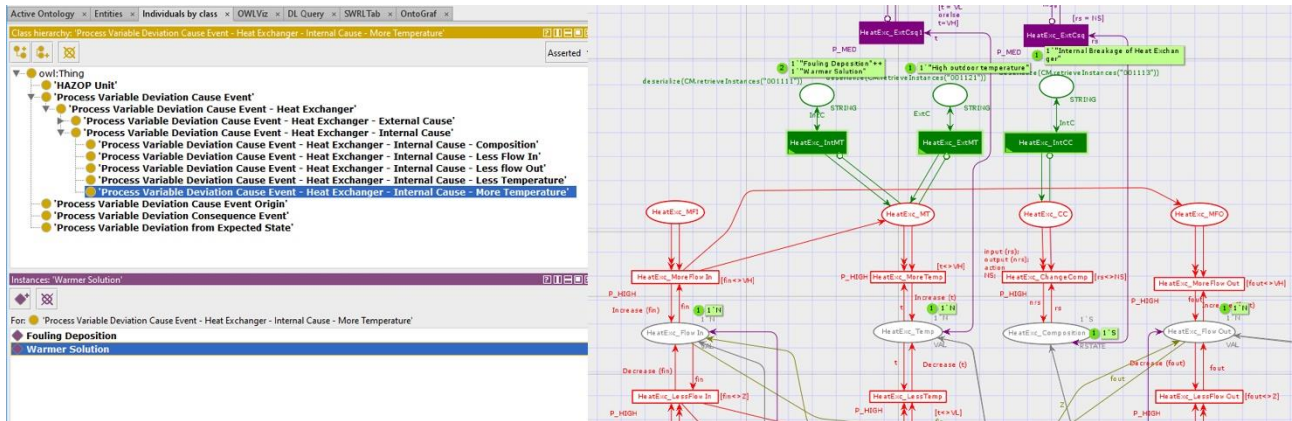


Figure 61 (Left side) Context Linked Data on Protégé; (Right Side) CPN model with Context Linked Data

Simulation results

Among the functionalities provided by CPN Tools, we can find the State-Space analysis, i.e. the analysis of the states that can be reached by the Coloured Petri Net along with a series of properties related to the latter. Unfortunately, such a tool is very resource (time and computation effort) consuming. This is why we opt in for an automatic simulation-based analysis, which produces a simulation report that can be filtered in a way to obtain a HAZOP-like report. The depth (roughly related to the number of steps) of the simulation is set during a testing phase. 10000 steps can be executed in around 10 seconds, assuring the complete coverage of the all state space, thus, providing an exhaustive list of all the possible reachable states. As a result, the simulation tool produces a text report that looks like Figure 62. Data is therefore imported in Excel and filtered by a VBA macro which, as a first step, sort out all the rows as shown in Figure 63. Internal Causes are highlighted in red, the External ones in (yellow) the – chain of – consequences, instead, are highlighted in green. This mid filtering step allows the user to identify already all the effects (responses of the system) to all the foreseen unplanned events *Process Variable Deviation Cause Event* [CPNHZP:001]. Then, the HAZOP-like report is also generated (Figure 64), hence, presenting the causes and consequences of any meaningful combination of guidewords and parameters.

```

1 Valve_ExtOF
- ExtC = "Manually open valve"
- vs = C
- nvs = FO
2 Valve_MoreFlowOut
- fout = N
3 Reactor_MoreFlowIn
- fin = N
4 Reactor_IntML
- fin = H
- fout = N
5 Reactor_MoreLevel
- l = N
6 Reactor_IntML
- fin = H
- fout = N
7 Reactor_MoreFlowOut
- fout = N
8 Reactor_MoreLevel
- l = H
9 HeatExc_MoreFlowIn
- fin = N
10 HeatExc_MoreFlowOut
- fout = N
11 HeatExc_MoreTemp
- t = N
12 HeatExc2_MoreFlowIn
- fin = N
13 HeatExc2_MoreTemp
- t = N
14 HeatExc2_MoreFlowOut
- fout = N
15 Valve2_MoreFlowIn
    
```

Figure 62 Simulation Report

STEP	COMPONENT	ACTION	BINDING		
			INPUT	OUTPUT	
1	Valve	ExtOF	vs = C	ExtC = "Manually open valve"	nvs = FO
2	Valve	MoreFlowOut	fout = N		
3	Reactor	MoreFlowIn	fin = N		
4	Reactor	IntML	fin = H	fout = N	
5	Reactor	MoreLevel	l = N		
6	Reactor	IntML	fin = H	fout = N	
7	Reactor	MoreFlowOut	fout = N		
8	Reactor	MoreLevel	l = H		
9	HeatExc	MoreFlowIn	fin = N		
10	HeatExc	MoreFlowOut	fout = N		
11	HeatExc	MoreTemp	t = N		
12	HeatExc2	MoreFlowIn	fin = N		
13	HeatExc2	MoreTemp	t = N		
14	HeatExc2	MoreFlowOut	fout = N		
15	Valve2	MoreFlowIn	vs = C	fin = N	
16	Valve2	MoreFlowOut	fout = N		
17	Reactor	ExtCsq2	l = VH	msg = "Possible solution crystallization"	
18	Reactor	IntMT	IntC = "Warmer gas"		
19	Reactor	MoreTemp	t = N	p = N	
20	Reactor	MoreReaction	r = N		
21	Reactor	MorePress	p = N		
22	Reactor	LessReaction	r = H		

Figure 63 Processing the simulation report (Causes and Consequences highlighted)

PARAMETER	GUIDEWORD	CAUSE	CONSEQUENCE
Flow	More	Valve ExtC = "Manually open valve"	Valve MoreFlowOut
			Reactor MoreFlowIn
		Valve2 IntC = "Locked-open valve"	Valve2 MoreFlowOut
		Valve IntC = "Locked-open valve"	Valve MoreFlowOut
			Reactor MoreFlowIn
			Reactor MoreFlowOut
			HeatExc MoreFlowIn
			HeatExc MoreFlowOut
			HeatExc MoreTemp
			HeatExc2 MoreFlowIn
			HeatExc2 MoreTemp
			HeatExc2 MoreFlowOut
			Valve2 MoreFlowIn
		Valve2 ExtC = "Manually open valve"	Valve2 MoreFlowOut
		Valve ExtC = "Increase in gas inflow"	Valve MoreFlowIn
			Valve MoreFlowOut
			Reactor MoreFlowIn
Flow	Less	Valve IntC = "Leakage"	Valve LessFlowOut
			Reactor LessFlowIn
		HeatExc IntC = "Loss from Heat Exchanger"	HeatExc LessFlowIn
			HeatExc LessFlowOut
			HeatExc2 LessFlowIn
			HeatExc2 LessTemp
			HeatExc2 LessFlowOut
			HeatExc LessTemp
			Valve2 LessFlowIn
			Valve2 LessFlowOut
			Low flow of fuel to the engines

Figure 64 HAZOP-like report generated from the Simulation Results

As presented in [126], there are many studies focused on creating a model to estimate the duration of a HAZOP analysis. Most of them are based on the verification of the precision of the mathematical model comparing the results with the timing regarding previous case studies. According to the model introduced by [134], the variables identified as the most impactful are:

- Size of the team;
- Previous experience in the HAZOP team;
- Assigned time for the study;
- Proper planning;
- Participation of team members;
- Availability of information.

The analysis of the performance of the case study here presented mainly focuses on the following four aspects:

- Analysis of the preliminary information;
- Brainstorming discussion;
- Discussion over unclear points;
- Report writing.

The time to complete the HAZOP analysis of the case study estimated by using the approach proposed in [134] (excluding the report writing) is about 18 hours. In this work instead - starting from the reasonable assumption that all the basics models all already included in CPN library for industrial components and HAZOP analysis – the time required to obtain the aforementioned HAZOP-like report is estimated to be:

- about 15 minutes for assembling each component model (time tests executed on users with different experience; results ranging from 5 to 25 minutes);
- about 5 minutes per component to check and eventually updated the knowledge base with all the unplanned events (causes and consequences);
- around 1 minute to run the simulation and collect the results;
- about 2 minutes for filtering data and getting the HAZOP report.

As a result, a medium-high complexity system of 6 components has been analysed, which requires about 123 minutes to be simulated and to generate the HAZOP report.

$$T1 = N_{\text{Components}} \times (T_{\text{Assembly}} + T_{\text{CausesConsequences}}) + T_{\text{Simulation}} + T_{\text{VBA}} =$$

$$= 6 \times (15+5) + 1 + 2 = 123 \text{ min} = 2.05 \text{ h}$$

Equation 15 HAZOP reporting time evaluation

This represents a reduction of about 89% of the time required to carry out a preliminary analysis supporting the HAZOP experts in such a time-consuming task.

6.3.4 Discussions around the application case

The use of a Petri Net to model and analyse fault propagations in industrial plants has extensively proved to be effective in literature. The approach described in this application case extends the one presented in two research works published by the author of this dissertation in the past few years [127] [126]. In particular, the CPN model has been enriched by adding new types of failure, components and component states, therefore, widening the application range of this method. By leveraging one of the most accurate mathematical models for HAZOP analysis time estimation, a reduction of about 89% of the total time to complete the analysis of a medium complexity case study using the proposed solution based on CPN-HAZOP has been shown. In both new and traditional approaches, the reporting phase was not considered. The HAZOP study resulting from the translating macro does not represent the final step of the analysis, but it is a useful and quick guide for the team of experts during the brainstorming phase.

The automation introduced in the HAZOP technique and the significant reduction of time required for the analysis could also allow the method to be disseminated to less burdensome contexts compared to typical HAZOP analysis application (e.g. plants at risk of a major accident subject to the Seveso Directive). In fact, the proposed approach would allow more accurate risk analysis also in smaller and less complex process plants, where the limited number of employers and, in some cases, their competencies are not consistent with a traditional HAZOP study process.

Further research is required in order to extend the library with more CPN-based component models, to translate the information into a more detailed user-friendly report, and to integrate the so-called “safeguards” in the CPN model. Eventually, the next developments of this smart tool should also include the introduction of stochastic data values in order to better simulate the occurrence of the likelihood of failure causes and allow a semi-quantitative risk assessment.

Chapter 7 Conclusion

This dissertation proposes a framework for applying state-of-the-art methods for semantic data and ontology management in the field of manufacturing systems modelling and simulation. The benefits and limitations of such a proposed framework are thoroughly discussed through the analysis of three application cases addressing diverse aspects of the domain in question: workforce allocation, manufacturing knowledge management, quantitative reliability assessment for an assembly station, as well as qualitative risk analysis for a petrochemical plant facing serious hazard and operability issues.

In the context of modelling and simulation for manufacturing system applications, the use of semantic web technologies as a framework has not yet been fully leveraged. Through the proposed approach, it is possible to outline a list of principles and best practices that will help future efforts towards the application of complex digital twin modelling and simulation-oriented solutions built on semantically-enriched manufacturing information systems.

The methodological approach adopted here is, therefore, designed on a set of technical principles and activities which characterize the development of each of the domain-specific ontologies presented in this work:

1. Clear definition of context and scope of the representation
2. Selection of the formats and related serialization
3. Analysis and reuse of the existing ontologies in the domain
4. Creation of each new entity starting from an Upper Ontology
5. Provision of textual definitions for each entity, using existing standards
6. Setting up of unique Identifiers & Naming Conventions for each new entity
7. Provision of a logic along with the set of domain specific entities to foster reasoning and machine-based inference

Therefore, each solution presented in the application cases above is built upon these principles and aim to – partly or fully – answer the three research questions presented in Section 2.2 and repeated below:

- RQ1. How can we **semantically enrich** manufacturing system models and exploit **context-awareness** for process analysis?
- RQ2. How can semantics be used to foster **reusability** and **interoperability** of manufacturing systems models?
- RQ3. How does the analysis of semantically-enriched manufacturing systems **impact on modelling and simulation** applications?

The first application, designed and analysed at the beginning of this research work, contributes to the investigation of ontology modelling and analytics issues related to the exploitation of semantically-enriched data, paving the way towards answering RQ1 and RQ2. As a result of the deployment of a so-called *semantic framework for human resource management through semantically-enriched data*, it is possible to draw two possible conclusions. Firstly, the semantic enrichment of historical and real-time information from shop floor work/task scheduling provides support for an intelligent system, which does not replace the existing information systems but rather empowers the latter with reasoning capabilities and advanced analytics capabilities. Secondly, a real-time task scheduling method towards HR optimization by utilizing Conditional Random Field (CRF) probabilistic models, semantically-enriched information, and semantic query and rule languages, namely SPARQL and SWRL, has been proven to be effective, although, some considerations should be done around the scalability of the proposed approach.

