Quality of (Digital) Services in e-Government

Tutor
Prof. Flavio Corradini

Dottoranda
Dott.ssa Barbara Re

Anno Accademico 2009-2010
Quality of (Digital) Services in e-Government

Advisor: Prof. Flavio Corradini
PhD Candidate: Dott.ssa Barbara Re

Academic Year 2009-2010
After the rain it comes the sun
It will be there for everyone
First of all I trust in you
More than that I love you

Yuppie Flu
Glueing All The Fragments
To my Family,
my love Ivan
and my best friend Chiara.
Abstract of the Dissertation

Internet growth in the nineties supported government ambition to provide better services to citizens through the development of Information and Communication Technologies based solutions. Thanks to the Lisbon conference, which in 2000 covered and investigated this topic, e-government has been recognized as one of the major priorities in Public Administration innovation process. As a matter of fact in the last 10 years the number of services provided to citizens through Information and Communication Technologies has increased rapidly. Nevertheless the increasing rate, the access and usage of digital services do not follow the same trend. Nowadays Public Administrations deliver many electronic services which are seldom used by citizens. Different reasons contribute to the highlighted situation.

The main assumption of the thesis is that quality of e-government digital services strongly affects real access to services by citizens. According to the complexity of quality in e-government, one of the main challenges was to define a suitable quality model. To reach such aim, domain-dependent characteristics on the services delivery have been investigated. The defined model refers to citizen, technology and service related quality characteristics. Correspondingly a suitable way to represent, assess, and continuously improve services quality according to such domain requirements has been introduced.

Concerning the service related quality aspects a methodology and a tool permitting to formally and automatically assess the quality of a designed service with respect to the quality model has been defined. Starting from an user friendly notation, both for service and quality requirements, the proposed methodology has been implemented as an user friendly tool supported by a mapping from user friendly notations to formal language. The tool allows to verify formally via model checking, if the given service satisfies one by one the quality requirements addressed by the quality model.

Additionally in some case an unique view on e-government service quality is
quite useful. A mathematical model provides a single value for quality starting from the assessment of all the requirements defined in the quality model. It relies on the following activities: homogeneity, interaction and grouping.

A set of experiments has been performed in order to validate the goodness of the work. Services already implemented in a local Public Administration has been considered. Literature review and domain experts knowledge were the main drivers of this work. It proofs the goodness of the quality model, the application of formal techniques in the complex field of study such as e-government and the quality aggregation via the mathematical model.

This thesis introduces advance research in e-government by providing the contributions that quality oriented service delivery in Public Administration promotes services used by the citizens. Further applications of the proposed approaches could be investigated in the areas of practical benchmarking and Service Level Agreement specification.
Acknowledgements

The writing of a dissertation can be a lonely and isolating experience, yet it is obviously not possible without the personal and practical support of numerous people.

First of all, I would like to express my gratitude to my advisor Prof. Flavio Corradini that with his experience and wisdom introduced and guided me in these years. My gratitude also goes to Ing. Alberto Polzonetti for his valuable support and consistent encouragement through my study.

I am thankful to Dott. Andrea Polini for the cooperation and the time spent in long discussions that contributed to my work.

I am also greatly indebted to many teachers in the past: among others Prof.ssa Emanuela Merelli and Dott. Giuseppe Alessandri for getting me interested in computer science research field.

My enormous debt of gratitude can hardly be repaid to my best friend Chiara Ercoli, that with patient and love support me during this long experience even when face with the greatest adversities. Thanks to all of my friends and colleagues of Phd course Oliviero Riganelli, Francesco De Angelis, Damiano Falcioni and Leonardo Vito. We not only studied, relaxed, and traveled well together but we share part of our precious daily life.

Among the other I have had also the great pleasure of working with a special set of people, excellent collaborators and friends in the staff of e-Lios s.r.l. the-UNICAM Spin-off Company. I thank all of them: Chiara Ercoli (again!) Federico Paoloni, Elia Brugnoni, Manuel Fabbrizi, Eleonora Paganelli, Paola Staffolani, Francesco Maccari and Anna Ascenzi.

A special thank to Dott. Fausto Marcantoni and Ing. Roberto Gagliardi for share with the Unicam e-Government Research Group their experiences.

There are also two important stages during the Phd experience corresponding to two research stays in the USA and Switzerland. I am grateful to all the people that I met in my first experience in Seattle (USA). Im thankful to Dr. Joachen
Shooll and his research group for the helpful discussions that made me aware of the completeness and richness of the e-government application domain.

The second was in Olten (Switzerland), the town more welcoming than you can imagine in the middle of the Europe. First of all I have to thank Prof. Knut Hinkelmann that gave me the opportunity to work with the Information and Knowledge Management Group. I will never forget the time spent at FHNW and the helpful discussions. Let me also thank the following people at FHNW: Barbara Thonssen, Rainer Telesko, Holger Wache, Daniela Feldkamp, Roman Brun, Andreas Martin and Simon Nikles. Thanks for the kind welcome and support.

I cannot forget the Spanish colleagues Prof. Luis Anido Rifon and Luis Alvarez Sabucedo. Thanks for fruitful collaboration that drives me toward the first cross-national experience of a research project.

Notable is also the support that some institutions gave me in different ways during the development of this work. First of all the University of Camerino and the School of Advance Study. I would like to thank University of Macerata for giving me the possibility to teach there, Marche Region and CNIPA (now DigitPA) for their pragmatical contribution on the research activities.

Finally, I must thank my family: my parents Franco and Mariella, my sister Milena for the untiring support and trust. I owe most of what I have learned to all these people and I will never be able to thank enough. Last but not least, I have to thank Ivan for his unconditioned love and patience.
List of Publications

Journal Papers


Book Chapters


International Conference Papers


C7 Flavio Corradini, Alberto Polzonetti, Barbara Re. The Meaning and the Role of Adaptability in e-Government. In the proceeding of 7th International Conference EGOV 2008, Proceedings of ongoing research, project contributions and workshops, pp. 70 - 77, Turin (Italy), 1-4 September 2008.


National Conference Papers


# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract of the Dissertation</td>
<td>vii</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>ix</td>
</tr>
<tr>
<td>List of Publications</td>
<td>xi</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xix</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xxiii</td>
</tr>
</tbody>
</table>

## I Introduction

### 1 Introduction

1.1 e-Government and Digital Services                                   | 3    |
1.2 Rational Behind the Research                                        | 4    |
1.3 Research Aim and Thesis Contributions                               | 8    |
1.4 Methodological Approach                                             | 9    |
1.5 Structure of the Thesis                                             | 11   |

## 2 e-Government and Digital Services

2.1 Research Context                                                    | 13   |
2.2 Cross-Disciplinary Research Field                                   | 15   |
2.2.1 Involved Disciplines                                             | 15   |
2.2.2 Research Themes                                                  | 18   |
2.3 Service Delivery in e-Government                                    | 21   |
2.3.1 Definitions and Characteristics                                  | 22   |
2.3.2 Classifications                                                  | 22   |
2.4 e-Government Architecture                                           | 23   |
CONTENTS

2.4.1 Reference Frameworks ........................................... 24
2.4.2 Service Oriented Architecture .............................. 26
2.4.3 Web Service ................................................... 28
2.5 Frameworks for Service IT Alignment ...................... 29
2.6 European and National Scenarios ......................... 31
   2.6.1 European Action Plans ..................................... 32
   2.6.2 Italian Action Plan ......................................... 34

3 Quality in Different Application Domains .................... 37
   3.1 Customer/Citizen Satisfaction ............................... 37
   3.2 Organizations Quality ........................................ 41
   3.3 Process Quality ............................................... 42
   3.4 Web Portal Quality ............................................ 44
      3.4.1 e-Commerce Domain .................................... 44
      3.4.2 e-Government Domain ................................. 45
      3.4.3 Web Site Usability ..................................... 47
   3.5 Web Service Quality ........................................... 47
   3.6 Network Quality ................................................ 49

II A Systematic View of Quality in e-Government ............ 53

4 A Quality Model for e-Government ............................. 55
   4.1 Introduction .................................................. 55
   4.2 Overview of Quality in e-Government .................... 56
   4.3 e-Government Subjective Parameters .................... 58
   4.4 Technology Related Parameters ........................................... 59
   4.5 Service Quality Parameters ........................................... 63
      4.5.1 Coordination .............................................. 63
      4.5.2 Control .................................................. 64
      4.5.3 Sharing .................................................. 65
      4.5.4 Transparency .......................................... 66
      4.5.5 Inclusion ................................................ 67
   4.6 Validation of the Model ........................................ 68
      4.6.1 Subjective and ICT Parameters ......................... 68
      4.6.2 Service Parameters ..................................... 70
   4.7 Considerations ................................................ 70
III  A Formal Methods in e-Government

5  Background: Business Process Management

5.1 History of Business Process Management
  5.1.1 Evolution of Management Area
  5.1.2 Evolution of Workflow Area

5.2 Understanding Business Process Management
  5.2.1 Business Process Definitions
  5.2.2 Business Process Life-Cycle

5.3 Business Process Design
  5.3.1 Business Process Modeling
  5.3.2 Business Process Modeling Notation
  5.3.3 BPMN vs. UML Activity Diagram

5.4 Quality in Business Process Management
  5.4.1 Benchmarking
  5.4.2 Metrics in Business Process

5.5 Applying the Technology in Business Process Management

5.6 Business Process in e-Government

6  Background: Process Algebras and Formal Verification

6.1 Process Algebra

6.2 Communication Sequential Process

6.3 Formal Verification

6.4 Formalizing Business Process

7  Quality of Digital Services in e-Government

7.1 Introduction

7.2 From Specification to Verification
  7.2.1 BPMN2CSP
  7.2.2 Domain Related Quality Requirements and Mapping
  7.2.3 Verification

7.3 BP4PA: A Tool Chains for Business Process Verification in the
  e-Government Domain
  7.3.1 Implementation Details for the BP4PA Plug-In
  7.3.2 Specification and Verification Issues

7.4 BP4PA in Use: Case Studies
  7.4.1 Newborn Registration Service
  7.4.2 Moving Service

7.5 Useful Approaches on Applying Formal
  Methods
  7.5.1 Formal Models in e-Government
CONTENTS

7.5.2 User Friendly Model Checking ........................................ 131
7.5.3 Services Composition Verification ................................. 133
7.6 Considerations ............................................................. 135

IV Quality Aggregation ......................................................... 137

8 Aggregation Model ............................................................... 139
8.1 Introduction ................................................................. 139
8.2 Terminology ................................................................. 140
8.3 Measurement and Derivation of Input Values ......................... 141
8.4 Aggregation Model ......................................................... 142
8.5 Experimentations .......................................................... 143
  8.5.1 Quality Behavior ....................................................... 143
  8.5.2 Dependences Among Parameters ................................. 145
8.6 Considerations ............................................................. 146

V Conclusions ................................................................. 149

9 Conclusions and Further Work ........................................... 151

Bibliography ................................................................. 153

Appendix A - e-Government Availability and Use .................... 177
Appendix B - Mapping from BPMN and CSP ......................... 181
Appendix C - BPMN Modeler Tools ....................................... 257
Appendix D - Acronyms ....................................................... 259
# List of Figures

1.1 e-Government on-Line Availability (Source Eurostat) 6
1.2 e-Government Usage by Individuals by Gender (Source Eurostat) 6
1.3 e-Government Usage by Enterprises (Source Eurostat) 7
1.4 Methodological Approach Followed During Research 9

2.1 Ramadier’s Disciplinary Toward Trans-disciplinary (Source [167]) 16
2.2 Main Research Areas (Source [221]) 18
2.3 e-Government Research Themes (Source [38]) 19
2.4 Framework of e-Government Architecture (Source [55]) 25

3.1 ACSI Model 38
3.2 Kano Model (Source [146]) 39
3.3 SERVQUAL Model (Source [159]) 40
3.4 BSC Model 42
3.5 CAF Model 43

4.1 Layers of Quality Assessment (Source [84]) 56
4.2 Our Quality View: Quality in e-Government 57

5.1 BPM Hype Cycle (Source [105]) 78
5.2 History of Workflow Research (Source [246]) 79
5.3 History of Commercial Workflow Systems (Source [246]) 80
5.4 Business Process Life-cycle 82
5.5 BPMN Diagram of Business Process Life-cycle 83
5.6 BPMN Elements 87

7.1 Proposed Approach’s Logical View 103
7.2 Mapping Rules From BPMN Pool sending to CSP 107
7.3 Mapping Rules From BPMN Flow to CSP ........................................... 107
7.4 Mapping Rules From BPMN Start Event to CSP ......................... 108
7.5 Mapping Rules From BPMN Task to CSP ...................................... 108
7.6 Mapping Rules From BPMN Task Sending to CSP ....................... 109
7.7 Mapping Rules From BPMN Task Receiving to CSP ..................... 110
7.8 BP4PA Tool Chain ...................................................................... 113
7.9 PAT Architecture ...................................................................... 115
7.10 Coordination Check: Collaboration ............................................. 117
7.11 Proactive Control Level ........................................................... 118
7.12 Reactive control level .............................................................. 119
7.13 No Specific Control Level .......................................................... 119
7.14 Sharing Check: Data Sharing ..................................................... 120
7.15 Transparency Check: Role Transparency .................................... 121
7.16 Inclusion Check: Channel Inclusiveness ..................................... 121
7.17 Case Study: Service Architecture ............................................. 124
7.18 Case Study: Newborn Registration Service ............................... 126
7.19 Case Study: Moving Service .................................................... 129
8.1 QoS Assessment Model Phases ................................................... 141
8.2 Quality Distribution Low interaction .......................................... 144
8.3 Quality Distribution High interaction ......................................... 145
8.4 Frequency Distribution of Quality ................................................ 146
8.5 Dependence Between Time Processing and Time Data Retrieval ...... 147
8.6 Dependence Between Time Execution and Time Data Retrieval ...... 148
8.7 Dependence Between Time Processing and Time Execution ............ 148
9.1 e-Gov Service Availability - Part I (Source Eurostat) ....................... 177
9.2 e-Gov Service Availability - Part II (Source Eurostat) ...................... 177
9.3 e-Gov Service Use by Citizens - Part I (Source Eurostat) ............... 178
9.4 e-Gov Service Use by Citizens - Part II (Source Eurostat) .............. 178
9.5 e-Gov Service Use by Citizens - Female - Part I (Source Eurostat) .... 178
9.6 e-Gov Service Use by Citizens - Female - Part II (Source Eurostat) ... 178
9.7 e-Gov Service Use by Citizens - Male - Part I (Source Eurostat) ....... 179
9.8 e-Gov Service Use by Citizens - Male - Part II (Source Eurostat) ........ 179
9.9 e-Gov Service Use by Enterprise - Part I (Source Eurostat) ............. 179
9.10 e-Gov Service Use by Enterprise - Part II (Source Eurostat) ........... 180
9.11 Mapping Pool ......................................................................... 182
9.12 Mapping Lane ......................................................................... 182
9.13 Mapping Sequence Flow ........................................................... 184
9.14 Mapping Conditional Sequence Flow ......................................... 185
LIST OF FIGURES

9.15 Mapping Empty Start Event (one Output Flow) .......................... 186
9.16 Mapping Empty Start Event (n Output Flows) ......................... 187
9.17 Mapping Start Event with Conditional Type ........................... 188
9.18 Mapping Start Event with Signal Type ................................. 189
9.19 Mapping Empty End Event ............................................... 191
9.20 Mapping End Event with Signal Type .................................. 192
9.21 Mapping End Event with Terminate .................................... 194
9.22 Mapping End Event with Terminate in a Subprocess .................. 194
9.23 Mapping End Event with Error ......................................... 196
9.24 Mapping End Event With Error in a Subprocess ....................... 197
9.25 Mapping Empty Intermediate Event ................................. 198
9.26 Mapping Timer Intermediate Event .................................... 199
9.27 Mapping Conditional Intermediate Event .............................. 200
9.28 Mapping Signal Intermediate Event (Catching) ....................... 202
9.29 Mapping Signal Intermediate Event (Throwing) ...................... 203
9.30 Mapping Error Intermediate Event (Catching) ....................... 205
9.31 Mapping Link Event (Catching) ........................................ 206
9.32 Mapping Link Event (Throwing) ....................................... 207
9.33 Mapping Task ............................................................ 209
9.34 Mapping Task with Standard Loop .................................... 211
9.35 Mapping Task with Multi Instance in Sequence ....................... 214
9.36 Mapping Task with Multi Instance in Parallel ......................... 216
9.37 Mapping Subprocess ...................................................... 218
9.38 Mapping Looping Subprocess .......................................... 221
9.39 Mapping Subprocess with Multi Instance in Sequence ............... 224
9.40 Mapping Subprocess with Multi Instance in Parallel ............... 227
9.41 Mapping Subprocess with Intermediate Error Event at the Border .. 229
9.42 Mapping Exclusive (Data-Based) Splitting ............................. 232
9.43 Mapping Exclusive (Data-Based) Merging ............................. 234
9.44 Mapping Inclusive Gateway (splitting) ............................... 236
9.45 Mapping Inclusive Gateway (merging) ................................ 238
9.46 Mapping Parallel Gateway (splitting) ................................ 240
9.47 Mapping Parallel Gateway (merging) ................................ 241
9.48 Mapping Data Object .................................................... 242
9.49 Mapping Send Task ....................................................... 245
9.50 Mapping Receive Task ................................................... 246
9.51 Mapping Receiving Start Event ....................................... 247
9.52 Mapping End Event Sending ............................................ 248
9.53 Mapping Intermediate Event Sending ................................. 249
9.54 Mapping Receiving Intermediate Event ............................... 250
9.55 Mapping Loop Task Sending .......................................... 251
LIST OF FIGURES

9.56 Mapping Receiving Loop Task . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 252
9.57 Mapping Multiple Instance Task in Sequence Sending . . . . . . . . . . . . . . . . . . 253
9.58 Mapping Multiple Instance Task in Sequence Receiving . . . . . . . . . . . . . . . . . . 254
9.59 Mapping Multiple Instance Task in Parallel Sending . . . . . . . . . . . . . . . . . . . 255
9.60 Mapping Multiple Instance Task in Parallel Receiving . . . . . . . . . . . . . . . . . . 256
List of Tables

3.1 BSC vs CAF ......................................................... 51
4.1 BP4PA Quality Dimensions ....................................... 63
4.2 Macro-Areas Weight Based on the Users ICT Expertise .... 69
4.3 Macro-Areas Weight Based On The Users Age ............... 70
Part I

Introduction
Chapter 1

Introduction

European and world-wide research agenda presents many exciting research challenges since e-government has been introduced. It goes beyond the frontiers of traditional Information Communication Technology (ICT) by fostering cross-disciplinary research collaborations around novel research ideas and themes. It produces new practices changing the way of doing research. This thesis presents a novel approach within the cross-disciplinary area of e-government.

This Chapter is devoted to introduce thesis research context, main reasons and aims of the research as well as the methodological approach of the work. Structure of the thesis concludes the Chapter.

1.1 e-Government and Digital Services

In the last 10 years an increasing interest has been devoted to e-government by the research community. European Commission (EC) and most of the national member states invested on it as one of the major priorities in Public Administration (PA) innovation process. Today e-government is a global pervasive phenomenon. Both government and citizens benefit from the advantages of e-government. On the one hand, management administration cost is reduced, internal efficiency is improved and resources allocation is better governed. On the other hand, administrations are at citizens disposal 24 hours a day and 7 days a week without any moving to government offices.

To better understand the phenomenon we refer to a well know e-government definition. It is given by the European Commission and it refers to e-government as:

“the use of Information and Communication Technologies in Public Administrations combined with organizational change and new
skills in order to improve public services and democratic processes and strengthen support to public policies” [68].

As to the above definition, public services, democratic processes and public policies are the main components in e-government. Our focus is on services according to the way they deliver to the citizens. As a matter of fact e-government digital services are the main channel between PA and citizens and their importance is also recognized by European Commission [68]. In most of the cases various PAs contribute to the service delivery where a collection of business activities and informative resources become available through the usage of Information an Communication Technology. It is important to remark that ICT is just one aspect in the delivery.

More detailed discussion on e-government and services delivery will follow in this thesis.

1.2 Rational Behind the Research

A lot of money was devoted to e-government by European Commission and Member States striving its diffusion. However the complexity of e-government domain and a mature ICT impact on services development and delivery. A lot of e-government project are potentially successful, but proving quite simple service implementation they result useless in most of the case.

In November 2009 the most recent European Commission study in the context of e-Europe Benchmarking framework was published. It shows that 83% of the basic services are available on-line [40]. The resulting percentage increased towards the results of similar surveys conducted in 2007 and 2006 and characterized by 76% [39] and 60% [215] of services availability respectively.

Nevertheless a different perception is evident if e-government efficacy is considered. Indeed many of the available digital services are seldom accessed and used by citizens. According to the Accenture study published in 2005 citizens are not exactly queuing up to use e-government services even when these are available [1]. It has been estimated that up to two-thirds of citizens in industrialized countries access e-government services less than once per year; and only two-thirds of citizens in such countries have ever accessed such services. It is observed that in most of the European countries e-government projects are predominantly politically and not economically, socially or organizationally driven [112]. As mentioned by Cene Bavec in the paper title “On stimulus for citizens’ use of e-government services” “practical experiences and researches confirm that users’ acceptance is not guaranteed itself. Public approval is quite often below what developers expected” [13]. In other words citizens do not use e-government
services just because they are available. Such scenario is also described by more scientific contributions (see for instance [210]).

Going into detail we refer to structural indicators on e-government availability and usage by individuals and enterprisers as provided by Eurostat. For each of them we compare the European scenario with the national one. The indicator in Figure 1.1 shows the percentage of the 20 basic services which are fully available online i.e. for which it is possible to carry out full electronic case handling. Measurement is based on a sample of URLs of public websites agreed with Member States as relevant for each service. Italy shows a more solid scenario than the European average services development. Theoretically the Italian citizens if they wish can use digital services to interact with administrations. The indicators in Figure 1.2 shows the percentage of individuals aged 16 to 74 using the Internet for interaction with public authorities (i.e. having used the Internet for one or more of the following activities: obtaining information from public authorities websites, downloading official forms, sending filled in forms). In this case it is clear that the citizens don’t use the services just because they are available on-line. Other policies are at the base of the real involvement of citizen in on-line Public Administration. The indicator in Figure 1.3 shows the percentage of enterprisers using the internet to interact with public authorities. In this case the scenario is completely different respect to the usage of the service by individuals. Concerning enterprisers the digital channel for law and regulation request is rather use than the traditional. It means that the general policy is that business has to use the e-government services and citizens should use the services. As a matter of fact the research showed that European governmental bodies developed above all services that produce a direct income for PAs supplying such services.

Many different reasons contribute to the highlighted situation. Certainly the well known digital divide phenomenon [147] has to be considered one of the possible causes for low digital services usage. Nevertheless our impression, supported by a little investigation conducted among the people in our department and in local Public Administrations is that other important causes strongly contribute to reduced digital services usage.

Indeed digital service delivery strategies are often too much focused on technological aspects skipping social, anthropological and organizational views. In most of the cases services delivery processes just reflect already available processes, typically based on “only-human based” interactions. Defining a good digital services process is certainly important to take advantage of the new opportunities given by the usage of ICT technologies, nevertheless social, anthropological and organizational characteristics should not be forgotten. So, for instance, in a “only-human based” process citizen trust is raised by direct interactions with

---

1http://epp eurostat ec europa eu/portal/page/portal/information_society/introduction
CHAPTER 1. INTRODUCTION

Figure 1.1: e-Government on-Line Availability (Source Eurostat).

Figure 1.2: e-Government Usage by Individuals by Gender (Source Eurostat).
CHAPTER 1. INTRODUCTION

Figure 1.3: e-Government Usage by Enterprises (Source Eurostat).

the civil servant, which represents citizens direct access point to information concerning the request and its execution. Delivering a digital service it is important to keep in mind that access to a web site does not provide itself the same level of trust. Too many things remain hidden in the perception of the citizen. Then it is important to revise the process in order to introduce mechanisms that can help to increase citizen trust. It is somehow paradigmatic the case of an insurance company in Italy which is advertising a new way of delivering the customer care support service. In particular to increase trust perception all the incoming calls from a customer are always redirected to the same phone operator. In this case the idea is to create a more trustable connection with the organization. Similar considerations should be taken into account when planning services delivery in e-government.

Last but not least, a fully integration among the involved administrations has to be promoted. The implementation of a digital interface is not enough to transform the e-government in term of organizational efficiency and citizens satisfaction. Concrete inter-administration procedures supporting back-office and front-office coordination have to be implemented to improve the only human based process implementation too often realized. An interesting metaphor shows the PA front-office like a horse and the back office like a cow. The administrative challenge
orients digital government towards suitable digital service delivery strategies.

1.3 Research Aim and Thesis Contributions

This thesis aims at contributing to the field of e-government digital service novel approach in term of methodologies, technologies and tools to improve services quality, so to strive their use by the citizens. We believe that our approach provides a suitable starting point to drive e-government toward the true transformation [175]. The specific research questions are reported below.

- What are the services related quality requirements for e-government domain that impacts on the use of service by the citizens?
- How to check quality requirements in e-government services?
- How to aggregate quality requirements?

According to the research questions we have tried to help advance research by providing the following contributions.

- Find out domain dependent characteristics that strongly impact on the quality and the use of e-government digital services by the citizens.
- Provide suitable approaches to represent, assess and continuously improve the quality of services, according to domain dependent characteristics output of the previous step. On one site we stress quality of process at business level and on the other site quality of enabling ICT at technological level.
- Provide an aggregation model enabling homogenization, interaction and grouping.

The approaches have been implemented throughout the user-friendly tools enabling quality oriented e-government.