The second application contributes to existing knowledge by answering (almost exhaustively) all three RQs. The findings, indeed, span aspects such as the semantic enrichment of manufacturing system models, the reusability of modelling primitives, the impact of semantically-enriched manufacturing system models in modelling and simulation (M&S) applications. Here, two key aspects are addressed by the proposed ontology-based solution: (i) ontology-driven creation and semantic enrichment of the manufacturing system's modelling elements (or modelling primitives of a specific modelling language, such as Petri Net formalism); (ii) semantics-driven analysis of a digital twin's simulation outcomes aimed at enhancing the decision-making process. The study of an ontology-based solution for reliability assessment of an automated assembly station demonstrates how effectively such a solution can support manufacturing system's data collection, manufacturing process modelling, simulation, and simulation results analysis.

The third application has been presented with the aim of further investigating research questions 1 and 3. The proposed solution does not introduce any new assessment paradigms but rather combines existing technologies (Petri nets and Ontologies) with the purpose of rendering the well-known, as well as, time consuming methodology called HAZOP more effective and less costly. While the use of PNs to model and analyse fault propagations in industrial plants has been already documented in literature, the employment of ontologies to empower such a PN-based manufacturing system model with context-awareness capabilities has been here presented. The proposed solution has proven to reduce about 89% of the time required to carry out a preliminary analysis, thus supporting the Hazard & Operability (HAZOP) experts in such a time-consuming task.

Finally, while recognising some limitations of the second application case, the exploratory research here presented lays the initial groundwork for future research. Starting from the observations above, this dissertation provides a new angle of looking at the digital transformation – key enabler of future modelling and simulation applications – for manufacturing systems, therefore, paving the way towards the development of a new analytical framework for semantics-driven modelling and simulation of context-aware manufacturing systems. Recommendations for further research include:

- Investigation of emerging software tools and standards for Model Based System Engineering (MBSE) solutions, which leverage ontologies and semantic reasoning for manufacturing system modelling and simulation.
- In depth analysis of alternative languages such as SysML, perhaps, by comparing this with the features of the PN formalism presented in this work. The level of adoption should not be the only indicator for establishing the “right” language to be used, but rather the range of applicability of the latter should be.
- Alignment to the most recent developments in Ontology Engineering for manufacturing. In this regard initiatives such as the Industrial Ontologies Foundry (IOF) certainly represents a good source of information and an environment where these aspects can be all discussed and further refined.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

References

- [1] K. Schwab, "The Fourth Industrial Revolution What It Means and How to Respond," 2015.
- [2] F. Erhard, H. Markus, U. Rainer, I. Daniele, and B. Jörg, "Industry 4.0 at McKinsey's model factories Get ready for the disruptive wave," *McKinsey*, 2016.
- [3] R. C. Schlaepfer and M. Koch, "Industry 4.0: Challenges and solutions for the digital transformation and use of exponential technologies," 2015.
- [4] A. Parrott and W. Lane, "Industry 4.0 and the digital twin," *Deloitte Univ. Press*, pp. 1–17, 2017.
- [5] H. Burkhardt and B. Smith, *Handbook of metaphysics and ontology*. Philosophia Verlag, 1991.
- [6] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquis.*, vol. 5, no. 2, pp. 199–220, Jun. 1993.
- [7] W. N. Borst and W. N. Borst, "Construction of Engineering Ontologies for Knowledge Sharing and Reuse," Sep. 1997.
- [8] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge engineering: Principles and methods," *Data Knowl. Eng.*, vol. 25, no. 1–2, pp. 161–197, Mar. 1998.
- [9] R. Arp, B. Smith, and A. D. Spear, *Building ontologies with basic formal ontology*. 2015.
- [10] M. Gruninger and M. Uschold, "Ontologies: Principles, Methods and Applications," 1996.
- [11] M. Uschold, "Building Ontologies: Towards a Unified Methodology," in *Proceedings of Expert Systems '96, the 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems*, 1996, pp. 16–18.
- [12] A. Bernaras, I. Laresgoiti, and J. M. Corera, "Building and Reusing Ontologies for Electrical Network Applications," in *European Conference on Artificial Intelligence (ECAI)*, 1996.
- [13] A. Gómez-Pérez, "Towards a Framework to Verify Technology Knowledge Sharing," *Expert Syst. Appl.*, vol. 11, no. 4, pp. 519–529, 1996.
- [14] M. Fernández-López, A. Gómez-Pérez, and N. Juristo, "METHONTOLOGY: From Ontological Art Towards Ontological Engineering," *Proc. Ontol. Eng. AAAI-97 Spring Symp. Ser. | AAAI-97 Spring Symp. Ser. | 24-26 March 1997 | Stanford Univ. EEUU*, 1997.
- [15] A. Gómez-Pérez, "Knowledge Sharing and Reuse," in *The Handbook of APPLIED EXPERT SYSTEMS*, 1997.
- [16] M. F. Lopez, A. Gomez-Perez, J. P. Sierra, and A. P. Sierra, "Building a chemical ontology using Methontology and the Ontology Design Environment," *IEEE Intell. Syst.*, vol. 14, no. 1, pp. 37–46, Jan. 1999.
- [17] B. Swartout, R. Patil, K. Knight, T. R.-P. of the T. W. on, and undefined 1996, "Toward distributed use of large-scale ontologies," *aaai.org*.
- [18] M. Fernández López and F. López, "Overview Of Methodologies For Building Ontologies," in *Proceedings of the IJCAI-99 workshop on Ontologies and Problem-Solving (KRR5)*, 1999.
- [19] G. Schreiber, B. Wielinga, and W. Jansweijer, "The KACTUS View on the 'O' Word," in *Proceedings of the IJCAI workshop on basic ontological issues in knowledge sharing*, 1995.
- [20] K. Knight and S. K. Luk, "Building a Large-Scale Knowledge Base for Machine Translation," 1994.
- [21] K. Kevin *et al.*, "Filling knowledge gaps in a broad-coverage machine translation system," 1995.
- [22] A. Gómez-Pérez, ... M. F.-L.-... K. M., and undefined 2004, "Methodologies and methods for building ontologies," *Springer*.
- [23] Y. Sure, S. Staab, and R. Studer, "On-To-Knowledge Methodology (OTKM)," in *Handbook on Ontologies*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 117–132.
- [24] H. Pinto, S. Staab, C. T.-P. of the 16th, and undefined 2004, "DILIGENT: Towards a fine-grained methodology for distributed, loosely-controlled and evolving," *books.google.com*.
- [25] M. del C. Suárez de Figueroa Baonza, "NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse," 2010.

- [26] B. Smith, "On classifying material entities in Basic Formal Ontology," 2012.
- [27] R. Arp, B. Smith, and A. Spear, "Building ontologies with basic formal ontology," 2015.
- [28] D. P. Hill, B. Smith, M. S. McAndrews-Hill, and J. A. Blake, "Gene Ontology annotations: what they mean and where they come from," in *Proceedings of the 10th Bio-Ontologies Special Interest Group Workshop 2007. Ten years past and looking to the future*, 2008, vol. 9, no. Suppl 5, p. S2.
- [29] M. Almeida *et al.*, "Basic Formal Ontology 2.0 - Specification and user's guide," 2015.
- [30] M. Kamath and N. Viswanadham, "Applications of petri net based models in the modelling and analysis of flexible manufacturing systems," in *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, vol. 3, pp. 312–317.
- [31] A. A. (Alan A. . Desrochers, R. Y. Al-Jaar, and IEEE Control Systems Society., *Applications of petri nets in manufacturing systems : modeling, control, and performance analysis*. IEEE Press, 1995.
- [32] F. DICESARE and A. A. DESROCHERS, "Modeling, Control, and Performance Analysis of Automated Manufacturing Systems Using Petri Nets," *Control Dyn. Syst.*, vol. 47, pp. 121–172, Jan. 1991.
- [33] Ö. Başak and Y. E. Albayrak, "Petri net based decision system modeling in real-time scheduling and control of flexible automotive manufacturing systems," *Comput. Ind. Eng.*, vol. 86, pp. 116–126, 2014.
- [34] J. L. Peterson and J. Lyle, *Petri net theory and the modeling of systems*. Prentice-Hall, 1981.
- [35] T. Murata, "Petri Nets: Properties, Analysis and Applications," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, 1989.
- [36] C. A. Petri, "Kommunikation mit Automaten," *Fakultät für Mathematik und Physik*, vol. Doktor. p. 128, 1962.
- [37] P. Merlin, "A Study of the Recoverability of Computer Systems," University of California, 1974.
- [38] C. Ramchandani, "Analysis of asynchronous concurrent systems by timed Petri nets." p. 225, 1974.
- [39] C. Kahraman and F. Tüysüz, "Manufacturing system modeling using Petri Nets," *Prod. Eng. Manag. under Fuzziness*, pp. 95–124, 2010.
- [40] M. K. Molloy, "Performance Analysis Using Stochastic Petri Nets," *IEEE Trans. Comput.*, vol. C-31, no. 9, pp. 913–917, 1982.
- [41] R. Valette, J. Cardoso, and D. Dubois, "Monitoring manufacturing systems by means of Petri nets with imprecise markings," in *IEEE Conference*, 1989, p. 6.
- [42] K. Valavanis, "On the hierarchical analysis and simulation of flexible manufacturing systems with extended Petri nets," *IEEE Trans. on Systems, Man, and Cybernetics*, pp. 94–100, 1990.
- [43] M. Marsan, "Stochastic petri nets: an elementary introduction," *Adv. Petri nets 1989*, p. 29, 1990.
- [44] K. Jensen, *Coloured Petri Nets Basic Concepts, Analysis Methods and Practical Use*. Springer-Verlag Berlin Heidelberg, 1992.
- [45] W. M. P. Van der Aalst, "State Space Analysis: Properties, Reachability Graph, and Coverability graph," 2011. .
- [46] M. Zhou and K. Venkatesh, *Modeling, simulation, and control of flexible manufacturing systems: a Petri net approach*, First. World Scientific Publishing, 1999.
- [47] R. Y. Al-Jaar and A. A. Desrochers, "Performance evaluation of automated manufacturing systems using generalized stochastic petri nets," *IEEE Trans. Robot. Autom.*, vol. 6, no. 6, pp. 621–639, 1990.
- [48] A. Bobbio, "System modelling with Petri nets," *Syst. Reliab. Assess. Proc. Ispra course held Esc. Tec. Super. Ing. Nav. Madrid, Spain, Sept. 19-23, 1988, Collab. with Univ. Politec. Madrid*, p. 103, 1990.
- [49] F. DiCesare, G. Harhalakis, J. M. Proth, M. Silva, and F. B. Vernadat, "Practice of Petri Nets in Manufacturing," *J. Oper. Res. Soc.*, vol. 45, no. 9, pp. 1094–1094, 1993.
- [50] D. Coman, A. Ionescu, and M. Florescu, "Manufacturing System Modeling Using Petri Nets," *Behaviour*, vol. 1, no. 3, p. 2, 2009.
- [51] M. Ajmone Marsan, A. Bobbio, and S. Donatelli, "Petri nets in performance analysis: An introduction," *Lect. Petri Nets I Basic Model.*, pp. 211–256, 1998.
- [52] H. Kaid, A. M. El-tamimi, E. A. Nasr, S. Arabia, and S. Arabia, "Applications of Petri nets Based Models in Manufacturing Systems : A Review," no. 1, pp. 516–528, 2015.
- [53] F. Tuysuz and C. Kahraman, "Modeling a flexible manufacturing cell using stochastic Petri nets with fuzzy parameters," *Expert Syst. Appl.*, vol. 37, no. 5, pp. 3910–3920, 2010.
- [54] I. Ben Abdallah, H. A. Elmaraghy, and T. Elmekawy, "Deadlock-free scheduling in flexible manufacturing systems using Petri