It is worth noting that in line with what is recommended in [178], our proposal can be used for practical benchmarking of PA digital services delivery process facing some important issues still unexplored by the current benchmarking studies [24]. Another important application of our proposal refers to Service Level Agreement (SLA). Due to limited staff and budget project out-sourcing it is often the adopted solution within Public Administrations. Our approach represents a suitable way to enrich contracts with explicit, clear and strict specifications that have to be guaranteed by software producers. At the same time our approach could enrich service level agreement typically introduced in e-government applicative cooperation [11] too often just technology oriented.
1.4 Methodological Approach

This section describes the methodological approach of how the research work is carried out in order to answer proposed research questions. Notable is the characterization on the research field introduced below as a cross-disciplinary area of study. We will contribute focusing on two main areas such as public management and computer science. The methodological approach drives our work and the structure of this thesis. Figure 1.4 shows how the methodology is followed.

Figure 1.4: Methodological Approach Followed During Research.

The research structure of the thesis is divided into five phases. The first phase
is actually the selection of research domain - in our case it is e-government - and the evaluation of the context knowledge about that domain. Further European Union (EU) studies evaluation and background experiences described on the first phase and contribute as input at the research. It leads to the identification of the problem existing in the selected research domain. Our intention was to improve e-government service use via a quality oriented approach. Then the problems are analyzed and research questions are created in order to settle few aims that are to be achieved at the end of the thesis. The research questions are broken down further by setting a small objective which helps to achieve the main goal of the research. In order to reach to a specific objective, certain strategies has been created and the lead to the accomplishment of small objectives.

In order to meet the expected outcome of the thesis, a literature review was carried out to gain primarily the domain knowledge on quality in e-government and in similar areas such as quality management in organization. The e-government characteristics were studied during this phase. Interesting works were also analyzed in the area of e-business and they inspired the final solution. After the first round where basically the first proposal in particular concerning the quality requirements investigation were taken from a literature review. Many meetings were held with practitioners and domain experts with the brainstorming purpose. The meetings ended with fruitful results enabling the validation of the solution.

The gained knowledge has been later used to propose final solution toward (i) development of a quality vocabulary, (ii) formalization of the measurements and (iii) definition of a quality aggregation function. The first phase - quality definition - refers to the identification of parameters and related metrics at different levels according to the e-government quality views. Such views are citizens satisfaction, ICT related quality, process quality, and organization quality and they are discussed in detail the following of the thesis. Starting from a review of the e-government domain this phase defines a comprehensive e-government quality model. We have investigated the domain at different abstraction levels and we have developed a quality vocabulary. We consider parameters as attraction, availability and execution time (just to cite a few). Related to metrics we define a unit of measure that is in line with a specific procedure for quality measurement. The second phase - quality measurement - refers to the formalization of the measurements. We have introduced a model that allows a suitable abstraction level of the problem description and a formal background of the applicative solutions. The model is scalable with respect to the considered set of parameters. Starting from the parameters taxonomy previously detected and the classification given by quality views we investigated suitable measurement models and approaches. Notable is the decision to apply in the quality check, the mature formal techniques based on the mathematical theories and supported by the industrially developed tools and methods. As discussed by Davies et. al. in [47] it is the time to provide
CHAPTER 1. INTRODUCTION

a solid background to applied research field such as e-government. In this way
e-government will be more usable and accessible to the most of the citizens with
the guarantee of the e-government and accuracy. Finally, the third phase - qual-
ity interaction - introduces a mathematical model to define a quality function and
assess a quality value starting from the sets of parameters. The model can be event-
ually used when it is needed an integrate view on quality. The model relies on
parameters homogenization, interaction and grouping. In particular, homogeniza-
tion of the input is useful to reason over different metrics (for instance, time-based
measurements need to be aggregated with security-based boolean measurements
or some other metrics). The homogenization takes also into account whether a
given parameter grows in a proportional or in an inverse proportional way with
respect to the overall quality measurement. Interaction between different param-
eters is also considered by the mathematical model. In such a way we can take
into account how parameters influence each others (for instance, how trust pa-
parameter influences usability [19]). Finally, the proposed model groups parameters
and manages them with different importance. User-friendly tools where finally
implemented to assure automatically the supported the solutions.

1.5 Structure of the Thesis

The remainder of this thesis is organized as follows.

- Chapter 2 presents the necessary research context for our study. In partic-
  ular it introduces e-government domain and digital service framework in
e-government.

- Chapter 3 sums up some background and related works on the wide area of
  quality.

- Chapter 4 presents our contribution toward the definition of a systematic
  view of quality in e-government services.

- Chapter 5 proposes an overview on Business Process Management as a suit-
  able background to understand the advantages of the proposed solution.

- Chapter 6 introduces some notions on formal methods. In particular we
  provide a brief discussion on Communication Sequential Process language
  and model checking.

- Chapter 7 presents our contribution upon the area of quality of service in
e-government.
• Chapter 8 introduces an aggregation model suitable to have an unique quality value.

• Chapter 9 sums up the contribution of the thesis and it drives some future research directions.

• Appendix A, B, C and D we report details on availability and use of the e-government services, the implemented mapping from BPMN to CSP, a list on the most interesting BPMN modeling tools and finally acronyms used in the work respectively.
Chapter 2

e-Government and Digital Services

Public sector is making a solid contribution to the future improvement of Information Society (IS), even if its role is often underestimated or goes unrecognized. For years a number of research projects on IS are lunched. They are sponsored by the European Commission as well as national and local governments. The investment in public sector toward the development of Information Society critically impacts on competitiveness and innovation of the government. Consequentially, the body of e-government research is rapidly growing and key questions about e-government as legitimate discipline have been raised [182]. A comprehensive understanding of e-government as cross-disciplinary research in holistic eco-system such as PA [222] [183] [221].

This Chapter is dedicated to give an overview to e-government and digital services. Cross-disciplinary characteristics are observed as well as research themes usually investigated in e-government with a particular focus on frameworks and architectures that drive digital service and related delivery process. Finally, an overview on e-government in Europe and Italy is presented to conclude the Chapter.

2.1 Research Context

Starting from 1980’s New Public Management definition [154], the development of the Internet in the nineties supported government ambition to provide better services to citizens through the development of ICT-based solutions. Thanks to the Lisbon conference, which in 2000 covered and investigated this topic, e-government has been recognized as one of the major priorities in PA innovation process.

The complexity of e-government as applied research field provide interesting
challenges to researchers. A research community has been set up to support PAs modernization with a solid background on the different sciences [183] and interesting scientific quality contribution (see [81] [80] [82] for a qualitative review of e-government research field).

In literature, several definitions for e-government can be found. Citizens, services and administrative process are emphasize at different levels. Let us outline some of them.

- e-Government is defined as “the use of information and communication technology and its application by the government for the provision of information and basic public services to the people” [204].

- e-Government refers to “the use by government agencies of information technologies (such as Wide Area Networks, the Internet, and mobile computing) that has the ability to transform relations with citizens, businesses, and other arms of government” [200].

- e-Government “is the delivery of government information and services online thought the Internet or other digital means ” [216].

- e-Government “is a sophisticated process based on using information and communication technologies with different kind of services as result designated for satisfying stakeholders needs” [118].

Notable is the well known EC e-government definition already mentioned in the introduction section. Such definition refers to e-government as following.

“e-Government refer to the use of Information and Communication Technologies in Public Administrations combined with organizational change and new skills in order to improve public services and democratic processes and strengthen support to public policies” [68].

This definition underlines the combination of “Information and Communication Technology”, and “organization change and new skills” as the right mixture to use to provide e-government digital services without replacing the government, but making it closer to the citizens [28]. This is in line with our view on service delivery where ICT is combined at process level with the complexity of PA.

Our vision of e-government focuses on the capability of the administrations to provide services in a cooperative environment that involves Public Administrations in institutional and social setting [153] leading to a mesh of social, technical, and organizational complexities [182] [175].

Cross-disciplinary characteristics and relationships should be considered to provide successfully research. In the next sections a characterization of the research field is provided.
2.2 Cross-Disciplinary Research Field

e-Government is a field or a domain of study which spans across the boundaries of a quite a number of existing disciplines including Public Administration, political science, organizational sciences, information science, computer science, information system research, sociology, library science, statistics, low and ethics, and a lot of other disciplinary sciences [182].

Before going into the detail on the characterization of e-government as discipline or interdisciplinary research field we can best clarify the differences between disciplinary, multi-disciplinary, inter-disciplinary and trans-disciplinary as showed in Figure 2.1 and described below.

- Disciplinary community researches a problem or phenomenon based on its particular worldview including accepted methods and inquiry procedures.

- Multi-disciplinary studies use to join together two or more disciplines without integration, they highlight the different dimensions of the studied object and respect the plurality of different points of view. However, each contribution remains within its disciplinary boundaries.

- Inter-disciplinary research strives to develop a shared model of understanding of the given problem by engaging the participating disciplinary scholars in a dialog with the goal of reaching synthesis.

- Trans-disciplinary research challenges its inter-subjectivity, reflexivity and context-dependency as well as its dependence on the real possibility to be realized in practices. It implements the act of crossing disciplinary boundaries to explain one subject or method in the terms of another one.

2.2.1 Involved Disciplines

Different works prove e-government as a cross-disciplinary research field and some frameworks have been studied to integrate the various disciplines and perspectives of e-government (see for instance [220]). Figure 2.2 illustrates some of the disciplines which methods and concepts go to make up e-government. Later on we report a brief introduction on these disciplines underlining their impact on e-government as introduced by Maria Wimmer [221].

Social and Human Science. Research in this field broadens the definition of users to include a variety of stakeholders and institutional aspects. Social, economic and psychological sciences investigate these themes. The investigation includes how the users interact with governments, how governments can establish a
Figure 2.1: Ramadier’s Disciplinary Toward Trans-disciplinary (Source [167]).

(Modified after T. Ramadier, Futures 2004;36:423-439)
better relationship model with their customers and how employees interact within their organizations and across organizational boundaries. Aspects of e-inclusion and the digital divide are further topics of interest as well as the degree of new technology impacts on society and e-government role adding value to government services and productivity.

**Political Science and Jurisprudence.** They concern the impact of ICT usage on the procedures of decision-making, being it at the political or strategic level of government. This area also investigates issues of ICT supported democracy and the direct participation of citizens in democratic decision-making facilitated by ICT. Recently reactivated topics are the support of social networks as a means of liberation the individual from traditional networks of influence on democratic interactions fostered via an active marketplace of ideas. Co-governance between state institutions and civic actors, civic networks and models of creative community to facilitate concrete community involvement are already other areas of investigation in this research field. Moreover this, the impact and effect of laws in the design of ICT-supported public services is being analyzed in legal informatics. Concepts of structuring legal texts and the opportunities for ICT-support in the low drafting are among the e-government research themes in these disciplines.

**Information and Knowledge Research Sciences.** Since by its very nature the public sector deals with information and knowledge resources, it also needs the intelligent search and retrieval mechanisms. Effective support with ICT is needed for the structuring, distribution, computation, evaluation, storage and creation of knowledge. There is a pressing need for new technologies of knowledge structuring, such as ontology and semantic web service applied in e-government contexts. A special feature of this field which distinguishes it from other research work is that the users in e-government are very heterogeneous, so different means of search and visualization of information and knowledge are required.

**Public Management, Organizational Sciences and Economics.** This area develops concept of organizational structures in the public sector, including networked governments, public-public as well as public private partnerships and their effects on productivity, efficiency and legal validity. Keywords that come out of this type of research include good governance, better governance, new public management, modernizing governments, accountability, transparency, quality of service and public value generation. The area of monitoring and benchmarking can also be assigned to this domain of research.

**Computer Sciences.** This area concerns concepts and solutions for the technical implementation of e-government. Examples of research aspects are interoperability between bureaucratic systems as a whole and across regional and national borders, standardization, tools and service for seamless public service provision and by means of various communication channels, advanced comprehensive portals for digital services, electronic identification, security, encryption, digital sig-
natures, electronic payments, etc.

![Diagram showing main research areas](pill.png)

Figure 2.2: Main Research Areas (Source [221]).

By assuming such scenario as general understanding on the dimension of e-government research Scholl assumes that “the current intra- and inter-disciplinary fragmentation prevents both e-government research and practice from tapping its full potential of understanding and impact. e-Government research might be most effective when established as a multi-, inter-, and trans-discipline representing a more integrative understanding of knowing” [182].

### 2.2.2 Research Themes

e-Government research topics are investigated in the EU-founded project named eGovRTD2020[38] promoted by Codagnone and Wimmer. The study was conducted involving European and world-wide stakeholders coming from governments, ICT industry, consulting and academia. A large number of research themes have been identified by the result of a survey proposed to 261 experts in terms of importance for future e-government developments (1 not important, 6 very important) as showed in Figure 2.3.

Information on this section are extracted by a project delivery. Each theme is introduced in detail.