- nets," *Int. J. Prod. Res.*, vol. 40, no. 12, pp. 2733–2756, 2002.
- [55] R. A. Wysk, N. Yang, C. Li, and S. Joshi, "Resolution of Deadlocks in Flexible Manufacturing Systems: Avoidance and Recovery Approaches," *J. Manuf. Syst.*, vol. 13, no. 2, pp. 128–138, 1994.
- [56] Y. Tat Leung and G. J. Sheen, "Resolving deadlocks in flexible manufacturing cells," *J. Manuf. Syst.*, vol. 12, no. 4, pp. 291–304, 1993.
- [57] M. P. Fantì, C. Maione, P. Bari, and V. David, "Deadlock Detection and Recovery in Flexible Production Systems with multiple Capacity Resources," pp. 237–241, 1996.
- [58] M. Grüninger and C. Menzel, "The Process Specification Language (PSL) Theory and Applications," *AI Mag.*, 2003.
- [59] L. M. Deshayes, O. El Beqqali, and A. Bouras, "The use of process specification language for cutting processes," *Int. J. Prod. Dev.*, vol. 2, no. 3, p. 236, 2005.
- [60] F. Ameri, C. Urbanovsky, and C. McArthur, "A Systematic Approach to Developing Ontologies for Manufacturing Service Modeling."
- [61] L. Lacy and W. Gerber, "Potential Modeling and Simulation Applications of the Web Ontology Language - OWL," in *Proceedings of the 2004 Winter Simulation Conference, 2004.*, vol. 1, pp. 257–262.
- [62] P. Leitão and F. Restivo, "ADACOR: A holonic architecture for agile and adaptive manufacturing control," *Comput. Ind.*, vol. 57, no. 2, pp. 121–130, Feb. 2006.
- [63] R. F. Babiceanu and F. F. Chen, *Journal of intelligent manufacturing.*, vol. 17, no. 1. Chapman & Hall.
- [64] P. Leitao, V. Marik, and P. Vrba, "Past, Present, and Future of Industrial Agent Applications," *IEEE Trans. Ind. Informatics*, vol. 9, no. 4, pp. 2360–2372, Nov. 2013.
- [65] P. Vrba *et al.*, "Rockwell Automation's Holonic and Multiagent Control Systems Compendium," *IEEE Trans. Syst. Man, Cybern. Part C (Applications Rev.)*, vol. 41, no. 1, pp. 14–30, Jan. 2011.
- [66] P. Leitão, "Agent-based distributed manufacturing control: A state-of-the-art survey," *Eng. Appl. Artif. Intell.*, vol. 22, no. 7, pp. 979–991, Oct. 2009.
- [67] J. Barbosa, P. Leitão, E. Adam, and D. Trentesaux, "Dynamic self-organization in holonic multi-agent manufacturing systems: The ADACOR evolution," *Comput. Ind.*, vol. 66, pp. 99–111, Jan. 2015.
- [68] S. Lemaignan, A. Siadat, J.-Y. Dantan, and A. Semenenko, "MASON: A Proposal For An Ontology Of Manufacturing Domain," in *IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06)*, pp. 195–200.
- [69] H. Panetto, M. Dassisti, and A. Tursi, "ONTO-PDM: Product-driven ONTOlogy for Product Data Management interoperability within manufacturing process environment," *Adv. Eng. Informatics*, vol. 26, no. 2, pp. 334–348, Apr. 2012.
- [70] A. Tursi, H. Panetto, G. Morel, and M. Dassisti, "Ontological approach for products-centric information system interoperability in networked manufacturing enterprises," *Annu. Rev. Control*, vol. 33, no. 2, pp. 238–245, Dec. 2009.
- [71] ISO/TC 184/SC 4, "ISO 10303-21:2016 - Industrial automation systems and integration -- Product data representation and exchange -- Part 21: Implementation methods: Clear text encoding of the exchange structure."
- [72] ISO/TC 184/SC 5, "IEC 62264-1:2013 - Enterprise-control system integration -- Part 1: Models and terminology."
- [73] ISO/TC 184/SC 5, "IEC 62264-2:2015 - Enterprise-control system integration -- Part 2: Objects and attributes for enterprise-control system integration."
- [74] H. K. Lin and J. A. Harding, "A manufacturing system engineering ontology model on the semantic web for inter-enterprise collaboration," *Comput. Ind.*, vol. 58, no. 5, pp. 428–437, Jun. 2007.
- [75] L. Mazzola, P. Kapahnke, M. Vujic, and M. Klusch, "CDM-Core: A Manufacturing Domain Ontology in OWL2 for Production and Maintenance," in *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, 2016*, pp. 136–143.
- [76] M. Compton *et al.*, "The SSN ontology of the W3C semantic sensor network incubator group," *J. Web Semant.*, vol. 17, pp. 25–32, Dec. 2012.
- [77] A. Günel, A. Meshram, T. Bley, M. Klusch, and A. Schütze, "Statistical and Semantic Multisensor Data Evaluation for Fluid Condition Monitoring in Wind Turbines," in *In Proc. 16th Intl. Conf. on Sensors and Measurement Technology, Germany, 2013*.
- [78] N. Petersen *et al.*, "SCORVoc: Vocabulary-based Information Integration and Exchange in Supply Networks."
- [79] N. Petersen *et al.*, "SCORVoc: Vocabulary-Based Information Integration and Exchange in Supply Networks," in *2016 IEEE Tenth International Conference on Semantic Computing (ICSC), 2016*, pp. 132–139.

- [80] J. Billington *et al.*, "The Petri Net Markup Language: Concepts, Technology, and Tools," in *International Conference on Application and Theory of Petri Nets*, 2003, pp. 483–505.
- [81] ISO/IEC JTC 1/SC 7, "ISO/IEC DIS 15909-1 - Systems and software engineering -- High-level Petri nets -- Part 1: Concepts, definitions and graphical notation."
- [82] ISO/IEC JTC 1/SC 7, "ISO/IEC DIS 15909-1 - Software and Systems Engineering – High-level Petri Nets Part 2: Transfer Format," 2005.
- [83] M. Klein, D. Fensel, F. van Harmelen, and I. Horrocks, "The relation between ontologies and XML schemas," 2000.
- [84] Ma Bing-xian and Xu Ying-lei, "Integrating PNML with OWL for Petri nets," in *2009 2nd IEEE International Conference on Computer Science and Information Technology*, 2009, pp. 228–230.
- [85] D. Gašević and V. Devedžić, "Reusing Petri Nets Through the Semantic Web," Springer, Berlin, Heidelberg, 2004, pp. 284–298.
- [86] D. Gašević and V. Devedžić, "Petri net ontology," *Knowledge-Based Syst.*, vol. 19, no. 4, pp. 220–234, Aug. 2006.
- [87] J. Zedlitz, J. Jörke, and N. Luttenberger, "From UML to OWL 2," Springer, Berlin, Heidelberg, 2012, pp. 154–163.
- [88] D. Berardi, D. Calvanese, and G. De Giacomo, "Reasoning on UML class diagrams," *Artif. Intell.*, vol. 168, no. 1–2, pp. 70–118, Oct. 2005.
- [89] F. Silva, P. Parreiras, S. Staab, and A. Winter, "TwoUse TwoUse: Integrating UML Models and : Integrating UML Models and OWL OWL Ontologies Ontologies," 2007.
- [90] D. Gasevic, D. Djuric, V. Devedzic, and V. Damjanovi, "Converting UML to OWL ontologies," in *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters - WWW Alt. '04*, 2004, p. 488.
- [91] M. L. R. Varela and R. A. Ribeiro, "Distributed Manufacturing Scheduling Based on a Dynamic Multi-criteria Decision Model."
- [92] S. Perini, D. Arena, D. Kiritsis, and M. Taisch, *An ontology-based model for training evaluation and skill classification in an industry 4.0 environment*, vol. 513. 2017.
- [93] D. Arena, S. Perini, M. Taisch, and D. Kiritsis, "The Training Data Evaluation Tool: Towards a unified ontology-based solution for industrial training evaluation," *Procedia Manuf.*, vol. 23, pp. 219–224, Jan. 2018.
- [94] T. D. Zikos S, Rogotis S, Krinidis D, Ioannidis D, "Human-Resources Optimization & Re-Adaptation Modelling in Enterprises."
- [95] A. Tsolakis *et al.*, "Semantically enriched industry data & information modelling: A feasibility study on shop-floor incident recognition," in *IEEE International Conference on Industrial Informatics (INDIN)*, 2017.
- [96] D. Arena *et al.*, *Towards a semantically-enriched framework for human resource management*, vol. 513. 2017.
- [97] D. Arena *et al.*, "Human resource optimisation through semantically enriched data," *Int. J. Prod. Res.*, 2017.
- [98] A. deVos, M. S. IEEE Widergren, S. Member, J. Zhu, and M. Ieee, "XML FOR CIM MODEL EXCHANGE."
- [99] R. Volk, J. Stengel, and F. Schultmann, "Building Information Modeling (BIM) for existing buildings — Literature review and future needs," *Autom. Constr.*, vol. 38, pp. 109–127, Mar. 2014.
- [100] A. Perdikakis, "Towards Seamless Continuation of Knowledge in Product Lifecycle Management," 2016.
- [101] D. Arena *et al.*, "Human resource optimisation through semantically enriched data," *Int. J. Prod. Res.*, vol. 56, no. 8, pp. 2855–2877, 2017.
- [102] C. Ziogou *et al.*, *Decision Support based on a Semantically-Enriched Notification Platform at a Process Plant Floor*, vol. 40. 2017.
- [103] D. Arena, D. Kiritsis, C. Ziogou, and S. Voutetakis, "Semantics-driven knowledge representation for decision support and status awareness at process plant floors," in *2017 International Conference on Engineering, Technology and Innovation: Engineering, Technology and Innovation Management Beyond 2020: New Challenges, New Approaches, ICE/ITMC 2017 - Proceedings*, 2018, vol. 2018–Janua.
- [104] J. J. Carroll and G. Klyne, "Resource Description Framework (RDF): Concepts and Abstract Syntax," Feb. 2004.
- [105] A. Maedche and S. Staab, "Ontology Learning for the Semantic Web Ontologies for the Semantic Web."
- [106] E. Prud'hommeaux and A. Seaborne, "SPARQL Query Language for RDF."
- [107] J. Lafferty, A. McCallum, F. C. N. Pereira, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," pp. 282–289, 2001.
- [108] C. Sutton and A. McCallum, "An Introduction to Conditional Random Fields," *Mach. Learn.*, vol. 4, no. 4, pp. 267–373, 2011.