**Trust in e-government.** Trust is a fundamental element in all aspects of governments, including e-government. The processes by which the trust is built, destroyed, used, or abused are poorly understood and differ from one culture to another. Research is needed to understand which conditions are necessary and which mechanisms are needed to build and maintain trust in e-government processes and services. In this respect there is also a need to identify the different

---

1. www.egovrtd2020.org
kinds of trust related to e-government, e.g. trust in government or trust in ICT, and its special characteristics.

**Semantic and cultural interoperability of public services.** Globalization and population movements are making societies increasingly multicultural. In principle, increased Internet access and the potential of the web for communication and education should bridge cultural boundaries. Yet, cultural and language differences continue to block effective communication and action across different countries, lobbies, and governmental functions. To facilitate cross-organizational collaboration among the various users, semantic and cultural interoperability are preconditions.

**Information quality.** Governments, the market, and individuals increasingly need well-defined, timely, accurate, reliable and appropriate information drawn from many sources. In the future, guaranteeing information quality will become both more important and more difficult as the information sources number and variety (including informal sources such as wikis and weblogs) continue to grow.

**Assessing the value of government ICT investments.** After years of substantial investments of public funds, the potential benefits of e-government can no longer be assumed, but must be demonstrated. Proper frameworks, methods, tools and metrics to monitor and evaluate the efficiency as well as benefits of e-government investments are lacking. Above all, a clear understanding of the value of e-government, and value for whom, is needed.

**e-Participation, citizen engagement and democratic processes.** In using Information and Communication Technology, elected officials and civil servants
must remain open and accountable in their activities, behavior, and decision-making. At the same time, government must ensure that those individuals and groups that wish to participate in democratic processes have the opportunity and means to do so.

**Mission-oriented goals and performance management.** Many projects do not start with the primary missions of government in mind. Instead, they are often dominated by a technology-driven approach. This is similar to the situation in which a budget is structured and evaluated by the nature of expenses rather than by the public service goals that expenditures support. In both cases management attention is diverted away from the core mission.

**Cyber infrastructures for e-government.** Future e-government technology platforms might consist of a reliable, ubiquitous infrastructure that supports systems and applications assembled out of readily-available, re-usable components. However, realization of this possibility requires research in various domains including whether and how a building block-oriented ICT-industry could be developed, and what types of architectures, building blocks and standards are needed.

**Ontologies and intelligent information and knowledge management.** All governments are currently struggling with huge information overloads, with new and emerging ICT capabilities, and with a shortage of information management skills and human expertise. Ontologies and knowledge management facilities (such as search, retrieval, visualization, text mining, and intelligent reasoning) seem promising to be exploited to achieve information quality and economy, and to support knowledge management processes in e-government settings.

**Governance of public-private-civic sector relationships.** Increasingly, a high number of governmental functions and public services incorporate significant roles for private sector or civic organizations. These roles play out in a variety of relationships from advisory, to collaborative, to contractual, to full partnerships. Adequate principles and frameworks are lacking, which facilitate and set the ground of collaboration in advancing and deploying e-government in regards to share responsibilities and exchanging information among networks of different organizations in ways that generate public value and satisfy public requirements for fairness, accountability and competence.

**Government’s role in the virtual world.** Global electronic markets, virtual organizations, virtual identities, virtual products and services, and Internet-related crime are growing in prominence and importance. In a world that is increasingly non-physical and borderless, government’s roles, responsibilities and limitations are subject to change and are blurring.

**Crossing borders and the need for governance capabilities.** The scope of problems and trends that governments need to cope with vary widely in size, intensity, and complexity. Social networks, gender issues, environmental concerns, political movements, etc. reach beyond local, regional or national borders. It
is unclear, how these phenomena can be steered and governed properly across organizational boundaries, especially through exploiting capabilities available in neighborhood regions and contexts.

**e-Government in the context of socio-demographic change.** Demographic trends with global consequences (such as age distribution, wealth distribution, immigration, and mobility and distribution of workers) are generating pressing issues in both developed and developing countries. Within the European Union, facilitating mobility of citizens and trade across the whole internal European market are strategic aims to foster. These strategic goals as well as the demographic movements and changes require the public sector at the various administrative and political levels to act and react with according public service offers.

**Data privacy and personal identity.** Data privacy and personal identity have become important aspects in the Information Society. On the one hand, the potential of modern ICT could be exploited to take advantage of personal information to improve the performance and quality of government services. On the other hand, privacy and personal data need to be secured and protected in order to prevent misuse and fraud.

### 2.3 Service Delivery in e-Government

The most general concept of service refers to the traditional market and it was rephrased for electronic environment to support the definition of electronic services [173]. Service have been the major source of growth [174] and the introduction of ICT makes heavy this growth.

Literature proposes several definitions for digital services. Focusing on the e-commerce/e-business domain Tiwana and Ramesh refers to service as “Internet-based applications that fulfill service needs by seamlessly bringing together distributed, specialized resources to enable complex, (often real-time) transactions” [201].

An interesting taxonomy and classification of services distinguish physical, digital and pure services [201]. Physical services are services in which the primary product-process supported by the e-service is a physical good, and the service itself is concerned with its assembly, design, aggregation, or delivery, meanwhile in digital services the primary end product delivered by the service is digital information. Finally, about pure services the end product for some services is neither a packaged information product nor a physical artifact. These are pure services in the true sense of the word. With pure service it is not possible to obtain everything before the service delivery [90].

In the next sections definitions, characterization and classification of digital e-government service are presented. A discussion on reference frameworks for
e-government services provision in PA context is also provided.

### 2.3.1 Definitions and Characteristics

Services in e-government play a very important role, they represent the main way to support government in reaching citizens with specific, dynamic, explicit and implicit needs [68]. e-Government distinguishes citizen from traditional web users [137] and it refers to the service as “electronic media including information, communication, interaction and contracting, and transaction” [28]. In other words, digital government services encapsulate Public Administration functionalities and information making them available through digital interfaces.

As mentioned for pure service the main aspect of e-government service is their non-existence before their delivery. In e-government it is not possible to split the service from its delivery process. Services are deeply linked with the service delivery system defined as the set of activities and a complex network of partnership, most of them are integrated across different Public Administrations and agencies. Before of service delivery potential users and (i) methodologies and rules, (ii) physical structure, and (iii) people with knowledge and professional skills able to support the delivery process are observable, but it is not possible observe the services itself. In other words, ICT, information, persons, process, and organization management represent the actors of e-government in the services delivery. As already remarked ICT represents only one components in the service delivery and services are independent from the delivery channel that however has to be considered (see [164] [54] [163] [219] for a discussion on e-government channels).

In such context, the main value of e-government services consists on the ability to force a PA to rethink, reorganize and streamline their delivery before going on-line [175]. It is too easy to build a web site without really improving the traditionally provided service. It is very important to focus on the flow of work in the administrations and design them in line with the citizens needs and administration capabilities. It is also clear that it is not possible to delivery all the services, but it is important to make better selections instead of total uniform making [209].

### 2.3.2 Classifications

A common classification of services is connected to the interaction levels [15] [32]. It is below reported.

1. Informative services are those in which just information is presented.

2. One-way services are those in which the information of the service and more structured documents are available for download.
3. Two-way services support complex workflows with the help of which citizens can submit or receive Public Administration information and documents.

4. Transactional services support the inter- and intra-administration case (in type class of service the payments are supported too).

Another common classification of services in e-government is related to the users: Government-to-Citizen (G2C) services provide full support to citizens, Government-to-Business (G2B) services to firms and Government-to-Government (G2G) services to the same or different administration [65].

**Government-to-Citizen** This category services deal with the relationships between government and citizens. They allow to citizens to access government information and services instantly, conveniently, from everywhere, and, even, using multi-channels solutions.

We can also consider the case of Government-to-Employee (G2E). This area tackles the support for the civil servants themselves with services to manage their carrier, productivity and so on.

**Government-to-Business** It drives e-transactions initiatives between government and the private sector such as e-procurement. It also supports specific tools for paying on-line taxes. The opportunity to conduct on-line transactions with government reduces red tape and simplifies regulatory processes. Therefore it helps businesses to become more competitive.

Close to this area, we can also refer to Government-to-Nonprofit (G2N). This area deals with the special needs of non government organizations such access to specific support for their initiatives, information about funding and related issues, etc.

**Government-to-Government** This kind of services provides government departments or agencies cooperation and communication and internal exchange of information and commodities. As a matter of fact, governments depend on other levels of government to deliver services and allocate responsibilities in the efficiency. The introduction of full interpretability, inside Public Administrations, facilitate the sharing of data, resource and capabilities, enhancing the efficiency, and effectiveness of processes.

### 2.4 e-Government Architecture

The design and development of an ICT architecture that drive e-government services represents an important issues that Public Administration have to solve.
Such architecture should be governed by quality oriented approach.

Under an overall view of Public Administration governance a number of technologies and systems have to be adopted to provide services to the citizens. In most of the case shared platform are built up. For example, an e-government portal requires a common and integrated architecture framework that allows different Public Administrations to share and exchange data, independent of formats, devices and underlying architecture. To do that a clear view on both the technical and information management level is requested to the administrations.

In this section we provide some details about e-government adoption issues focusing on the ICT infrastructure. The role of technologies related to e-government architecture is presented. An overview on Service Oriented Architecture (SOA) and Web Service (WS) is also introduced.

2.4.1 Reference Frameworks

A number of studies have discussed the e-government architecture. Notable among others is the paper of Ebrahim and Irani that propose different views on the general e-government architecture [55]. The framework is structured into four layers connected through two-direction arrows which present the hierarchical level of e-government implementation and portray the logical connection of each relevant layer that allow two-way transmission of data and services. Figure 2.4 shows the architecture framework of e-government which is divided into four layers: access layer, e-government layer, e-business layer and infrastructure layer. They are following presented.

The top level of the framework represents the access layer that illustrates who might use the government services and what are the channels of access. This level support the users in the interaction with the different administration stakeholders via different channels according to specific requests and requirements. Different communication channels are used enabling communication device. For example, web sites accessible from desktop and laptop personal computers, kiosks, mobile phones, digital TV, and call/contact centers.

Throughout access channels, the e-government portal should integrate all government information and services from disparate Public Administrations, which represent the e-government layer. A web portal is an integrated gateway into state government and provides both external and internal government interface with a single point of contact for on-line access to state information and resources. In most of the case one-stop e-government portal has been implemented to support services delivery. Government web-portals are as a key priority for public sector organizations. In particular the use of an integrated web-portal is an important component of e-government infrastructure, since it allows citizens to reduce cumbersome process to a single step. An interesting discussion on how government is
CHAPTER 2. E-GOVERNMENT AND DIGITAL SERVICES

Figure 2.4: Framework of e-Government Architecture (Source [55])
using web portal to enhance electronic service delivery is reported in [72] [179] [199] [73].

In connection to the e-government layer, the e-business layer is emerged to manipulate and integrate government data sources across government bodies and make information and services available to the e-government portal in real-time. At this level both applications and data has to be considered in an integrated fashion. For what concern application it is needed to upgraded to a web-enabled level administrative functionalities beyond PA boundaries and to achieve full communication between all the information systems and their processes. Among others technologies it is important underline the role of Service Oriented Architecture usually implemented via Web Service technology (see [129] and [130] for some discussion on web service e-government architectures). At data level the integration of government database systems, processes and applications play a critical role in this layer since e-government relies to a significant degree on existing basic government data, systems and processes.

Finally in the bottom level of the framework, the ICT infrastructure of e-government should be built to reach out all parts of government and hence, support the e-government operation and provide effective and reliable e-government services. This layer focuses on technologies that should be in place before e-government services can be offered reliably and effectively to the public. Basic technologies, such as Local Area Network (LAN) that allow integration with current hardware resources such as PCs, laptops and mobile phones are considered. As well as they should support the provision of user-friendly and innovative online services involving the transmission of data of various formats such as text, graphics, audio and video. At this level notable is the role of security, it is an ongoing risk associates with most of Information Technology (IT) projects and in term of e-government. See for instance the discussion in [106]. The degree of risk is escalating as the use of public networks increases together with databases that hold citizens profiles and government information.

### 2.4.2 Service Oriented Architecture

Service Oriented Architecture refers to an architectural style that aims to enhance the efficiency, agility and productivity of an enterprise using services to represent solution logic. It revolves around the service-orientation design paradigm and from a technical perspective consist of a combination of technologies, products, supporting infrastructures, etc. that support the creation, execution, and evolution of service-oriented software.

According to the Thomas Erl’s definition “services in service oriented technology exist as a physically independent software programs with distinct design characteristics that support the attainment of the strategic goals associated with
service-oriented computing. Each service is assigned its own distinct functional context and is comprised of a set of capabilities related to this context. Those capabilities suitable for invocation by external consumer program are commonly expressed via a published service contract” [59].

In this sense a service is the fundamental unit of solution logic and service-orientation is the design paradigm on which SOA base its strength. Service-orientation combines design elements of these approaches with new design elements. The application of principles of service design will results in:

- increased consistency in the representation of functionality and data;
- reduced dependencies between units of solution logic;
- increased opportunities to reuse solution logic;
- increased opportunities to combine solution logic into different configurations;
- increased behavioral predictability;
- increased availability and scalability;
- increased awareness of available solution logic.