- [109] L. R. Rabiner and B. H. Juang, "An Introduction to Hidden Markov Models."
- [110] D. Arena and D. Kiritsis, *A methodological framework for ontology-driven instantiation of petri net manufacturing process models*, vol. 517. 2017.
- [111] D. N. Arena and D. Kiritsis, *A method towards modelling and analysis of semantically-enriched reconfigurable manufacturing systems*, vol. 488. 2016.
- [112] J. KEUN-CHAE and YEONG-DAE KIM, "Performance analysis of assembly/disassembly systems with unreliable machines and random processing times," *IIE Trans.*, vol. 30, pp. 41–53, 1998.
- [113] B. Smith and W. Ceusters, "Aboutness: Towards Foundations for the Information Artifact Ontology."
- [114] A. Ferscha and S. K. Tripathi, "Parallel and Distributed Simulation of Discrete Event Systems."
- [115] R. E. Nance and R. E., "A history of discrete event simulation programming languages," in *History of programming languages--II*, New York: ACM, 1996, pp. 369–427.
- [116] B. Safarinejadian, "Discrete Event Simulation and Petri net Modeling for Reliability Analysis," *Int. J. Soft Comput. Softw. Eng.*, vol. 2, no. 5, pp. 2251–7545, 2012.
- [117] G. Rubino and B. Tuffin, *An Introduction to Monte Carlo Methods and Rare Event Simulation Monte Carlo : the basics*, First. Wiley, 2009.
- [118] G. Balbo, "Introduction to Generalized Stochastic Petri Nets," in *Formal Methods for Performance Evaluation*, First., Springer, 2007, pp. 1–83.
- [119] J. Wang, "Petri Nets for Dynamic Event-Driven System Modeling," in *Handbook of Dynamic System Modeling*, no. 4, P. Fishwick, Ed. 2006, pp. 1–17.
- [120] E. G. Amparore, G. Balbo, M. Beccuti, S. Donatelli, and G. Franceschinis, "30 Years of GreatSPN," Springer, Cham, 2016, pp. 227–254.
- [121] E. G. Amparore, M. Beccuti, and S. Donatelli, "(Stochastic) Model Checking in GreatSPN," Springer, Cham, 2014, pp. 354–363.
- [122] E. A. Gilberto and S. Donatelli, "DSPN-Tool: A New DSPN and GSPN Solver for GreatSPN," in *2010 Seventh International Conference on the Quantitative Evaluation of Systems*, 2010, pp. 79–80.
- [123] M. Bertolini, "Reliability design of industrial plants using Petri nets," *ESCOLHA - J. Qual. Maint. Eng.*, vol. 3, no. 1997, pp. 29–39, 2009.
- [124] A. Petrone, L. Scatagliani, and F. Fabio, "Process Methodological relationship between RAM and QRA," in *International Petroleum Technology Conference*, 2009.
- [125] N. Trapani, M. Macchi, and L. Fumagalli, "Risk driven engineering of Prognostics and Health Management systems in manufacturing," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 995–1000, Jan. 2015.
- [126] D. Arena, F. Criscione, and N. Trapani, "Risk assessment in a chemical plant with a CPN-HAZOP Tool," *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 939–944, Jan. 2018.
- [127] D. N. Arena, D. Kiritsis, and N. Trapani, "A behaviour model for risk assessment of complex systems based on HAZOP and coloured Petri Nets," in *Advances in Production Management Systems: Innovative Production Management Towards Sustainable Growth*, 2015, no. Part I, pp. 573–581.
- [128] E. Zio, P. Baraldi, and F. Cadini, *Basics of reliability and risk analysis : worked out problems and solutions*. World Scientific, 2011.
- [129] M. Glossop MEng, A. Ioannides BEng, and J. Gould, "HSL/2005/58 REVIEW OF HAZARD IDENTIFICATION TECHNIQUES," 2005.
- [130] K. Jensen and G. Rozenberg, *High-level Petri Nets : Theory and Application*. Springer Berlin Heidelberg, 1991.
- [131] K. Jensen, *Coloured Petri Nets : Basic Concepts, Analysis Methods and Practical Use*. Springer Berlin Heidelberg, 1997.
- [132] E. Serral, J. De Smedt, M. Snoeck, and J. Vanthienen, "Context-adaptive Petri nets: Supporting adaptation for the execution context," *Expert Syst. Appl.*, vol. 42, no. 23, pp. 9307–9317, Dec. 2015.
- [133] S. A. McCoy *et al.*, "HAZID, a computer aid for hazard identification: 3. The fluid model and consequence evaluation systems," *Process Saf. Environ. Prot.*, vol. 77, no. 6, pp. 335–353, 1999.
- [134] F. I. Khan and S. A. Abbasi, "Mathematical model for HAZOP study time estimation," *J. Loss Prev. Process Ind.*, vol. 10, no. 4, pp. 249–257, Jul. 1997.

Annex A – List of worker's skills and worker groups

Table 26 List of worker's skills

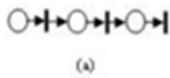
Set of Skills	
Skill 1.	Decision making
Skill 2.	Knowledge on manufacturing processes
Skill 3.	Management of human resources
Skill 4.	Administrative skill
Skill 5.	Management of shop floor
Skill 6.	Able to collaborate and coordinate actions
Skill 7.	Time Management skill
Skill 8.	Knowledge on pilot plant operations
Skill 9.	Able to use monitoring systems
Skill 10.	Management of the operations of pilot plants
Skill 11.	Automation, electrical and mechanical knowledge
Skill 12.	Maintenance of the shop floor facilities
Skill 13.	Able to respond fast and efficient to incidents
Skill 14.	Maintenance of the pilot plants
Skill 15.	Knowledge of mechanical design software
Skill 16.	Preserve information related to performed procedures
Skill 17.	Knowledge on automation software
Skill 18.	Able to use tools for construction of Pilot Plants
Skill 19.	Able to handle malfunctions
Skill 20.	Sharing information ability
Skill 21.	Knowledge and maintenance of IT infrastructures
Skill 22.	Knowledge on electrical systems
Skill 23.	Able to understand electrical schematics and manuals
Skill 24.	knowledge of electrical design software
Skill 25.	Able to recognize and report malfunction
Skill 26.	Knowledge of safety measures on Pilot Plants operation

Table 27 List of worker's skills

Worker Groups and related set of skills
A - Floor Manager - (1, 2, 3, 4, 5, 6)
B - Process Supervisor - (1, 2, 3, 4, 7, 8, 9, 10)
C - Maintenance Manager - (5, 6, 7, 11, 12, 20)
D - Maintenance Supervisor - (3, 6, 8, 11, 13, 20)
E - Process Operator - (6, 7, 9, 13, 18, 25,26)
F - Process Technician - (6, 13, 14, 17, 18, 19)
G - Electrical Technician - (6, 13, 14, 19,22, 23, 24)
H - Control Automation - (6, 7, 13, 14, 17, 19, 21, 22, 23, 24)
I - IT Technician - (6, 7, 13, 17, 19, 21)

Annex B1 – PN modelling patterns

1...14



(A)

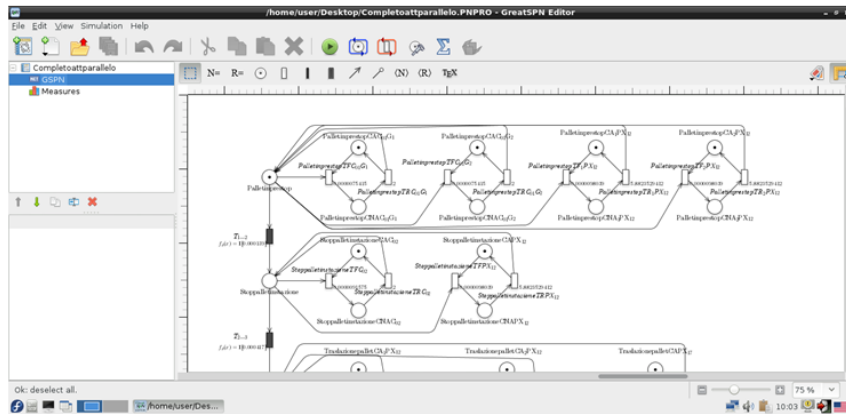
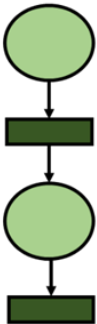
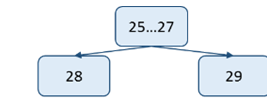


Figure 65 Proposed sequential pattern and representation in the model



AND – split (fork)

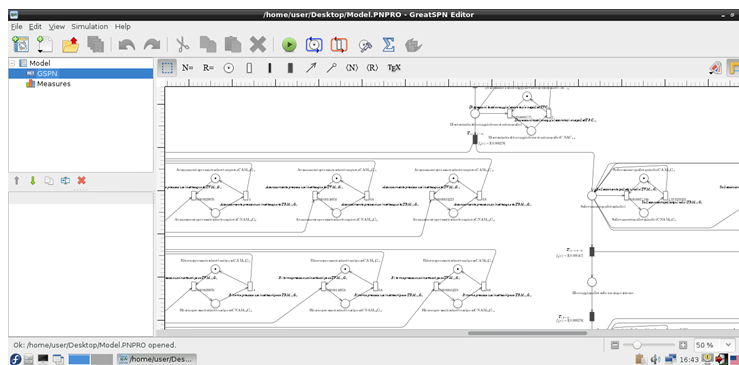
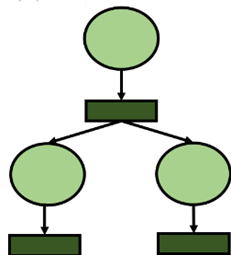


Figure 66 Proposed parallel pattern and representation in the model

19

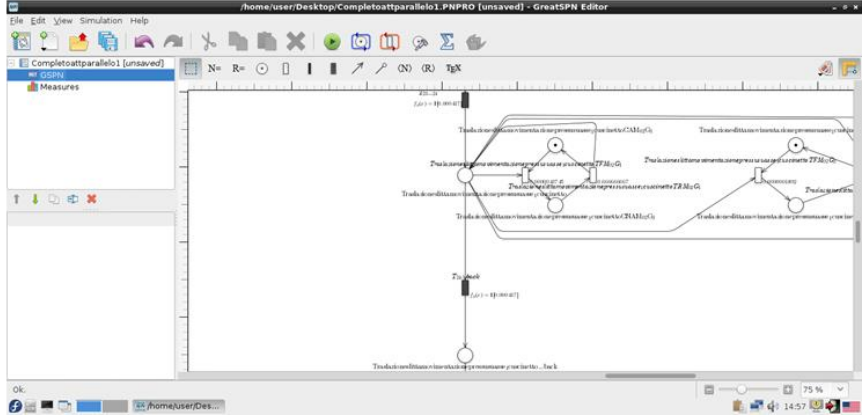
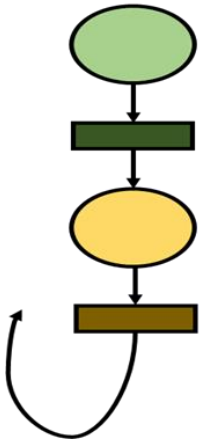


Figure 67 Proposed branch ultimate pattern and representation in the model

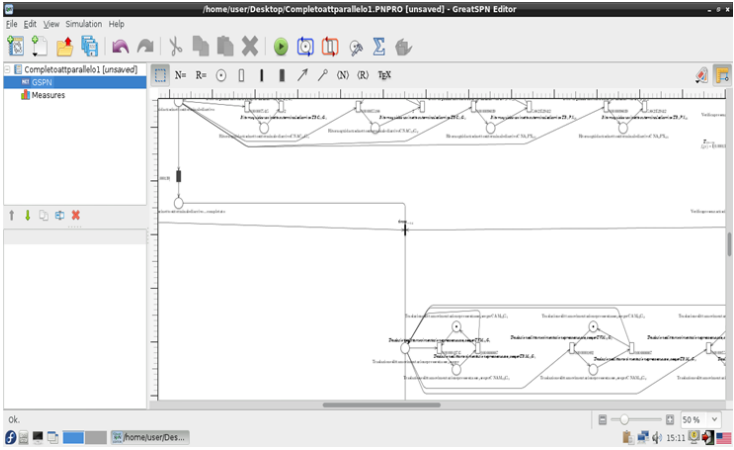
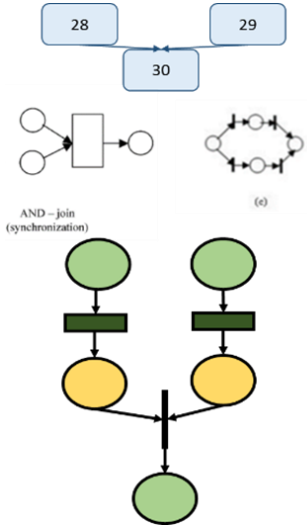


Figure 68 Proposed synchronisation pattern and representation in the model

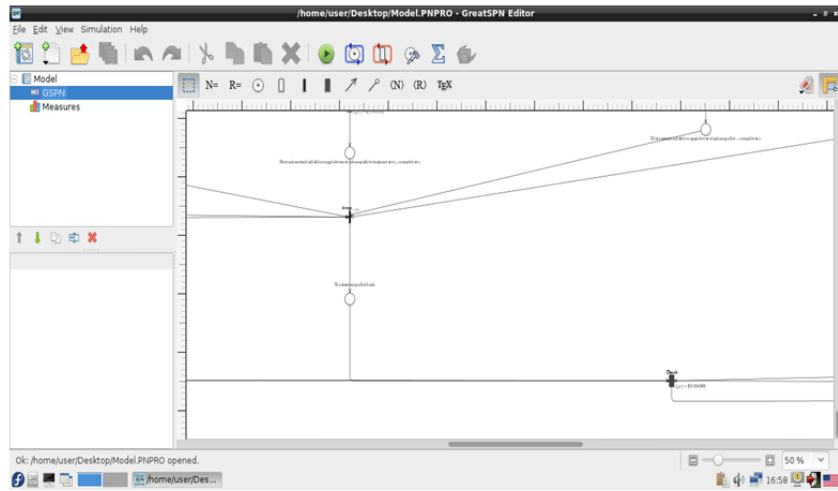
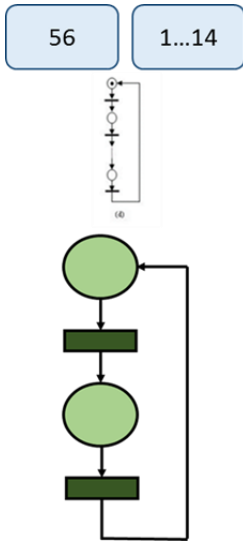