A design principle can be defined as “a recommended guideline for shaping solution logic with certain goals in mind” [59]. The goals stated previously can be achieved applying patterns when designing software programs. It is important that these principles take a larger, prominent role during development. Eight principle were presented as they are state in [59]. They can be defined as: (i) principles that results in specific service design characteristic, that comprise Standardized Service Contract, Service Reusability, Service Autonomy, Service Statelessness and Service Discoverability; and (ii) principles that regulate the application of the other as Service Loose Coupling, Service Abstraction and Service Composability. In particular for what concern discoverability and composability we following present the SOA scenario.

The process of searching for and finding solution logic within a specified environment is referred as discovery. An architectural resource that can be discovered is considered to have a measure of discoverability. Discovery of services require a consistent mean to communicate information about resources. The goal is that these information must be accurately defined and clearly documented in a consistent format to allow their access and interpretability for the research of available services. Meta information about a resource (a service) comprise the purpose of the resource, their capabilities and the limitation of these capabilities. Discovery
allow enterprise to keep track of the inventory content. When a capability is needed, this is search in the inventory and if there is, it will be used. Instead, if the capability is not in the inventory, it will be built, used, and then cataloged in the inventory. In this way, the discovery function is fundamental, if information about resources is inadequate, the user may lose opportunity to reuse an existing resource and the new build resource may introduce redundancy in the enterprise software asset. Reuse and normalization are undermined and the architecture become bloated and convoluted. The primary motivation behind discovery is the use of service registry to establish a mechanism for on-demand location, retrieval, and interpretation of service metadata.

Reuse is considered a core principle of service orientation, which realization is related to an effective and repeated aggregation of services in compositions. Separation of concerns encourage an approach in which a big problem is decomposed in small problems that are solved by units of solution logic. These units are assembled and coordinated into specific configurations to build a composition that has the aim to solve the original big problem. In this way, solution logic exists as a composable units that can be used and reused to solve new problems. The goal of this principle is to provide the medium through which the ultimate goal of service orientation can be achieved. Enterprises with inventory of highly reusable services have the means to satisfy future business needs. To participate in a composition, services must have a highly efficient execution environment able to manage concurrency, and service contracts need to be flexible to facilitate data exchange with different levels of granularity. As a consequence, hosting runtime environments need to be scalable and reliable as possible.

### 2.4.3 Web Service

SOA is agnostic to any one technology platform. An enterprise is free to pursue its strategic goals leveraging future technology infrastructure but in the current marketplace, the reference implementation used to build SOA solutions is the Web Service one. Although SOA is not related exclusively with web services, this is the main implementation technology we refer to.

The Web Service platform is composed by a continuously growing number of industry standards. These can be partitioned in two generations. The first is related to the original standards like Web Service Description Language (WSDL), XML Schema Definition Language (XSD), Simple Object Access Protocol (SOAP), Universal Description, Discovery, and Integration (UDDI), and WS-I Basic Profile. The second concerns new proposals for message-level security, cross-service transactions, reliable messaging, and so on, and is known as WS-* extensions. These supply rich features for designing services and to implement them.

In the platform, a web service comprises three main things: (i) a service con-
tract that consist in a WSDL description, a XML Schema definition and eventually some WS-policy policies; (ii) a programming logic developed for the service or derived form existing components reengineering for service usage; and (iii) a messages processing logic that is common provided by the runtime environment.

2.5 Frameworks for Service IT Alignment

The importance of fully functional e-government characterized by vertical and horizontal integration is clearly addressed in [115]. From our point of view the truly benefits of e-government can be reached when the system integration across different functions in term of one stop shopping for citizens is implemented. Complex poll of organizations and technological challenges has to be considered, and for this reason interesting framework are proposed in literature. A well known model also supporting this integrated view is Wimmer’s holistic reference framework for e-government [226], its main aim is to provide guideline for successful e-government applications. Later on the capability maturity framework for e-government proposed by Iribarren et al. [98] provide strategic references for Public Administration and discusses the clear relationship between business model and IT in e-government. “IT is supporting business process which are executed by people that develop their activities in an organizational context” [98].

Business Process (BP) model potentiality and capability are relatively unexplored in e-government, it deals with the extension of e-business and e-commerce business model or the role of Internet based model as front-end proxy of e-government Business Process model [103] [55]. As far as we know the most comprehensive discussion on e-government business models is proposed by Janssen and Kuk [102]. They underline the importance of cross-organizational service delivery and propose a framework for studying e-government business model which involves the design and implementation of digital services. They provide a very interesting discussion and argumentation to support the development and improvement of service distributions. They introduce business logic in e-government domain via the following elements.

- **Organizations Participating in the Public Service Network** - Governmental agencies need to collaborate with each other to form a network and effectively coordinate the pools of resources and adjust them using their pooled capabilities.

- **Service Offerings** - Better bundling of the existing and new services is likely to enhance and modify the existing service offerings and to improve the perceived quality of the service offering.
• **Network Coordination** - Mechanisms comprising of managerial and organizational structures have to be in place to facilitate coordination and overcome problems caused by the division of tasks and labor.

• **Business Processes** - In a public service network, which involves the coordination among actors and infrastructures, the focus is on managing the intermediary processes and defining interfaces to enhance inter- and intra-organizational information flows and enabling new levels of emphasis on coordination across agencies via workflow management, project management and supply chain management.

• **Shared Resources** - The role of all kinds of resources, including IT infrastructures, humans, and technologies should aim at supporting the business models, including streamlining through system integration and standardization.

• **Network Capabilities** - The adaptation of IT aims to better reuse of the existing knowledge and expertise dispersed in a public network through building interfaces among management, operation and the design and development of the infrastructures.

One more interesting framework is given by Alter as “work system in which human participants or machines perform work using information, technology and other resources to produce products and services for internal or external customers” [5]. As showed in Figure 2.5 the service work-system characterized in e-government proposes the following elements.

• **Citizens** are the people who use and receive direct benefits from the products and services produced by the PA work system. They may be external citizens who receive the organization’s products and/or services or they may be citizens that work inside the administration.

• **Product and Services** are the combination of physical things, information and services that the work system produces for its citizens.

• **Process and Activities** is the set of work steps or activities that are performed within the work system. These steps may be defined precisely in some situations or may be relatively unstructured in others. In some situation, different participants might perform the same step differently based on differences in their skills, training, and interests.

• **PA Employees** are people who perform the work steps in the Business Process. Some participants may use computers and information technology extensively, whereas others may use little or no technology.
• **Information** is the main input used by the participants to perform their work. Some kind of information may be computerized, but other important information may never be captured on a computer.

• **Technology** is the hardware, software, and other tools and equipment used by the participants while doing their work. The technology considered to be within a work system is dedicated to that system, whereas technical infrastructure is technology shared with other systems.

Further elements that impact on the service work-system in e-government are the following.

• **Environment** includes PA culture and relevant regulations, policies and procedures, competitive issues, organizational history, and technical developments.

• **Strategies** of the PA to which work system should be aligned, although in many situations they may not be articulated clearly.

• **Infrastructure** consists of human, information, and technical resources used by the PA work system but are shared with other work systems and managed and controlled outside the work system.

### 2.6 European and National Scenarios

The development of the Information Society represents one of the main objective in strive the knowledge-based state economy and facilitate the increase of standards of life of the whole society and the individual. European and national governments continuously contribute with different actions, guideline and best practices, as well as with founding, to the development of Information Society. Jan Servaes and Jean-Claude Burgelman propose the most detailed discussion about the development of information system in Europe from 1993 to 2000 [185]. More detailed discussion about the flow of events at European level can be found in the European Commission’s web site. Concerning the Italian scenario the most compressive overview is given in [41] and Baldoni et. al. propose an interesting discussion on the effort for defining and developing a nationwide e-government enterprise [11].

In the following sections we propose a summary on the action lines distinguishing the European and the Italian level.
2.6.1 European Action Plans

The European plans for Information Society grow up under the influence of Japanese and American governments. After the Clinton Gore official lunch of Information Society the Europe quite immediately integrate it into its own plan. The first step was under the label of trans-European network reported in Delors White Paper [149]. This rather neo-Keynesian White Paper was followed by the much more neo-liberal Bangemann-report in 1994 on the basis of an initiative by the Council [151] [150]. It focusses more on the issues of liberalisation of telecommunications and the primacy of the private sector in the development of the information system. In the following years few Green Papers where published by the European Commission. Green Papers are discussion documents intended to stimulate debate and launch a process of consultation, at European level, on a particular topic. A list contains all Green Papers published since 1993 is fully available with the documents themselves in the EU web-site\(^2\). According to the development of Information Society few of them are notable. We report here below. In 1995 the “Green paper on the Innovation” was published. The objective of this Green Paper was to identify the factors - positive or negative - on which innovation in Europe depends, and to formulate proposals to measure the innovation capacity of the EU. One year later the “Green Paper - Living and Working in the Information Society:

\(^2\)http://europa.eu
People First” was published. In 1997 the “Green Paper on the convergence of the telecommunications, media and information technology sectors and the implications for regulation” was introduced. It analyzes the phenomenon of convergence and its impact on the existing legislative framework in the areas of telecommunications, the media and Information Technology. Finally, in 1998 the “Green Paper on public sector information in the information society” was introduced. It underlines the importance of access for European citizens and opportunities for economic growth and employment.

Summing up, the objectives of the initiatives and strategies from 1987 to 1999 where:

- To re-launch the development in the European Union;
- To make aware the European Union about the importance of the Information Society;
- To accelerate the process of finishing the liberalization of telecommunications in particular infrastructures.

The results can be summed in (i) free concurrency on the telecommunications markets and (ii) the first initiatives to introduce the Information Society throughout other policies in education, labor, R&D and regional context.

At this point European scenario was mature enough. At the Lisbon summit in 2000, European Union leaders set out a new strategy on economic, social and environmental development, in order to prepare the EU for the new global challenges at a time of important changes at international level. From Lisbon it come up that Information and Communication Technologies represent both a major challenge and a significant opportunity for job creation. After the initial five years characterized by very modest results, the spring European Council in 2005 decided to re-launch the Lisbon Strategy, stressing the role of growth and employment and promoting a new process of governance. The strategy was reorganized in triennial cycles, strengthening the ownership and sense of responsibility of the different stakeholders, and establishing a clear distinction between reforms to be undertaken by Member States and those the EU should carry out in order to take on a leading role at global level.

Under the eEurope 2002 and eEurope 2005 initiatives the European Commission wants to reach the following aims:

- To re-launch the economic growth of the European Union;
- To accelerate the process of review the telecom regulatory framework;
- To accelerate the use of Internet;
• To consolidate the R&D activities on Information Society technologies.

In this case the results can be observed as (i) the new regulatory framework on telecommunications, (ii) a specific set of initiatives for the development of the Information Society such as e-leaning, e-inclusion, e-content, e-TEN and e-government, (iii) the first set of national plans for the development of the broadband, and (iv) the consolidation of the IS technology plans on the 6th R&D Framework Programme.

At this point the Strategy i2010 represents the policy objectives of EU Information Society. i2010 is specifically identified in context of the EU’s Lisbon strategy, a strategy which aims to make Europe, by 2010, the most competitive and the most dynamic knowledge-based economy in the world. i2010 aims to promote creation of more unified information space, revising regulations and enhancing location-specific content. e-Inclusion, a key objective with the i2010, is intended to promote an inclusive society by accommodating diverse interests of aged, disabled and minority sections of society.

Under the Strategy i2010 the following aims were pursued:

• To re-launch the economic growth of the European Union;
• To accelerate the process of review the telecom regulatory framework;
• To push the regulatory convergence between telecom and audiovisual sector;
• To accelerate the development of the broadband infrastructures;
• To consolidate the R&D activities on Information Society Technologies.

Results coming up are (i) the specific strategies on e-accessibility, e-inclusion and e-government, (ii) the definition of the new IS technology plans on the 7th FP and (iii) a new review of the regulatory framework on telecommunications.

### 2.6.2 Italian Action Plan

Before 1999, the scenario of the ICT in Italian Public Administration was quite heterogeneous. Some central PA were excellent but they were almost isolated. Outsourcing was the main driver of PA innovation and it provided explosion of cost and lack of efficiency. The first effort towards the improvement of digital administration was the development of “Rete Unitaria della Pubblica amministrazione” (RUPA) [128], the national cooperative network. It provided security and basic interoperability services.
According to the European Commission action plans e-Europe 2002, e-Europe 2005 and i-2010 the national government has recognized the role of Information Society. National action plans were developed together with guidelines to promote the diffusion of new technologies in Italy. It supported the efficiency and simplification of the relationship between Public Administration and citizens. The reform on the diffusion of new technology came together with the national political reform, that attributed new possibilities for local autonomy. Decision on ICT choices as well as at different organizational processes were delegated at the local administration and only part of them were supported by central government. Such context strove the heterogeneity that already characterized the Italian administrations.

Under the e-government national plan in April 2002 the national government used the income coming from the sale of UMTS licences to co-fund a national tender for the e-government in practice. The result was 134 projects coordinated by local administrations and regions. Such projects referred to front-office service and back-office infrastructures and they involved different areas such as healthcare, employment, register office, tax office, etc. Heterogeneity was more and more stimulated by the intensive introduction of such projects.

Moreover, starting from 2003 the CNIPA\(^3\) (National Center for IT in Public Administration) started the coordination of a nationwide bottom-up consensus operation, from basic telecommunication services to advanced applications cooperation. Such initiative involved 300 representatives of central and local Public Administrations, universities and research centers, and Italian companies. The outcome has been a set of technical and organizational recommendation on the SPC (Public Internet-working System). It included SPCoop for the application cooperation among PAs\(^10\).

In February 2005 the Italian Government issued a Law Decree, namely the Digital Administration Code (CAD) (Law Decree n. 82/05) with the following aims: (i) the rights of citizens and enterprisers on Public Administration (ii) citizens and enterprisers must be placed at center of PAs services (iii) digital signatures and legal validity; (iii) contracts, payments and accounting deeds; and (iv) development, acquisition and reuse of software in PAs. CAD defined also the role of SPCoop and SPC, they were exclusive way of cooperation among administrations and all the administrations have review their way to work in line with organizational and management aspects evinced by SPCoop and SPC. Under the SPCoop umbrella some regional projects on e-government have been launched. These process proved the bottom-up approach on the applicative cooperation and the development of Information Society in Italy. The biggest one was Interoper-

\(^3\)www.cnipa.gov.it
ability and Applicative Cooperation among Regions (ICAR). It started in June 2006 with 17 partners including 16 of 19 Italian Regions.

Evidently, the main driver of the national government contemplated a deep reorganization of back-offices. the concomitant front-office changes, a complete re-think from scratch of the whole system and philosophy of service design, production and delivery.

The most recent action of the Italian government named “Piano e-Gov 2012” (2012 e-Gov Plan) was presented in January 2009 by the current minister for Public Administration and Innovation Prof. Renato Brunetta. Its main aim is to modernize governments and increase competitiveness of the regions, for instance, reducing by 25 percent the administrative burdens, by 2012. The principles of the plan are aimed to government innovation, to spread on-line services, to increase efficiency, accessibility and transparency, in order to bring it closer to the citizens and business needs. The plan hopes to achieve the goals set out by the European Union member states in Lisbon in 2007. The E-Gov Plan 2012 consists of 80 projects, structured around 4 priority areas namely: (i) sectoral, referring to central government and universities; (ii) local, covering either the Regions and their capitals; (iii) structural, including infrastructure projects, e.g. projects for reducing the digital divide or for improving the accessibility of government services; and (iv) international, which is aimed to maintain Italy’s major involvement in the European-scale networks focused on infrastructures, innovation and best practice dissemination.

4www.progettoicar.it
5www.e2012.gov.it
Quality in Different Application Domains

A lot of interesting works has been done in the wide area of quality according to different application domains. In this chapter we provide a literature review focusing on the most influencing papers for our study. We introduce the main works related to customer satisfaction as well as the role of quality in organizations. We also discuss on process quality as another hot topic particularly for what concern process optimization and related efficiency measure. Finally we report the work done on quality of technology infrastructure with a particular focus on Service Oriented Architecture.

3.1 Customer/Citizen Satisfaction

Customer satisfaction is ambiguous and abstract concept, it can vary from person to person and service to service. The state of satisfaction depends on different psychological and physical variables which are correlate with other options the citizen may have and other service against which the citizen can compare the PA’s services.

Citizen satisfaction is usually measured via a survey with a set of statements organized as a scale. The citizens are asked to evaluate each statement in term of their perception and expectation of performance of the Public Administration being measured.

Different methodologies and theories for understanding customer needs and measure customer satisfaction has been developed. Customer Satisfaction Index (CSI) is one of such methodologies. The basic structure of the CSI model has been developed over a number of years and is based upon well established theories and
approaches to consumer behaviour, customer satisfaction and product and service quality [66]. The structure of the CSI is continually undergoing review and subject to modifications. Although the core of the model is mostly standard, there are some variations between the Swedish Customer Satisfaction Barometer (SCSB), the American Customer Satisfaction Index (ACSI), the European Customer Satisfaction Index (ECSI), the Norwegian Customer Satisfaction Index (NCSB), the Italian Customer Satisfaction Index (ICSI) and other indices. In particular the American Customer Satisfaction Index is an economic indicator that measures the satisfaction of consumers across the United States economy [60]. The ACSI model is showed in Figure 3.1 and it is produced by the National quality Research Center at the University of Michigan. The ACSI interviews about 80,000 Americans annually and asks about their satisfaction with the goods and services they have consumed. Respondents are screened to cover a wide range of business-to-consumer products and services, including durable goods, services, non-durable goods, local government services, federal government agencies, and so forth. Results from data collection and analysis are released each quarter. ACSI data is common used by academic researchers, corporations and government agencies, market analysts and investors, industry trade associations, and consumers.

![Figure 3.1: ACSI Model.](image)

Other approaches support the measurement of citizen satisfaction. The Kano model showed in Figure 3.2 is a theory of product development and customer satisfaction developed in the 1980s by Professor Noriaki Kano that classifies customer preferences into five categories: attractive, one-dimensional, must-be, indifferent and reverse [146]. The Kano model offers some insight into the product
attributes which are perceived to be important to customers. Kano also produced a methodology for mapping consumer responses to questionnaires onto his model.

Another widely used framework for measuring customer satisfaction is SERVQUAL [159]. The Figure 3.3 introduces the graphical representation of SERVQUAL model. It was developed in the 1980s by Zeithaml, Parasuraman and Berry. The method is also known as the RATER model, because it prescribes measuring satisfaction in these five dimensions.

- **Reliability** - A company’s ability to perform the promised service dependably and accurately.
- **Assurance** - The knowledge, competence and courtesy of employees and their ability to convey trust and confidence.
- **Tangibles** - Physical facilities, equipment and appearances that impress the customer.
- **Empathy** - The level of caring, individualized attention, access, communication and understanding that the customer perceives.
- **Responsiveness** - The willingness displayed to help clients and provide prompt service.
Figure 3.3: SERVQUAL Model (Source [159]).
3.2 Organizations Quality

In the area of traditional public service quality interesting approaches are given by sophisticated performance measurement frameworks as the result of effective management of various parameters within the organization. In the following we mainly refer to the Balanced Scorecard (BSC) approach, CAF model, Six Sigma and Baldrige Criteria. In particular in Table 3.1 it showed a comparison between BSC and CAF.

The Balanced Scorecard is a strategic planning and management system that is used extensively in business and industry, government, and non-profit organizations worldwide to align business activities to the vision and strategy of the organization, improve internal and external communications and monitor organization performance against strategic goals [107]. The Balanced Scorecard suggests that we view the organization from four perspectives, and to develop metrics, collect data and analyze it refers to each of the perspectives that are reported below and showed in Figure 3.4.

- The learning and growth perspective includes employee training and corporate cultural attitudes related to both individual and corporate self-improvement.

- The business process perspective refers to internal business processes. Metrics based on this perspective allow the managers to know how well their business is running, and whether its products and services conform to customer requirements (the mission).

- The customer perspective shows the role of customer in the overall quality evaluation.

- The financial perspective encourages the identification of a few relevant high-level financial measures.

One of the main drawbacks of BSC is that each organization has to properly find out their metrics that accurately capture progress toward goal attainment, but organizations often struggle to identify such aspects.

An interesting evolution of traditional BSC is proposed by Yu, she discusses about a value-centric e-government service framework based on the business model perspective for guiding and ensuring successful development, management, and delivery of e-government systems and services [239]. It adopts and adapts the Balanced Scorecard to include 4 new dimensional views to plan and evaluate e-government services, namely, public beneficiaries, government internal organizations and processes, government service chain, as well as society and national environments.
CAF is a common European quality framework that can be used across the public sector as a tool for organizational self-assessment. It has been jointly developed under the aegis of the Innovative Public Services Group, an informal working group of national experts in order to promote exchanges and cooperation where it concerned innovative ways of modernizing government and public service delivery in EU Member States.

Six Sigma is a methodology to manage process variations that cause defects, defined as unacceptable deviation from the mean or target and to systematically work towards managing variation to eliminate those defects [157]. The objective of Six Sigma is to deliver high performance, reliability, and value to the end customer.

The Baldrige Criteria for performance excellence provides a systems perspective for understanding performance management [117].

3.3 Process Quality

An important area of quality refer to Business Process Management (BPM). In the following we will consider it in-depth. Anyway to have a complete view on quality in this section we give a brief overview on the topic and we refer to some of the most contribution in such area.

Business Process Management encompasses methods, techniques, and tools
that allow organizing, executing, and measuring the processes of an organization. An important aspect in the Business Process Management is the performance of business processes. Performance requirements on business processes are specified as Key Performance Indicators (KPIs) or performance indicators with target values which are to be achieved in a certain analysis period. They are measure of performance which are commonly used to help an organization define and evaluate how successful it is, typically in terms of making progress towards its long-term organizational goals. KPIs may be monitored using Business Intelligence techniques to assess the present state of the business and to assist in prescribing a course of action. The act of monitoring KPIs in real-time is known as business activity monitoring. KPIs are frequently used to value difficult to measure such as the benefits of leadership development, engagement, service, and satisfaction. KPIs are typically tied to an organization’s strategy using concepts or techniques such as the Balanced Scorecard. Typical KPIs are below reported.

- Cycle time is the total time from the beginning to the end of your process, as defined by you and your customer. Cycle time includes process time, during which a unit is acted upon to bring it closer to an output, and delay time, during which a unit of work is spent waiting to take the next action.

- The amount of time, defined by the supplier, that is required to meet a customer request or demand.

- The capability of a process is the ability to perform its specified purpose based on tested, qualified or historical performance, to achieve measurable results that satisfy established requirements or specifications.
There are also domain dependent business process metrics such as:

- % of stakeholders satisfied with IT quality;
- % of sick days (illness rate);
- Average number of sick days per employee;
- Average training costs per employee;
- Average number of training hours per employee;
- % of successful software upgrades;
- Average time to restore off-site backup;
- Deviation of planned budget;
- % of financial reports issued on time;
- % of backup operations that are successful;
- Average numbers of training hours per employee;
- Average training costs per employee;
- % of time of employees available for improvement activities.

### 3.4 Web Portal Quality

In the area of web portal different approaches can be consider to evaluate the overall quality of the portal mainly in the business to consumer domain. An interesting literature review on web portal quality can be find in [240]. In the following we refer to the most interesting works related to quality in e-commerce, e-government portal and we stress the role of usability as one of the main aspect in web portal quality.

#### 3.4.1 e-Commerce Domain

Most of the studies of web portal quality are exploratory or conceptual having no empirical validation. For instance, Liu and Arnett find out four factors that are critical to web portal success in electronic commerce. They are: (i) information and service quality, (ii) system use, (iii) playfulness, and (iv) system design quality [121]. Webb and Webb in 2001 developed a conceptual model focusing
on the factors affecting consumer perceptions of Business To Citizen (B2C) web sites [212]. The underlying premise is that two major quality constructs, one focused on information (accessibility, contextual, representational and intrinsic) and the other focused on service (reliability, responsibility, accuracy, empathy, tangibility), determine B2C web portal quality. Yang et. al. present five dimensions web portal quality model composed by usefulness of the content, adequacy of the information, usability, accessibility, privacy/security and interaction [235]. The approach of Lin and Wu provides general hints on the development of a portal in order to keep people continuing to visit the portal site [119]. The aim of their work is to explore users’ intention and behavior of the portal defining the on-line service quality as the basis of discrepancies between customer expectation and perception of the service being offered. Underline dimensions are reported below.

- **Information content** - Ease to understand the text, graphic, article, and proper information.

- **Customization** - Personalization service, understand the customers’ need, provide the information to fit with the users, and ease to use.

- **Reliability and response** - Problem solving, on-line service, display transmission service correctly, and in time.

- **Security** - Security of transmission, privacy protection.

Moraga et al. make a proposal of a portal quality model composed by five dimensions: tangibles, reliability, responsiveness, assurance, empathy and data quality [138].

A lot of effort was also devoted to develop and testing suitable instruments to measure quality of web site. Following we cite some of them. SITEQUAL provides guidelines and an instrument to measure users feedbacks on the overall quality of B2C electronic commerce Web portal over time [213]. The E-S-QUAL model represents a multiple-item scale measuring the service quality delivered by on-line shop web portal [160]. WEBQUAL is an instrument to assess usability of information, and service interaction quality of web sites, particularly those offering e-commerce facilities [123] [12]. Finally the eTailQ model of web site quality consists of four major factors: (i) web site design, (ii) privacy and security, (iii) fulfillment and reliability, and (iv) customer service [228].

### 3.4.2 e-Government Domain

Focusing on the e-government domain. The introduction of a proper web portal represents a fundamental part of the e-government services delivery. An interesting review of the literature about service quality delivery in Public Administration
can be find in [84]. Following we present some interesting research contributions in this area.