Figure 69 Proposed repetitive pattern and representation in the model

Annex B2 – SWRL-driven PN model creation

Step 1. Component Layer Creation – Module

$\text{Component}(?c) \wedge \text{failsDueTo}(?c, ?fce) \wedge \text{FailureCauseEvent}(?fce) \wedge \text{requiresComponentReplacing}(?fce, ?cre) \wedge \text{ComponentReplaceEvent}(?cre) \wedge \text{swrlx:makeOWLThing}(?m, ?fce) \rightarrow \text{Module}(?m)$
 $\wedge \text{isCharacterizedByFailureCauseEvent}(?m, ?fce) \wedge \text{isCharacterizedByComponentReplaceEvent}(?m, ?cre) \wedge \text{characterizesComponent}(?m, ?c)$

The screenshot shows the RAASO web interface with the SWRL Tab active. The interface is divided into several panels:

- Class hierarchy: Component**: A tree view showing the ontology structure. The **Component** class is selected, and its subclasses are visible: **CompBI**, **CompBR**, **CompFC**, **CompFF**, **CompFW**, **CompMI**, **CompRF**, and **CompUC**.
- Usage: CompBF**: A section showing the usage of the **CompBF** class. It lists 12 uses, including **CompBF Type Component**, **Individual: CompBF**, **CompBF isInvolvedIn AA9**, and **CompBF failsDueTo FailureOfBF**.
- Instances: CompBF**: A list of instances for the **CompBF** class, including **CompBI**, **CompBR**, **CompFC**, **CompFF**, **CompFW**, **CompMI**, **CompRF**, and **CompUC**.
- Description: CompBF**: A section showing the description of the **CompBF** class, including the **Component** type and the **isInvolvedIn AA9** property.
- Property assertions: CompBF**: A section showing property assertions for the **CompBF** class, including **failsDueTo FailureOfBF** and **isInvolvedIn AA9**.

Figure 70 SWRL Tab: Step 1. Components

RAASO (http://www.semanticweb.org/RAASO/) : [\\Sti\files.epfl.ch\sci-sti-dk\arena\Annual Reports\4th Year - Thesis\Application Cases\2-Reliability assessment of a Semi-Automated Assembly Station\InitialScenario_Case1_Test1.owl]

File Edit View Reasoner Tools Refactor Window Ontop Mastro Help

Active Ontology x Entities x Individuals by class x DL Query x SWRLTab x OntoGraf x

Class hierarchy: Module

Annotations Usage

Usage: e0103a5e_bc00_4fb6_af10_fbb03ffa022a

Show: this different

Found 16 uses of e0103a5e_bc00_4fb6_af10_fbb03ffa022a

- e0103a5e_bc00_4fb6_af10_fbb03ffa022a isComposedByWSCP 2c96e75e_1203_4284_afcd_ff80c49e6119
- e0103a5e_bc00_4fb6_af10_fbb03ffa022a isComposedByRCT 4a9f15f9_2103_4c09_87ed_ea8aa0c76e02
- e0103a5e_bc00_4fb6_af10_fbb03ffa022a isCharacterizedByComponentReplaceEvent ReplaceBF
- e0103a5e_bc00_4fb6_af10_fbb03ffa022a isComposedByFCT 857ae71b_9f50_41e8_a4c5_9e98bd7b2720
- e0103a5e_bc00_4fb6_af10_fbb03ffa022a isCharacterizedByFailureCauseEvent FailureOfBF
- e0103a5e_bc00_4fb6_af10_fbb03ffa022a characterizesComponent CompBF
- e0103a5e_bc00_4fb6_af10_fbb03ffa022a Type Module
- e0103a5e_bc00_4fb6_af10_fbb03ffa022a isComposedByFSCP 31e73a87_e1cd_4ac2_95af_3092d5dd4dc6

Instances: e0103a5e_bc00_4fb6_af10_fbb03ffa022a

Description: e0103a5e_bc00_4fb6_af10_fbb03ffa022a

Property assertions: e0103a5e_bc00_4fb6_af10_fbb03ffa022a

Types

- Module

Same Individual As

Different Individuals

Object property assertions

- isComposedByWSCP 2c96e75e_1203_4284_afcd_ff80c49e6119
- isComposedByRCT 4a9f15f9_2103_4c09_87ed_ea8aa0c76e02
- isCharacterizedByComponentReplaceEvent ReplaceBF
- isComposedByFCT 857ae71b_9f50_41e8_a4c5_9e98bd7b2720
- isCharacterizedByFailureCauseEvent FailureOfBF
- characterizesComponent CompBF
- isComposedByFSCP 31e73a87_e1cd_4ac2_95af_3092d5dd4dc6

Data property assertions

Negative object property assertions

Negative data property assertions

To use the reasoner click Reasoner > Start reasoner Show Inferences

Figure 71 SWRL Tab: Step 1. Modules from Components

Step 2. Component Layer Creation – Component State Places (Failure/Working)

Module(?m) ^ swrlx:makeOWLThing(?fscp, ?m) ^ swrlx:makeOWLThing(?wscp, ?m) -> FailureStateComponentPlace(?fscp) ^ WorkingStateComponentPlace(?wscp) ^ isComposedByFSCP(?m, ?fscp) ^ isComposedByWSCP(?m, ?wscp)

The screenshot displays the RAASO web interface with the following components:

- Class Hierarchy (Left):** Shows a tree structure starting with `owl:Thing`. Under `Place`, `FailureStateComponentPlace` is selected.
- Usage: SOS2 (Top Right):** Shows 26 uses of the `SOS2` property. Examples include:
 - `AA1` hasEndState `SOS2`
 - `AA2` hasStartState `SOS2`
 - `bd8dfe10_607a_45c6_be1b_828b35e44181` isCharacterizedBySystemOperationalState `SOS2`
 - `SOS2` isInitialOperationalStateOf `AA2`
 - `SOS2` isTerminalOperationalStateOf `AA1`
 - `SOS2` Type `SystemOperationalState`
 - `SOS2` Type `SystemState`
 - `SOS2` characterizesSystemPlace `bd8dfe10_607a_45c6_be1b_828b35e44181`
- Instances (Bottom Left):** Lists 10 instances of `FailureStateComponentPlace` with their URIs.
- Description: SOS2 (Bottom Middle):** Lists types associated with `SOS2`:
 - `InitialOperationalState`
 - `SystemOperationalState`
 - `SystemState`
 - `TerminalOperationalState`
- Property assertions: SOS2 (Bottom Right):** Shows object and data property assertions for `SOS2`, including:
 - `isInitialOperationalStateOf AA2`
 - `isTerminalOperationalStateOf AA1`
 - `characterizesSystemPlace bd8dfe10_607a_45c6_be1b_828b35e44181`
 - `characterizesPNElement bd8dfe10_607a_45c6_be1b_828b35e44181`

Figure 72 SWRL Tab: Step 2. Failure State Component Place

The screenshot displays the RAASO web interface with the following components:

- Class Hierarchy (Left):** A tree view showing the ontology structure. The class `WorkingStateComponentPlace` is selected and highlighted in blue.
- Usage: SOS2 (Top Right):** A pane showing the usage of the `SOS2` property. It lists 26 uses, including:
 - `AA1` hasEndState `SOS2`
 - `AA2` hasStartState `SOS2`
 - `bd8dfe10_607a_45c6_be1b_828b35e44181` isCharacterizedBySystemOperationalState `SOS2`
 - `bd8dfe10_607a_45c6_be1b_828b35e44181` isCharacterizedBySystemElement `SOS2`
 - `SOS2` isInitialOperationalStateOf `AA2`
 - `SOS2` isTerminalOperationalStateOf `AA1`
 - Individual: `SOS2`
 - `SOS2` Type `SystemOperationalState`
 - `SOS2` Type `SystemState`
 - `SOS2` characterizesSystemPlace `bd8dfe10_607a_45c6_be1b_828b35e44181`
- Instances (Bottom Left):** A list of instances for the class `WorkingStateComponentPlace`, including URIs like `2b8b1670_1ccc_4c7f_9346_8e12e9f5d68c`.
- Description: SOS2 (Bottom Middle):** A pane showing the types associated with `SOS2`:
 - `InitialOperationalState`
 - `SystemOperationalState`
 - `SystemState`
 - `TerminalOperationalState`
- Property assertions: SOS2 (Bottom Right):** A pane showing object and data property assertions for `SOS2`, such as:
 - `isInitialOperationalStateOf AA2`
 - `isTerminalOperationalStateOf AA1`
 - `characterizesSystemPlace bd8dfe10_607a_45c6_be1b_828b35e44181`
 - `characterizesPNElement bd8dfe10_607a_45c6_be1b_828b35e44181`

Figure 73 SWRL Tab: Step 2. Working State Component Place

Step 3. Component Layer Creation - Failure Component Transition (lambda conversion)

Module(?m) ^ isCharacterizedByFailureCauseEvent (?m, ?fce) ^ swrlm:eval(?l, "1.0/mtbf", ?mtbf) ^ hasMTBF(?fce, ?mtbf) ^ FailureCauseEvent(?fce) ^ swrlx:makeOWLThing(?fct, ?m) -> FailureComponentTransition(?fct) ^ hasLambda(?fct, ?l) ^ isComposedByFCT(?m, ?fct) ^ isCharacterizedByFailureCauseEvent (?fct, ?fce)

The screenshot shows the RAASO web interface with the following components:

- Class hierarchy:** A tree view on the left showing classes like Arc, AssemblyActivityExecutionTime, AssemblyStation, etc., with FailureComponentTransition selected.
- Instances:** A list of 14 instances for FailureComponentTransition, with the first one (02a9fa16_2c5b_4d47_b71f_6686dec81ee0) selected.
- Description:** A text area showing the SWRL rule: `Module(?m) ^ isCharacterizedByFailureCauseEvent (?m, ?fce) ^ swrlm:eval(?l, "1.0/mtbf", ?mtbf) ^ hasMTBF(?fce, ?mtbf) ^ FailureCauseEvent(?fce) ^ swrlx:makeOWLThing(?fct, ?m) -> FailureComponentTransition(?fct) ^ hasLambda(?fct, ?l) ^ isComposedByFCT(?m, ?fct) ^ isCharacterizedByFailureCauseEvent (?fct, ?fce)`
- Property assertions:** A list of assertions for the selected instance, including `isCharacterizedByFailureCauseEvent FailureOFF` and `hasLambda "0.0018999999981"^^xsd:double`.

Figure 74 SWRL Tab: Step 3. Failure Component Transition

Step 4. Component Layer Creation - Repair Component Transition (mu conversion)

Module(?m) ^ isCharacterizedByComponentReplaceEvent (?m, ?cre) ^ swrlm:eval(?mu, "1.0/mttr", ?mttr) ^ hasMTTR(?cre, ?mttr) ^ ComponentReplaceEvent (?cre) ^ swrlx:makeOWLThing(?rct, ?m) -> ReplaceComponentTransition(?rct) ^ hasMu(?rct, ?mu) ^ isComposedByRCT(?m, ?rct) ^ isCharacterizedByComponentReplaceEvent (?rct, ?cre)

The screenshot shows the RAASO web interface with the following components:

- Class Hierarchy:** A tree view on the left showing classes like Arc, AssemblyActivityExecutionTime, AssemblyStation, etc. 'ReplaceComponentTransition' is highlighted.
- Instances:** A list of instance URIs for 'ReplaceComponentTransition' at the bottom left, with the first one selected.
- Description:** A central panel showing the selected instance's types ('ReplaceComponentTransition', 'Transition') and a data property assertion 'hasMu' with the value '0.057219989688957855'^xsd:double.
- Property Assertions:** A right panel showing object property assertions ('isCharacterizedByComponentReplaceEvent ReplaceFF') and data property assertions ('hasMu').