Sukasame developed a conceptual framework to elicit the factors affecting the e-service provided on the web portal of Thailand’s government [195]. In particular, it presents a conceptual framework and some factors (content, linkage, reliability, ease of use and self-service) affecting the digital services provided on the web portal of the Thailand’s government.

The Western Norway Research Institute initiated a project to develop a set of quality criteria to evaluate public web sites in Norway [101].

The g-quality method proposes an extension of Nielsen’s heuristic traditional evaluating method, applied to the information, services and citizens’ participation categories [74]. In this case broad accessibility, interoperability, security and privacy, information truth and precision, service agility and transparency are considered. The g-quality method was instrumental as an objective evaluation form. It was applied to 127 Brazilian e-gov sites.

Norwegian Approach is another interesting contribution on evaluation of public web site in Norway as a view of benchmarking as a method to ensure public agencies commitment to standard [101].

Scott discussed another contribution in the area of e-government web portal and proposes five characteristics reported below [184].

- Transparency - the site makes it easy for users to monitor official public records and to communicate with city official.
- Transactions - the site allows users to complete a wide range of on-line transactions.
- Connectivity - the site connects users with other individuals or organizations that contribute to the local civic interest.
- Personalization - the site personalizes Web content based on analysis of user preferences and behavior.
- Usability - the site is reliable and easy to use.

Finally, the main interesting work on quality in e-government was made in the FIT project[1]. It proposes an interesting approach to monitoring the front office and back office quality taking into account three different perspectives [194]. The first perspective, the subjective one, represents citizens/users of e-government services. The second, the substitute perspective, represents the service provider,

---

1 www.fit-project.org
i.e. the public organization, and finally the third perspective is the objective perspective, represented by expert groups in the e-government domain and also by operational data of the e-government portal. The project introduces web based questionnaires, a web log filtering tool and a process log filtering tool. It supports quality measurement focusing on the proposed perspectives. It focus on service (interaction, service reliability and support mechanisms), content (usability and quality of information) and system parameters (security, front office performance indicators and back office performance indicators). In particular, concerning back office performance indicators the FIT project refers to the percentage of process steps that were adapted, process execution cost, process execution time and percentage of workflow instances completed successfully.

### 3.4.3 Web Site Usability

It is easily understandable that in the context of web site usability represent one of the critical aspect to assess quality \[143\] \[156\] \[77\]. Researchers have identified various factors, primarily content layout and classification, web site structure, user interface, web site appearance and visual design, intuitiveness, readability/comprehension/clarity, search facilities, and ease of navigation. One critical requisite is technical adequacy, which concerns web site technical features; e.g., capacities of systems, networking, hardware and software and system integrity. Also, for e-government services usability represents one of the main quality parameters \[34\] particularly for what concern people with disabilities \[97\] and older adults \[17\]. It represents the ease of benefiting from the service and from the information it provides. Moreover, it is the starting point for the definition of quality in e-government focusing on the users.

### 3.5 Web Service Quality

The Service Oriented Architecture approach is a very popular choice today for the implementation of distributed systems. The use of Service Oriented Architecture or more specifically the Web Services technology is an important architecture decision. A software engineer should understand how different quality attributes for a system are impacted by that decision \[172\]. In such area there is a lot of work particularly for a quality oriented service discovery and composition. In the following we mention some of them. Open issues about quality are mainly investigate on in Service Level Agreements that are used to contract the level of service quality between service providers and users. More detailed on Service Oriented Architecture and Web Service will following in the thesis.
The wider approach in this area is given by the World Wide Web Consortium (W3C) [116]. Their proposal about quality of Web Service reports quality requirements and approaches supporting the Web Service quality. More specific approaches focus on quality aspect such as usability [126] [14], trust [71] [186] and security [233].

With a particular focus on service discovery we could refer to the papers discussed below. In 2003 Shuping Ran argues that quality of services is one of the main contribution toward Web Service adoption. The paper proposes a new Web Services discovery model in which functional and non-functional requirements are taken into account [168]. An interesting discussion on such parameters and on the way how to measure them is introduced. The proposed model should give Web Services consumers some confidence about the quality of service of the discovered Web Services. Farkas and Charaf introduce a software architecture to provide Web Services with high quality [61]. In particular they implement a broker for service discovery to reflect quality parameters stored in UDDI. Maximilien and Singh discuss the lack of a description of non-functional attributes needed for the discovery of Web Services [127]. They propose an ontology-based framework to describe quality in order to improve the stakeholders’ interaction. An expanded service model supporting quality of service is proposed in [166]. The authors discuss a model ablest to supports service publishing and selection based on quality of service. In the model, quality of service information is collected from different roles like service provider, service user and service monitor, then saved as publish quality of service, experience quality of service and monitor quality of service. quality of service measure is based on weighted the different quality of service information above. So it takes account of all kinds of aspects that affect the quality of service driven service selection. A more recent work on the area of service discovery has been published by Yan an Piao [234]. The paper presents an approach to achieving quality of service-based Web Services discovery. Data structures are proposed for both service providers and service consumers to describe non-functional information about services. Algorithms are developed for matching and ranking services according to consumers’ non-functional requirements.

In a cross-organizational setting, it is important for service customers obtain, monitor and enforce quality of service guarantees by service providers, usually expressed in the form of Service Level Agreements. Since the supervision and management of SLAs and the provisioning of corresponding systems should be automated for economic reasons, formal languages to define SLA are needed. Ludwig et al. propose a novel framework for specifying and monitoring Service Level Agreements for Web Services [124] [108]. More recently the first workshop on non-functional properties and Service Level Agreements in Service Oriented Architecture was held [158]. These issues were felt to be highly relevant.
due to the increasing popularity of Service Oriented Architecture and the fact that whilst the foundations of Service Oriented Architecture functionality are now well understood, non-functional properties are not.

According to the more general area of middleware application based on Service Oriented Architecture interesting papers have been published. Nahrstedt et al. discuss quality middleware information able to support quality-based applications like streaming and e-business In [140]. This work presents key aspects about service quality introducing application and process quality information at a low abstraction level. Tsetsekas et al. propose a middleware that drives service presentation to the users. It allows the description and the selection of quality parameters and the resources that support the quality [203]. O’Brien et al. discuss about how the different quality attributes of a system can be positively or negatively affected by the use of such service oriented technology [148]. It describes the factors related to each attribute, as well as possible trade-offs and existing efforts to achieve that quality.

Zeng et al. present a middleware platform which addresses the issue of selecting Web Services for the purpose of their composition in a way that maximizes user satisfaction expressed as utility functions over quality of service attributes, while satisfying the constraints set by the user and by the structure of the composite service [243]. Zeng et al. advocate that the selection of component services should be carried out during the execution of a composite service, rather than at design-time [242]. In addition, this selection should consider multiple criteria (e.g., price, duration, reliability), and it should take into account global constraints and preferences set by the user (e.g., budget constraints).

An interesting recent work examine the role of the Balanced Scorecard methodology in Web Services quality. In such case, a Balanced scorecard framework is developed for Web Services quality by identifying critical success factors that make up the business objectives, measures, targets, and initiatives [169]. It is the most general work on quality of Web Service.

### 3.6 Network Quality

In the field of computer networking and other packet-switched telecommunication networks, the traffic engineering term quality of service refers to resource reservation control mechanisms rather than the achieved service quality. Quality of service is the ability to provide different priority to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow. For example, a required bit rate, delay, jitter, packet dropping probability and/or bit error rate may be guaranteed. Quality of service guarantees are important if the network capacity is insufficient, especially for real-time streaming multimedia.
applications such as voice over Internet Protocol (IP), on-line games and IP-TV, since these often require fixed bit rate and are delay sensitive, and in networks where the capacity is a limited resource, for example in cellular data communication. In the absence of network congestion, quality of service mechanisms are not required.

We recognize such areas of study very important to support the delivery of service and their use by the citizens. However, according to our approach the focus of our work is more applicative and for such reason we skip a wide and in-depth investigation in the area of computer networking quality. A lot of interesting papers and research contribution can be easily find on this topic in the literature.
### Table 3.1: BSC vs CAF.

<table>
<thead>
<tr>
<th></th>
<th>BSC</th>
<th>CAF</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Monitoring and measurement of performance-creation of added value</td>
<td>Total quality Management for a culture of excellence</td>
</tr>
<tr>
<td><strong>Goals</strong></td>
<td>(i) Improvement of strategic and financial performance</td>
<td>(i) Continuous improvement through the encouragement of</td>
</tr>
<tr>
<td></td>
<td>(ii) translating the organisation’s strategy into practical activities,</td>
<td>the introduction of best practices</td>
</tr>
<tr>
<td></td>
<td>communication and monitoring</td>
<td>(ii) Indemnification of strengths and areas of improvement</td>
</tr>
<tr>
<td></td>
<td>(iii) Focus on success drivers</td>
<td>(iii) Action plan</td>
</tr>
<tr>
<td><strong>Results</strong></td>
<td>Set-up of strategic objectives with a logical cause-and effect</td>
<td>Qualitative assessment of managerial practices, process and results</td>
</tr>
<tr>
<td></td>
<td>relationship with indicators coming from 4 perspectives</td>
<td>according to the 9 criteria of the model expressed as strengths and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>areas of improvement.</td>
</tr>
<tr>
<td><strong>Approach</strong></td>
<td>(i) Strategy oriented</td>
<td>(i) Process and stakeholders oriented</td>
</tr>
<tr>
<td></td>
<td>(ii) Contains presumptions</td>
<td>(ii) Based on evidences</td>
</tr>
<tr>
<td></td>
<td>(iii) Macro-level examination by management</td>
<td>(iii) Detailed diagnosis trough self-assessment by employees</td>
</tr>
<tr>
<td></td>
<td>(iv) Future oriented</td>
<td>(iv) Present oriented: snapshot of the current situation</td>
</tr>
<tr>
<td></td>
<td>(v) Specifically tailored for each organization:</td>
<td>(v) Same areas of attention for each organization</td>
</tr>
<tr>
<td></td>
<td>objectives and measurement</td>
<td></td>
</tr>
<tr>
<td>**Focus and</td>
<td>Focused on important issues selected by the management</td>
<td>(i) Comprehensive assessment</td>
</tr>
<tr>
<td>Priorities</td>
<td></td>
<td>(ii) No priorisation</td>
</tr>
<tr>
<td>**Present and</td>
<td>This is what we will look like in the future</td>
<td></td>
</tr>
<tr>
<td>Future**</td>
<td></td>
<td>This is what we look like today</td>
</tr>
</tbody>
</table>
Part II

A Systematic View of Quality in e-Government
Chapter 4

A Quality Model for e-Government

This part of the thesis was published in [R1] [C10] [C8] [C7] [C6] [C3].

4.1 Introduction

Quality of e-government services is more and more emerging as a key issue within Public Administrations. Ensuring a proper quality level is mandatory to satisfy citizens and firms’ needs and to accept the use of ICT in our life. Introducing quality oriented strategies in Public Administration the gap between implementation and use of e-government service can be resolved. Quality is a complex topic and it presents different views according to the complexity of e-government [28]. Notable is the well know ISO’s quality definition where the quality is intended as “all the features and characteristics of a product or service that affect its ability to satisfy stated or implied needs” [43]. Rephrasing such definition in the following of the thesis we refer to quality as “all the features of digital services in Public Administrations that influence their capability to satisfy declared or implied citizens and firms’ needs”.

According to the complexity of quality and e-government an important part of our work has been the definition of quality parameters in line with the feature of digital service in PAs. To do that we have reviewed the e-government domain and, in this chapter, we report a comprehensive quality model. The model refers to citizens satisfaction in term of subjective parameters, considers technology infrastructure quality and implements process related quality requirements.
4.2 Overview of Quality in e-Government

Literature proposes interesting works related to quality in PA domain considering different aspects of the service. All of these work influence somehow the development of this thesis, but no one proposes a complete view of digital services quality requirements in e-government. Halaris et al. present the most comprehensive literature review on e-government quality and summed up a stack with quality assessment layers as showed in Figure 4.1 and following introduced [84].

- **Customer satisfaction** address the quality parameters perceived by the customers against their expectations.
- **Site quality** considers the web site usability and interface quality characteristics.
- **Technical performance** takes into account technical aspects of web site.
- **Process performance** introduces quality aspects typically related to traditional government services.

![Figure 4.1: Layers of Quality Assessment (Source [84]).](image)

Halaris et al. discussion is an interesting enrichment on the e-government quality literature. We consider the proposed layers as a good starting point based on the analysis of the authors on the literature. We review such layers as showed in Figure 4.2.

Our quality view considers the importance of customer satisfaction, organization, process and technology related quality as well as the role of substitutive
services whose quality impacts on the services delivery. Following we present the different levels of quality according to our study.

- **Citizens satisfaction** address the overall level of quality perceived by the users against their expectations. We just rename “customer satisfaction” level making explicit that citizens are the final consumer of the service in e-government.