Figure 75 SWRL Tab: Step 4. Repair Component Transition

Step 5. Component Layer Creation - Component PT Arc from FSCP to RCT

Module(?m) ^ isComposedByRCT(?m, ?rct) ^ ReplaceComponentTransition(?rct) ^ isComposedByFSCP(?m, ?fscp) ^ FailureStateComponentPlace(?fscp)^swrlx:makeOWLThing(?pta, ?m) -> PTComponentArc(?pta) ^ PTArcHasFSCP(?pta, ?fscp) ^ PTArcHasRCT(?pta, ?rct)

The screenshot shows the RAASO web interface with the following components:

- Class hierarchy (PTComponentArc):**
 - Arc
 - PTComponentArc
 - PTSystemArc
 - SystemToComponentArc
 - TPComponentArc
 - TPSystemArc
 - AssemblyActivityExecutionTime
 - AssemblyStation
 - AssemblyStationPlannedEvent
 - AssemblyStationUnPlannedEvent
 - Component
 - ComponentFailureRate
 - ComponentFunction
 - ComponentReplaceRate
 - ComponentState
 - Module
 - Place
 - FailureStateComponentPlace
 - SystemPlace
- Instances (458e529a_b2e4_4c6f_8960_6be7e9186710):**
 - 13fb3aaa_f631_4fcb_9515_7fda0c5e04c4
 - 1639e4c0_8b83_46ef_b261_8a69d5c5ebdd
 - 27458680_b0a5_4d95_aa0f_01acee73e107
 - 458e529a_b2e4_4c6f_8960_6be7e9186710**
 - 46b34af5_ad22_4da9_8c15_273de81ee19e
 - 56c05c68_5ff5_46b7_be8b_e1ba253a3f1f
 - 57283ed3_186a_42f3_95d0_20bd3bfab100
 - 599e57fe_0c4e_41fb_a159_c89db12162b5
 - 5cca752b_f192_4122_a156_60d59640b426
 - 77caab52_c1aa_469e_aa9f_4038573eb40d
 - 7a0c0680_bc6c_46e2_9cc7_2c3b7e3d6230
 - 7c371b1a_4b78_4e48_901f_8c93c392b815
 - 849d55e8_f941_4a58_9438_334dea5838e2
 - 918091fe_a5e3_4f53_be84_2b04ab09a4d8
 - 94a476e2_b23d_4805_b55d_81ae01c4808f
 - d7ed2298_947d_4c33_ad29_935a1a0ec2ef
 - df460106_72a1_49ce_a2c8_8c22644dcf15
 - e629bb8e_83e8_4da2_ad83_6c79ba7792fb
- SWRL Rule Uses (458e529a_b2e4_4c6f_8960_6be7e9186710):**
 - 458e529a_b2e4_4c6f_8960_6be7e9186710 Type Arc
 - 458e529a_b2e4_4c6f_8960_6be7e9186710 Type PTComponentArc
 - 458e529a_b2e4_4c6f_8960_6be7e9186710 PTArcHasFSCP 56e912a2_5a3b_492a_9320_b9936aa07ad8
 - 458e529a_b2e4_4c6f_8960_6be7e9186710 PTArcHasRCT 14b66d34_efcf_42b3_b0e5_239430e5e0c0
- Property Assertions (458e529a_b2e4_4c6f_8960_6be7e9186710):**
 - PTArcHasFSCP 56e912a2_5a3b_492a_9320_b9936aa07ad8
 - PTArcHasRCT 14b66d34_efcf_42b3_b0e5_239430e5e0c0

Figure 76 SWRL Tab: Step 5. Component PT Arc from FSCP to RCT

Step 6. Component Layer Creation - Component PT Arc from WSCP to FCT

Module(?m) ^ isComposedByFCT(?m, ?fct) ^ FailureComponentTransition(?fct) ^ isComposedByWSCP(?m, ?wscp) ^ WorkingStateComponentPlace(?wscp) ^ swrlx:makeOWLThing(?pta, ?m) -> PTComponentArc(?pta) ^ PTArcHasWSCP(?pta, ?wscp) ^ PTArcHasFCT(?pta, ?fct)

The screenshot shows the RAASO web interface with the following components:

- Class hierarchy (PTComponentArc):**
 - Arc
 - ComponentToSystemArc
 - PTComponentArc**
 - PTSystemArc
 - SystemToComponentArc
 - TPComponentArc
 - TPSystemArc
 - AssemblyActivityExecutionTime
 - AssemblyStation
 - AssemblyStationPlannedEvent
 - AssemblyStationUnPlannedEvent
 - Component
 - ComponentFailureRate
 - ComponentFunction
 - ComponentReplaceRate
 - ComponentState
 - Module
 - Place
 - FailureStateComponentPlace
 - SystemPlace

- Instances (13fb3aaa_f631_4fcb_9515_7fda0c5e04c4):**
- 13fb3aaa_f631_4fcb_9515_7fda0c5e04c4
- 1639e4c0_8b83_46ef_b261_8a69d5c5ebdd
- 27458680_b0a5_4d95_aa0f_01acee73e107
- 458e529a_b2e4_4c6f_8960_6be7e9186710
- 46b34af5_ad22_4da9_8c15_273de81ee19e
- 56c05c68_5ff5_46b7_be8b_e1ba253a3f1f
- 57283ed3_186a_42f3_95d0_20bd3bfab100
- 599e57fe_0c4e_41fb_a159_c89db12162b5
- 5cca752b_f192_4122_a156_60d59640b426
- 77caab52_c1aa_469e_aa9f_4038573eb40d
- 7a0c0680_bc6c_46e2_9cc7_2c3b7e3d6230
- 7c371b1a_4b78_4e48_901f_8c93c392b815
- 849d55e8_f941_4a58_9438_334dea5838e2
- 918091fe_a5e3_4f53_be84_2b04ab09a4d8
- 94a476e2_b23d_4805_b55d_81ae01c4808f
- d7ed2298_947d_4c33_ad29_935a1a0ec2ef
- df460106_72a1_49ce_a2c8_8c22644dcf15
- e629bb8e_83e8_4da2_ad83_6c79ba7792fb
- Description (13fb3aaa_f631_4fcb_9515_7fda0c5e04c4):**
- Types: Arc, PTComponentArc
- Property assertions (13fb3aaa_f631_4fcb_9515_7fda0c5e04c4):**
- Object property assertions:
 - PTArcHasWSCP 3a587cc4_9d9c_4232_8406_384d67441f1d
 - PTArcHasFCT ce834287_a21e_4eb4_9812_cb8299095774
- Data property assertions: +
- Negative object property assertions: +
- Negative data property assertions: +

Figure 77 SWRL Tab: Step 6. Component PT Arc from WSCP to FCT

Step 7. Component Layer Creation - Component TP Arc from FCT to FSCP

Module(?m) ^ isComposedByFCT(?m, ?fct) ^ FailureComponentTransition(?fct) ^ isComposedByFSCP(?m, ?fscp) ^ FailureStateComponentPlace(?fscp) ^ swrlx:makeOWLThing(?tpa, ?m) -> TPComponentArc(?tpa) ^ TPArcHasFSCP(?tpa, ?fscp) ^ TPArcHasFCT(?tpa, ?fct)

The screenshot shows the RAASO web interface with the following components:

- Class hierarchy:** TPComponentArc
 - Arc
 - ComponentToSystemArc
 - PTComponentArc
 - PTSystemArc
 - SystemToComponentArc
 - TPComponentArc (selected)
 - TPSystemArc
 - AssemblyActivityExecutionTime
 - AssemblyStation
 - AssemblyStationPlannedEvent
 - AssemblyStationUnPlannedEvent
 - Component
 - ComponentFailureRate
 - ComponentFunction
 - ComponentReplaceRate
 - ComponentState
 - Module
 - Place
 - FailureStateComponentPlace
 - SystemPlace
- Instances:** 0fde0404_2db5_4e67_8545_1a6eb618332a
 - 0fde0404_2db5_4e67_8545_1a6eb618332a (selected)
 - 355c02dd_f1fa_49f4_8920_e3e04381bdd8
 - 363dff14_3ad6_43e0_a810_4e62e61dae6b
 - 3e109991_e3a0_4f17_a71d_3882536a44eb
 - 3e8094c6_c7d2_4458_8d50_e6d9196fbd3b
 - 45943456_bb9f_43bb_94d5_3b025cd50ee2
 - 59b87183_5042_44ec_a9a6_d4b820d1f1d6
 - 5fb4ff91_efb3_43a4_81ec_d8c854aa1f16
 - 75d8dcd0_98a4_4cb0_8466_d9a6112f98cf
 - 836b3309_d087_41f2_853f_b881ba2d1800
 - 9213cf8_1862_4652_ae5f_953fb50ad448
 - 95e285d8_9fe9_465e_8ef2_1403a476d4ca
 - b0e26b62_5911_4750_9a3c_d2ded4ec897c
 - b3771efe_1ee8_4342_a8e1_5dbccea9b363
 - b44d1b5c_f9b1_4589_9dd2_9998cf8659df
 - bd96623e_7519_4cd1_a664_101e905da682
 - c4dee1c0_65f0_4531_a1cd_2c3e76f3ee1f
 - f86e12fa_c4c9_478a_9037_8cad3e2cd99a
- Description:** 0fde0404_2db5_4e67_8545_1a6eb618332a
 - Types: Arc, TPComponentArc
 - Property assertions: TPArcHasFCT 02a9fa16_2c5b_4d47_b71f_6686dec81ee0, TPArcHasFSCP 56e912a2_5a3b_492a_9320_b9936aa07ad8

Figure 78 SWRL Tab: Step 7. Component TP Arc from FCT to FSCP

Step 8. Component Layer Creation - Component TP Arc from RCT to WSCP

Module(?m) ^ isComposedByRCT(?m, ?rct) ^ ReplaceComponentTransition(?rct) ^ isComposedByWSCP(?m, ?wscp) ^ WorkingStateComponentPlace(?wscp) ^ swrlx:makeOWLThing(?tpa, ?m) -> TPComponentArc(?tpa) ^ TPArcHasWSCP(?tpa, ?wscp) ^ TPArcHasRCT(?tpa, ?rct)

The screenshot shows the RAASO Semantic Web Editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, Ontop, Mastro, and Help. The browser address bar shows the RAASO URL. The main workspace is divided into several panes:

- Class hierarchy:** Shows a tree of classes under 'TPComponentArc', including Arc, ComponentToSystemArc, PTComponentArc, PTSystemArc, SystemToComponentArc, TPComponentArc, TPSystemArc, AssemblyActivityExecutionTime, AssemblyStation, AssemblyStationPlannedEvent, AssemblyStationUnPlannedEvent, Component, ComponentFailureRate, ComponentFunction, ComponentReplaceRate, ComponentState, Module, Place, FailureStateComponentPlace, and SystemPlace.
- SWRL Tab:** Displays the SWRL query: `Module(?m) ^ isComposedByRCT(?m, ?rct) ^ ReplaceComponentTransition(?rct) ^ isComposedByWSCP(?m, ?wscp) ^ WorkingStateComponentPlace(?wscp) ^ swrlx:makeOWLThing(?tpa, ?m) -> TPComponentArc(?tpa) ^ TPArcHasWSCP(?tpa, ?wscp) ^ TPArcHasRCT(?tpa, ?rct)`
- Results:** Shows the results of the query for the instance `355c02dd_f1fa_49f4_8920_e3e04381bdd8`. It lists 8 uses of this instance, including `TPArcHasWSCP 37446ce9_82e3_4686_9b8d_1960d8179437`, `Type Arc`, `Type TPComponentArc`, and `TPArcHasRCT 14b66d34_efcf_42b3_b0e5_239430e5e0c0`.
- Property assertions:** Shows object property assertions for `TPArcHasWSCP 37446ce9_82e3_4686_9b8d_1960d8179437` and `TPArcHasRCT 14b66d34_efcf_42b3_b0e5_239430e5e0c0`.

Figure 79 SWRL Tab: Step 8. Component TP Arc from RCT to WSCP

Step 9. System Layer Creation – System Transitions from Assembly Activities

AssemblyActivity(?a) ^ hasActivityExecutionTime (?a, ?aet) ^ swrlx:makeOWLThing(?st, ?a) -> SystemTransition(?st) ^ hasFiringRate (?st, ?aet) ^ characterizesSystemTransition(?a, ?st)

The screenshot displays the RAASO Semantic Web Editor interface. The top navigation bar includes 'File', 'Edit', 'View', 'Reasoner', 'Tools', 'Refactor', 'Window', 'Ontop', 'Mastro', and 'Help'. The main workspace is divided into several panels:

- Class Hierarchy:** Shows a tree view of classes under 'SystemTransition', including 'TPSystemArc', 'AssemblyActivityExecutionTime', 'AssemblyStation', 'AssemblyStationPlannedEvent', 'AssemblyStationUnPlannedEvent', 'Component', 'ComponentFailureRate', 'ComponentFunction', 'ComponentReplaceRate', 'ComponentState', 'Module', 'Place', 'FailureStateComponentPlace', 'SystemPlace', 'WorkingStateComponentPlace', 'SystemState', and 'Transition'.
- Instances:** Lists instances of 'SystemTransition' with URIs such as '3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588', '5b5d3913_0d01_4218_8d0b_e51a1c257d32', '7cd0ce80_80c8_41cb_8143_ab6c07cba544', '7d4f1979_76bc_4d8b_a343_f7fcc1912c0b', '9620ea61_79d5_4175_8e1b_ba555be74ca5', '9e6c1d5c_1d47_48a6_9f6c_1491678f12d7', 'a2480f57_499d_4db5_8f67_add59a3210b4', 'b8ee4964_69c4_4ecb_9d45_85c8658cebe1', and 'd7691086_a133_4a0d_8337_c00e5679054f'.
- Usage:** Shows 26 uses of the URI '3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588', including 'SystemTransitionHasDownstreamPlace', 'Type SystemTransition', 'isCharacterizedByAssemblyActivity AA5', 'SystemTransitionHasPTarc', 'SystemTransitionHasTParc', 'hasFiringRate 1.22449', 'SystemTransitionHasUpstreamPlace', and 'isCharacterizedBySystemElement AA5'.
- Property Assertions:** Lists object and data property assertions for the selected instance, such as 'SystemTransitionHasDownstreamPlace', 'SystemTransitionHasPTarc', 'SystemTransitionHasTParc', 'isCharacterizedByAssemblyActivity AA5', 'isCharacterizedBySystemElement AA5', 'SystemTransitionHasUpstreamPlace', and 'hasFiringRate 1.22449'.

Figure 80 Step 9. System Transitions from Assembly Activities

Step 10. System Layer Creation – System Place from System States

SystemOperationalState(?ss) ^ swrlx:makeOWLThing(?sp, ?ss) -> characterizesSystemPlace(?ss, ?sp) ^ SystemPlace(?sp)

The screenshot shows the RAASO web interface with the following components:

- Class hierarchy:** SystemOperationalState
 - AssemblyStationUnPlannedEvent
 - Component
 - ComponentFailureRate
 - ComponentFunction
 - ComponentReplaceRate
 - ComponentState
 - Module
 - Place
 - FailureStateComponentPlace
 - SystemPlace
 - WorkingStateComponentPlace
 - SystemState
 - InitialOperationalState
 - SystemOperationalState (selected)
 - SystemFailureState
 - TerminalOperationalState
 - Transition
 - FailureComponentTransition
 - ReplaceComponentTransition
 - SystemTransition
- Usage: SOS1:** Found 26 uses of SOS1
 - 68b7feda_cc82_418f_a83a_34319b280408
 - 68b7feda_cc82_418f_a83a_34319b280408 isCharacterizedBySystemOperationalState SOS1
 - 68b7feda_cc82_418f_a83a_34319b280408 isCharacterizedBySystemElement SOS1
 - AA1
 - AA1 hasStartState SOS1
 - AA9
 - AA9 hasEndState SOS1
 - SOS1
 - SOS1 isInitialOperationalStateOf AA1
 - Individual: SOS1
 - SOS1 characterizesSystemPlace 68b7feda_cc82_418f_a83a_34319b280408
 - SOS1 Type SystemState
 - SOS1 characterizesPNElement 68b7feda_cc82_418f_a83a_34319b280408
 - SOS1 Type SystemOperationalState
- Instances: SOS1:** SOS1, SOS2, SOS3, SOS4, SOS5, SOS6, SOS7, SOS8, SOS9
- Description: SOS1:**
 - Types: InitialOperationalState, SystemOperationalState, SystemState, TerminalOperationalState
- Property assertions: SOS1:**
 - isInitialOperationalStateOf AA1
 - characterizesSystemPlace 68b7feda_cc82_418f_a83a_34319b280408
 - characterizesPNElement 68b7feda_cc82_418f_a83a_34319b280408
 - isTerminalOperationalStateOf AA9

Figure 81 SWRL Tab: Step 10. System operational States

The screenshot displays the RAASO Semantic Web Editor interface. The main window title is "RAASO (http://www.semanticweb.org/RAASO/) : [\\Sb1files.epfl.ch\sci-sti-dk\arena\Annual Reports\4th Year - Thesis\Application Cases\2-Reliability assessment of a Semi-Automated Assembly Station\InitialScenario_Case1_Test1.owl]". The interface is divided into several panes:

- Class Hierarchy (Left):** Shows a tree structure of classes under "SystemPlace", including "TPSystemArc", "AssemblyActivityExecutionTime", "AssemblyStation", "Component", "Module", "Place", "SystemPlace", "SystemState", and "Transition".
- SWRL Query Editor (Center):** Contains the query: `Usage: 11a1c667_1e08_4aa0_bf9b_efbbbee9429d`. Below it, a list of 24 uses is shown, including:
 - `11a1c667_1e08_4aa0_bf9b_efbbbee9429d Type SystemPlace`
 - `11a1c667_1e08_4aa0_bf9b_efbbbee9429d SystemPlaceHasPTArc 7b432239_1bf7_4e1a_89ef_7a596b5253fc`
 - `11a1c667_1e08_4aa0_bf9b_efbbbee9429d SystemPlaceHasTPArc 66106280_fbd2_4fae_b8eb_3829649e7c18`
 - `11a1c667_1e08_4aa0_bf9b_efbbbee9429d SystemPlaceHasDownstreamTransition 9620ea61_79d5_4175_8e1b_ba555be74ca5`
 - `11a1c667_1e08_4aa0_bf9b_efbbbee9429d Type Place`
 - `11a1c667_1e08_4aa0_bf9b_efbbbee9429d SystemPlaceHasUpstreamTransition a2480f57_499d_4db5_8f67_add59a3210b4`
 - `11a1c667_1e08_4aa0_bf9b_efbbbee9429d isCharacterizedBySystemOperationalState SOS3`
 - `11a1c667_1e08_4aa0_bf9b_efbbbee9429d isCharacterizedBySystemElement SOS3`
- Instances (Bottom Left):** Lists instances of "SystemPlace", with the first one selected: `11a1c667_1e08_4aa0_bf9b_efbbbee9429d`. Other instances include `498f5768_73fc_43d6_9629_07f4e5b1b0ab`, `68b7feda_cc82_418f_a83a_34319b280408`, etc.
- Property Assertions (Bottom Right):** Shows a list of object property assertions for the selected instance:
 - `SystemPlaceHasPTArc 7b432239_1bf7_4e1a_89ef_7a596b5253fc`
 - `SystemPlaceHasTPArc 66106280_fbd2_4fae_b8eb_3829649e7c18`
 - `SystemPlaceHasDownstreamTransition 9620ea61_79d5_4175_8e1b_ba555be74ca5`
 - `SystemPlaceHasUpstreamTransition a2480f57_499d_4db5_8f67_add59a3210b4`
 - `isCharacterizedBySystemOperationalState SOS3`
 - `isCharacterizedBySystemElement SOS3`

Figure 82 SWRL Tab: Step 10. System Place from System States

Step 11. System Layer Characterization- Downstream System State

characterizesSystemPlace(?ss, ?sp) ^ SystemPlace(?sp) ^ SystemTransition(?st) ^ AssemblyActivity(?a) ^ characterizesSystemTransition (?a, ?st) ^ SystemOperationalState(?ss) ^ hasEndState(?a, ?ss) -> SystemTransitionHasDownstreamPlace(?st, ?sp)

Step 12. System Layer Characterization- Upstream System State

characterizesSystemPlace (?ss, ?sp) ^ SystemPlace(?sp) ^ SystemTransition(?st) ^ AssemblyActivity(?a) ^ characterizesSystemTransition (?a, ?st) ^ SystemOperationalState(?ss) ^ hasStartState(?a, ?ss) -> SystemTransitionHasUpstreamPlace(?st, ?sp)

The screenshot shows the RAASO Semantic Web Reasoner interface. The main window displays a SWRL query in the central pane:

```

    Usage: 3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588
    Found 26 uses of 3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588
    3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 SystemTransitionHasDownstreamPlace 93f64b91_1cb8_4f32_af81_443f3f9e23c6
    3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 Type SystemTransition
    3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 isCharacterizedByAssemblyActivity AA5
    3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 SystemTransitionHasPTarc 98c6c410_50de_48c7_a009_9775a39b8879
    3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 SystemTransitionHasTParc c8cbdc12_05b0_4e30_9261_40dfa77f563d
    3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 hasFiringRate 1.22449
    3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 SystemTransitionHasUpstreamPlace 498f5768_73fc_43d6_9629_07f4e5b1b0ab
    3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 isCharacterizedBySystemElement AA5
    3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 Type Transition
    498f5768_73fc_43d6_9629_07f4e5b1b0ab SystemPlaceHasDownstreamTransition 3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588
    93f64b91_1cb8_4f32_af81_443f3f9e23c6 SystemPlaceHasUpstreamTransition 3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588
  
```

The right-hand pane shows the query's results and property assertions:

- Types:** SystemTransition, Transition
- Object property assertions:**
 - SystemTransitionHasDownstreamPlace
 - SystemTransitionHasPTarc 98c6c410_50de_48c7_a009_9775a39b8879
 - isCharacterizedByAssemblyActivity AA5
 - SystemTransitionHasTParc c8cbdc12_05b0_4e30_9261_40dfa77f563d
 - isCharacterizedBySystemElement AA5
 - SystemTransitionHasUpstreamPlace 498f5768_73fc_43d6_9629_07f4e5b1b0ab
- Data property assertions:**
 - hasFiringRate 1.22449

The bottom of the interface shows a list of instances for the class SystemTransition, including IRIs like 3b22d404_bff6_4dc4_8c6c_4bdd7e2c8588 and 5b5d3913_0d01_4218_8d0b_e51a1c257d32.

Figure 83 SWRL Tab: Steps 11 & 12. System Transitions

Step 13. System Layer Characterization- PT Arc

SystemPlace(?sp) ^ swrlx:makeOWLThing(?pta, ?st) ^ SystemTransition(?st) ^ SystemPlaceHasDownstreamTransition(?sp, ?st) -> SystemTransitionHasPTArc(?st, ?pta) ^ PTSystemArc(?pta) ^ SystemPlaceHasPTArc(?sp, ?pta)

Step 14. System Layer Characterization-TP Arc

swrlx:makeOWLThing(?tpa, ?sp) ^ SystemTransitionHasDownstreamPlace(?st, ?sp) ^ SystemPlace(?sp) ^ SystemTransition(?st) -> SystemTransitionHasTPArc(?st, ?tpa) ^ SystemPlaceHasTPArc(?sp, ?tpa) ^ TPSystemArc(?tpa)

The screenshot shows the RAASO Semantic Web Editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, Ontop, Mastro, and Help. The address bar shows the URL for RAASO. The main workspace is divided into several panes:

- Class hierarchy:** A tree view on the left showing the ontology structure. The class **SystemPlace** is selected.
- Annotations:** A pane showing the SWRL query: `swrlx:makeOWLThing(?pta, ?st) ^ SystemTransition(?st) ^ SystemPlaceHasDownstreamTransition(?sp, ?st) -> SystemTransitionHasPTArc(?st, ?pta) ^ PTSystemArc(?pta) ^ SystemPlaceHasPTArc(?sp, ?pta)`.
- Instances:** A list of instances for the selected class **SystemPlace**, including `0c56a995_7840_444d_8184_829f5c0d6944`.
- Property assertions:** A pane on the right showing the results of the query. It lists various property assertions for the selected instance, such as `isCharacterizedBySystemOperationalState SOS6`, `SystemPlaceHasTPArc 643bd9fa_1768_418c_9f30_ef32f1e14094`, and `SystemPlaceHasDownstreamTransition d9fc288b_af56_43b2_9098_537228d12d97`.

Figure 84 SWRL Tab: Step 13 & 14. SystemPlace

Step 15. Cross-Layers link creation - Component To System Arc

Module(?m) ^ characterizesComponent(?m, ?c) ^ Component(?c) ^isInvolvedIn(?c,?a) ^ AssemblyActivity(?a) ^ characterizesSystemTransition (?a, ?st) ^ SystemTransition(?st) ^ SystemTransitionHasUpstreamPlace(?st, ?sp) ^ SystemPlace(?sp) ^ swrlx:makeOWLThing(?tpa, ?m) ^isComposedByRCT(?m, ?rct) -> ComponentToSystemArc(?tpa) ^ CSArcHasSystemPlace(?tpa, ?sp) ^ CSArcHasRCT(?tpa, ?rct)

The screenshot displays the RAASO web interface with the following components:

- Class Hierarchy (Left):** Shows a tree structure starting from `owl:Thing`, with `Arc` expanded to show subclasses like `ComponentToSystemArc`, `PTComponentArc`, `PTSystemArc`, `SystemToComponentArc`, `TPComponentArc`, and `TPSystemArc`.
- Usage Information (Top Right):** Shows the URI `5fb5602a_f15f_4f8d_a42d_b9cc42c85864` and indicates it is used 8 times.
- Instances (Bottom Left):** Lists several instance URIs for `ComponentToSystemArc`, including `64752e0e_7807_4f8e_b950_3311e106f8d6` and `718ad2b3_5035_4648_9128_f1c8408debd7`.
- Property Assertions (Bottom Right):** Shows two assertions: `CSArcHasSystemPlace 0e7a93ea_05b7_4318_8840_450390613c90` and `CSArcHasRCT 760a8239_cb3d_4b3e_ad97_598949fbf582`.

Figure 85 SWRL Tab: Step 15. Component To System Arc

Step 16. Cross-Layers link creation -

Module(?m) ^ characterizesComponent(?m, ?c) ^ Component(?c) ^isInvolvedIn(?c,?a) ^ AssemblyActivity(?a) ^ characterizesSystemTransition (?a, ?st) ^ SystemTransition(?st) ^ SystemTransitionHasUpstreamPlace(?st, ?sp) ^ SystemPlace(?sp) ^ swrlx:makeOWLThing(?pta, ?m)^ isComposedByFCT(?m, ?fct) -> SCArcHasSystemPlace(?pta, ?sp) ^ SCArcHasFCT(?pta, ?fct) ^ SystemToComponentArc(?pta)

The screenshot displays the RAASO Semantic Web Editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, Ontop, Mastro, and Help. The main window shows a class hierarchy on the left, a central query editor, and a results pane at the bottom. The query is: `Module(?m) ^ characterizesComponent(?m, ?c) ^ Component(?c) ^isInvolvedIn(?c,?a) ^ AssemblyActivity(?a) ^ characterizesSystemTransition (?a, ?st) ^ SystemTransition(?st) ^ SystemTransitionHasUpstreamPlace(?st, ?sp) ^ SystemPlace(?sp) ^ swrlx:makeOWLThing(?pta, ?m)^ isComposedByFCT(?m, ?fct) -> SCArcHasSystemPlace(?pta, ?sp) ^ SCArcHasFCT(?pta, ?fct) ^ SystemToComponentArc(?pta)`. The results pane shows a list of instances for the class `SystemToComponentArc`.

Figure 86 SWRL Tab: Step 16. System to Component Arc

Annex C – Typical deviations for process industry

Table 28 Typical deviations for process industry

	More	Less	None	Reverse	As well as	Part of	Other than
Flow	more flow	less flow	no flow	reverse flow	deviating concentration	contamination	deviating material
Pressure	high pressure	low pressure	vacuum	-	delta-p		explosion
Temperature	high temperature	low temperature	-	-			
Level	high level	low level	no level		different level		
Reaction	fast reaction / runaway	slow reaction	no reaction				unwanted reaction
Time	too long / too late	too short / too soon	sequence step skipped	backward	missing actions	extra actions	wrong time
Agitation	fast mixing	slow mixing	no mixing				
Start-up / Shut-down	too fast	too slow			action missed		wrong recipe
Draining / Venting	too long	too short	none		deviating pressure	wrong timing	
Inertising	high pressure	low pressure	none			contamination	wrong material
Utility failure			failure				
DCS failure			failure				
Maintenance			none				
Vibrations	too low	too high	none				wrong frequency

Curriculum Vitae

Damiano Nunzio Arena

Route de Bussigny 6, 1023 Crissier, Vaud, Switzerland

+41 76 612 39 99 | damiarena@gmail.com

<https://linkedin.com/in/damianonunzioarena/>

Natl: Italian | DOB: 7.7.1985



Summary

- Doctoral candidate in Advanced Manufacturing.
- Focus on Digital Transformation and Semantic Web Technologies in the framework of Industry 4.0.

Work Experience

Research Assistant

April. 2015 - Present

ICT for Sustainable Manufacturing Research Team, EPFL

Lausanne, Switzerland

- Industrial research project on Semantically-Enriched Manufacturing System Modelling and Simulation.
- Teaching Assistant in Design for X and Product Lifecycle Management courses.
- Published more than 10 research papers and presented the results of my research in conferences and seminars.

In parallel to this, I have been working on some EU funded research and innovation projects:

Project Leader

January 2019 - Present

- **QU4LITY EU Project:** Digital Reality in Zero Defect Manufacturing (ZDM).
- **Topic:** Digital Manufacturing Platforms for Connected Smart Factories.
- **Main Results:**
 - Definition of an Autonomous Quality (AQ) paradigm in ZDM for operational efficiency increase, scrap reduction, prescriptive quality management, energy efficiency, and defect propagation avoidance.
 - Specification of Digital Models and Vocabularies for Digital Twins (DT) to ease data interoperability across the ZDM equipment and processes, simulation, and implementation of DT applications.

Project Leader

March 2018 - Present

- **DILECO EU Project:** Digitalization of ground-testing Life cycle with ECO design criteria.
- **Topic:** PLM Tools for Aircraft Ground Functional testing with Eco-design criteria.
- **Main Results:**
 - Design of an Ontological model for Semantics-based System Interoperability.
 - Development and deployment of a PLM Tool for Aircraft Ground Functional testing with Eco-design criteria.

Project Leader

April 2015 – December 2017

- **SATISFACTORY EU Project:** A collaborative and augmented-enabled ecosystem for increasing SATISfaction and working experience in smart FACTORY environments.
- **Topic:** Developing smart factories that are attractive to workers.
- **Main Results:**
 - Development of a Knowledge-base for Human Resource Optimization.
 - Implementation of a Semantically-enriched Framework for Analysis and Design of shop floor Operations.

Education

Ph.D. in Advanced Manufacturing

April 2015 – May 2019

École Polytechnique Fédérale de Lausanne (EPFL)

Lausanne, Switzerland

- Research Field: Information and Communication Technologies for Sustainable Manufacturing
- Ph.D. Thesis: Towards Semantics-Driven Modelling and Simulation of Context-Aware Manufacturing Systems.

M.Sc. in Management Engineering (110/110 cum laude)

October 2012 – November 2014

Università degli Studi di Catania (UNICT)

Catania, Italy

- Major: Industrial Systems, Production Planning, Logistics, Marketing
- M.Sc. Thesis - A new tool for risk assessment of complex system based on HAZOP and Colored Petri Nets

B.Sc. in Management Engineering (103/110)

October 2008 – November 2012

Università degli Studi di Catania (UNICT)

Catania, Italy

- Major: Probability and Statistics, Micro/Macroeconomics, Productions Systems Modelling and Management
- B.Sc. Thesis: Enterprise information flow and ERP module optimization for product returns optimization

Skills

- **Languages:** Italian (Native), English (C2), French (B2), Spanish (A2).
- **Programming:** VBA, XML, RDF, OWL, R, Python, JavaScript, Standard ML.
- **Software:** Microsoft Word, Excel, Power-Point, Visio, and Project, Adobe Illustrator, Photoshop, Autodesk Inventor, Tableau, Boothroyd Dewhurst DFMA, Anzo Smart Data lake, Open Semantic Framework.
- **Other:** Project Management, Public Speaking, Team Leadership, Critical Thinking, Problem Solving.

Honours & Awards

- **Burbigde Award for the Best Paper** at the APMS 2017 International Conference. Title of the paper: An Ontology-based model for training evaluation and skill classification in an Industry 4.0 environment.
- **PhD Research Proposal Award** at the PLM Jul 2017 International Conference. Title of the PhD research proposal: Towards Semantics-Based Analysis of Context-Aware Manufacturing Systems.

Publications

Journal Articles

1. Arena, Damiano; Tsolakis, Apostolos Charalampos; Zikos, Stylianos; Krinidis, Stelios; Ziogou, Chrysovalantou; Ioannidis, Dimosthenis; Voutetakis, Spyros; Tzouvaras, Dimitrios; Kiritsis, Dimitris; Human resource optimisation through semantically enriched data, *International Journal of Production Research*,56,8,2855-2877,2018, Taylor & Francis
2. Arena, Damiano; Trapani, Natalia; Kiritsis, Dimitris; Context-aware CPN for HAZOP assessment of chemical plants, *International Journal of Production Research*, Taylor & Francis (UNDER REVIEW)

Conference Proceedings

1. Arena, Damiano; Kiritsis, Dimitris; Trapani, Natalia; A behaviour model for risk assessment of complex systems based on HAZOP and Coloured Petri Nets, *IFIP International Conference on Advances in Production Management Systems*,573-581,2015,"Springer, Cham"
2. Tsolakis, Apostolos; Arena, Damiano; Krinidis, Stelios; Perdikakis, Apostolos; Ioannidis, Dimosthenis; Kyritsis, Dimitrios; Tzouvaras, Dimitrios; Semantically enriched industry data & information modelling: A feasibility study on shop-floor incident recognition,2016 IEEE 14th International Conference on Industrial Informatics (INDIN) 487-491,2016, IEEE
3. Arena, Damiano; Kiritsis, Dimitris; A Method Towards Modelling and Analysis of Semantically-Enriched Reconfigurable Manufacturing Systems, *IFIP International Conference on Advances in Production Management Systems*, 45-52,2016,"Springer, Cham"
4. Arena, Damiano; Ziazios, K; Metaxa, Ifigeneia N; Parcharidis, S; Zikos, S; Tsolakis, A; Krinidis, Stelios; Ioannidis, Dimosthenis; Tzouvaras, Dimitrios; Kiritsis, Dimitris; Towards a Semantically-Enriched Framework for Human Resource Management, *IFIP International Conference on Advances in Production Management Systems* 306-313,2017,"Springer, Cham"
5. Perini, Stefano; Arena, Damiano; Kiritsis, Dimitris; Taisch, Marco; An ontology-based model for training evaluation and skill classification in an industry 4.0 environment, *IFIP International Conference on Advances in Production Management Systems* 314-321,2017,"Springer, Cham"
6. Ziogou, Chrysovalantou; Arena, Damiano; Krinidis, Stelios; Ioannidis, Dimosthenis; Kiritsis, Dimitrios; Tzouvaras, Dimitrios; Voutetakis, Spyros; Decision Support based on a Semantically-Enriched Notification Platform at a Process Plant Floor, *Computer Aided Chemical Engineering*,40 ,2365-2370, 2017, Elsevier
7. Arena, Damiano; Kiritsis, Dimitris; A methodological framework for ontology-driven instantiation of petri net manufacturing process models, *IFIP International Conference on Product Lifecycle Management*, 557-567,2017,"Springer, Cham"
8. Arena, Damiano; Kiritsis, Dimitris; Ziogou, Chrysovalantou; Voutetakis, Spyros; Semantics-driven knowledge representation for decision support and status awareness at process plant floors,"2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC)", 902-908,2017, IEEE
9. Arena, Damiano; Perini, Stefano; Taisch, Marco; Kiritsis, Dimitris; The Training Data Evaluation Tool: Towards a unified ontology-based solution for industrial training evaluation, *Procedia Manufacturing*, 23, 219-224,2018, Elsevier
10. Arena, Damiano; Ameri, Farhad; Kiritsis, Dimitris; Skill Modelling for Digital Factories, *IFIP International Conference on Advances in Production Management Systems*, 318-326,2018,"Springer, Cham"
11. Arena, Damiano; Criscione, Francesco; Trapani, Natalia; Risk assessment in a chemical plant with a CPN-HAZOP Tool, *IFAC-PapersOnLine*,51,11,939-944, 2018, Elsevier