- **Technology Infrastructure Quality** address the quality of front-end, back-end and network infrastructure requested for the service delivery. Our view on front-end implements both “site quality” and “technical performance” layers discussed in the Halaris et al., back-end components refer to the service implementation usually via Service Oriented Architecture and network infrastructure enables service use.

- **Process Quality** is based on the “process performance” discussed by Halaris et al. According to our view such level points out the role of new government organizations ready to take fully advantage of the e-government process characteristics enriching back-office process performance.

- **Organizational Quality** introduces the quality of PA in term of financial and human resources allocation as well as administration capability to innovate. This view has been implicitly implemented by process performance layer on Halaris et al. proposal.

- **Substitutive Service Quality** introduces quality view on substitutive service. They play an important role in e-business where the market governs services co-existence. We recognize substitutive service as marginal
in e-government where the delivery policies govern service co-existence. However, it is needed to introduce a complete view on e-government quality where quality changes in line with the quality of service with the same functionalities offered by other Public Administrations.

In our study, we focus on the first three levels of the pyramid. We believe that with this focus, we can contribute to improve the current maturity on the e-government and solve the gap between implementation and use.

4.3 e-Government Subjective Parameters

e-Government service quality is deeply linked to citizens that have to be considered together with their experiences to assess carefully quality. In this case, a subjective evaluation is needed. In this section, we refer to attraction, trust, and usability. According to our investigation, they are the most important e-government subjective parameters which evaluation passes from subjective rating given by citizens.

- **Attractiveness** represents the incentive for the citizens to use online services rather than traditional ones.

- **Trust** represents the level of confidence of the citizens in the service use. Focusing on trust, we consider service trustworthiness. Trustworthiness is a moral value considered to be a virtue. A trustworthy Public Administration is an organization in whom we can place our trust and rest assured that the trust will not be betrayed. A Public Administration proves his trustworthiness by fulfilling its tasks. This parameter is implicitly governed by privacy and communication security. They are measured focusing on identity-related risks: (i) interception or revelation of secret authentication information, (ii) retention of secret authentication information in untrusted terminal, (iii) use of compromised credential, and (iv) use of credential after substantive change in circumstances.

- **Usability** refers to “a measure of the effectiveness, efficiency and satisfaction with which specified users can achieve specified goals in a particular environment” as provided by ISO [99]. Generally speaking, e-government usability represents the user’s ease of benefiting from the service informative resources, functionalities and meta-information. It is important to say that the degree of interaction – between the service and the user – and the involved technologies must be considered in determining the values of usability parameters. Usability in e-government services is specially important to
the elderly and disabled [18]. Citizens are more likely to have favorable attitudes about e-government if their experience with digital services in general has been positive [176]. Usability can be detailed as following.

– **Understandability** is the measure of how readily the citizens perceive the information significance and the way the service works.

– **Learnability** measures the ability of the service to support the citizens in learning how to use it.

– **Compliance** represents the level of the service necessary to present a harmonious environment with the traditional government in which its behavior in general and its different tasks are predictable.

– **Attractiveness** is the measure of the digital service ability to attract citizens rather than the tradicional service.

### 4.4 Technology Related Parameters

Focusing on the ICT enabling e-government services we introduce a three levels classification. The first level focuses on (i) presentation parameters that analyze the front-end of services with respect to the end users’ needs and specific domain dependent requirements of web portal implementation. Then, the second level discusses (ii) behavioral parameters that describe the back-end implementation of services, and data related parameters. Finally the third level presents (iii) network parameters related to the basic infrastructures enabling digital services.

About front-end related aspects we take into account the following parameters.

- **Cost** measures the average amount of money involved in a complete service transaction, capturing the economic condition of the service use. It summarizes cost related to the service delivery such as execution price and extra money requested for electronic transaction.

- **Accessibility** refers to the practice of making websites usable by people of all abilities and disabilities. When sites are correctly designed, developed and edited, all users can have equal access to information and functionality.

- **Adaptability** evaluates the service ability to change (or being changed) and make itself suitable for a new context. It is measured focusing on a mapping among service functionalities and the following items (i) user profile (ii) users context, and (iii) device used to access the service.

- **Popularity** is the service’s capability of being widely admired or accepted. It is measured considers the amount of population interested in the service
as well as the frequency of utilization with respect to a period of time and to the number of e-government users.

- **Internationalization** refers to the languages used for service delivery. It is partially important for informative service.

- **Originality** focuses on the service innovation level. It is measured considering the capabilities of (i) actively absorb technology, especially information technology, and strive to build e-government, (ii) adapt to the market economy of public management methods, focus on planning, use of the macro, indirect management methods, and enhance coordination, and (iii) implement innovative approach to introduce full theoretical research.

- **Contents** measures the capabilities of Public Administrations to provide useful and proper information. It is measured focusing on the following indicators: (i) information appropriateness, (ii) information updating, and (iii) information completeness.

- **Legality** refers to the specification of laws and norms that regulate service delivery and use.

- **Domain Security** measures the control level of the service. It is measured focusing on (i) the service and information adherence to law and regulation in relation to the communication standard, and (ii) adherence to law and regulation during the service delivery.

- **Promoting e-Democracy** estimates the impact of a service on the society. This supplies a quantitative value to show how a service can promote digital citizenship, namely, the set of practices of ICT used by citizens to take part in political choices at any level. The presence of forums, FAQs, mailing lists, etc. – related to the service – has a good impact on the value of this parameter.

- **Completeness** of a service represents the interaction levels. It is evaluated starting from the different ways to interact with the service according to the e-government service classification based on the interaction.

In the behavioral group we introduce the following items.

- **Interoperability** represents the amount of cooperative work among consumer applications, software agents and services in different development environments that implement and deploy procedures. It is measured by the degree of supported technical interoperability.
• **Applicative Security** represents the security level of web services introducing authentication and authorization policies and procedures sometime federated.

• **Integrity**, about data and transactions, is another important element. It measures the service ability to prevent unauthorized access to - or modification of - computer programs or data. On one hand it is based on the ACID properties: atomicity, consistency, isolation and durability [78]. On the other hand it is measured based on the standard such as business process management and transactions such as BPEL4WS, WS-Coordination and WS-Transaction.

• **Robustness/Flexibility** measures the service ability to work correctly even when not valid, incomplete or conflicting inputs occurs. It is affected by service stability in terms of its interface and/or implementation. It is measured focusing on (i) explanation related error or exception raised by the service execution, (ii) explanation related to frequently errors or exceptions, and (iii) automatic service recovery.

Related to infrastructure parameters we analyze the following items.

• **Availability** considers how the service is available when a client attempts to use it. Equation [4.1] represents capability to reply immediately to the users requests where $T_A$ is the total amount of time where the service is available during the interval $t_1$, $t_2$ and it is computed through service site measures. If $T_A = t_2 - t_1$, then we have the maximal availability.

\[
ava = \frac{T_A \times 100}{t_2 - t_1} \tag{4.1}
\]

• **Performance** represents how fast a service request can be completed. It measures the speed in completing tasks using service response time, latency and execution time. The service response time (rt) in Equation (4.2) measures the delay (at client site) between sending a request and receiving response.

\[
rt = t_{\text{clientReceive}} - t_{\text{clientSend}} \tag{4.2}
\]

The service execution time (et) in Equation (4.3) measures the time needed to process service instructions. In particular, it measures the delay between request reception and response forwarding by the service. The measure is executed at service site.
The service latency ($lat$) in Equation 4.4 measures the delay between sending a request and receiving a result at client site, without taking into consideration the execution time of the service.

$$lat = rt - et$$  \hspace{1cm} (4.4)

- **Reliability** is the ability of a service to perform its required functions under stated conditions for a specified period of time. In Equation 4.5, $F$ indicates the number of executions that the service has not been successfully completed within the interval $t_1$, $t_2$.

$$rel = \frac{F}{t_2 - t_1}$$  \hspace{1cm} (4.5)

- **Time Data Retrieval** evaluate the number of milliseconds spent interacting with data sources and data extensions for all data sets in the main report and all of its sub-reports. This value includes time spent opening connections to the data source and time spent reading data rows from the data extension.

- **Time Data Processing** consider the number of milliseconds spent in the processing engine for the request of data source.

- **Throughput** measures the transfer rate of the service in a given time interval. It is measured by successful execution request and successful execution rate. The service successful execution request ($sereq$) in Equation 4.6 is the number of service requests completed successfully in a given time interval. In Equation 4.6, $N_C$ indicates the number of times that the service has been successfully completed within the interval $t_1$, $t_2$.

$$sereq = \frac{N_C}{t_2 - t_1}$$  \hspace{1cm} (4.6)

The service successful execution rate ($serat$) in Equation 4.7 is the rate of service requests successfully completed in a given time interval related to the executed requests. It is related to the successful execution request. In Equation 4.7, $N_C$ indicates the number of service executions that has been successfully completed in the interval $t_1$, $t_2$ while $N_{C_{\text{max}}}$ indicates the maximum number of executions that the service is able to complete with success.
within the same time interval. \( N_{C_{\text{max}}} \) can be computed starting from \( N_C \) and \( F \).

\[
serat = \frac{N_C}{N_{C_{\text{max}}}} \tag{4.7}
\]

- **Scalability** refers to the capability of increase the number of operations or transactions processed in a fixed time by the service.

- **Scheduling** evaluates the quality level of the service assigning resources.

### 4.5 Service Quality Parameters

With respect to the process view our main focus is on every participant to service delivery that has to do its best to reconcile requirements from citizens and PA needs. To reach this aim we defined the BP4PA quality framework. It supports service delivery assessment based on the associated Business Process component. Moreover its application can drive administrations toward qualitative standard of design and delivery of digital services. BP4PA implements five quality dimensions and for each of them introduces domain dependent quality levels. They are summed up in Table 4.1

<table>
<thead>
<tr>
<th>Coordination</th>
<th>Control</th>
<th>Sharing</th>
<th>Transparency</th>
<th>Inclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lack of coordination</td>
<td>Reactive</td>
<td>No Sharing</td>
<td>No Transparency</td>
<td>Channel</td>
</tr>
<tr>
<td>Communication</td>
<td>Proactive</td>
<td>Data Sharing</td>
<td>Activity Aware</td>
<td>Profile</td>
</tr>
<tr>
<td>Collaboration</td>
<td>Creative</td>
<td></td>
<td>Role Aware</td>
<td>Language</td>
</tr>
<tr>
<td>Semantic Integration</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1: BP4PA Quality Dimensions.

4.5.1 **Coordination**

With the term “coordination” we mean the capability of two or more Public Administrations to work together to accomplish common goals through the delivery of a service to a citizen using ICT technologies. There is a general interest on e-government service coordination which includes both people and information systems [125] [162] [180] [161]. Our framework distinguishes PA coordination capabilities with respect to the delivery of an digital service according to the following levels.
Lack of Coordination is the lowest level of coordination observable within a PA according to two possible situations (with reference to the delivery of a specific service). In the first case direct interactions between administrations, which are participating to the delivery of a service, are not precisely established. Indeed in such a case it is in general the citizen that, knowing the PAs involved in the provisioning of the service, drives the process, financially moving back and forth from one administration to the other. Lack of coordination can also emerge when, given wrong service delivery specifications, interactions activities could result in blocking conditions, such as a message entering within a PA that is not handled by anyone, resulting in possible lost of citizen service requests.

Communication is implemented when a service enters the participating PA through messages carried on through ICT technologies. Nevertheless before leaving the organization the service requires some collaboration to a civil servant. This is the case in which the organization has introduced ICT technologies but has not fully integrated its electronic systems. So, for instance, the request coming from another PA has to use an e-mailing system. The request is then processed by a civil servant that could reply providing the needed information with another e-mail.

Collaboration enables PA participates to the e-service delivery with a fully automated Business Process. Request from other participating PAs enter the organization using ICT technologies and are handled in a completely automated manner without requiring the intervention of a civil servant. It is worth noting that in some cases regulation and laws could impose human intervention such as for instance when physical document signature is requested.

Semantic Integration implements the highest capability of coordination implementing collaboration mechanisms enriched with semantic support. Explicit formal specification of the reality related to the delivery process is shared between the participants to guarantee the understandability of the communications.

4.5.2 Control

With the term “control” we refer to the control paradigm applied to drive the digital service delivery from its start to the final fulfillment. Particularly critical in this respect is the modality in which an digital service process is activated and successively kept alive. Fostering digital service usage asks for implementation of services that announce themselves to interested citizens, instead of an approach in
which the service delivery only waits for an interested citizen. The two different paradigms have profound impact on usage. Too often digital services are not used since they are not known by citizens. According to this classification we distinguish three different levels.

**Reactive control** corresponds to the simplest way, for the PA, of controlling the activation and delivery of a digital service. It is implemented by Public Administrations that wait for citizens’ requests before they start the service delivery. Citizens’ awareness on to do list (in line with law specification) is the only driver for the service activation and delivery.

**Proactive control** enables that in the provision of the digital service the administration may work as a proactive participant. In such case the digital service announces its availability through direct communications to interested citizens also providing precise references to the access point of the service itself. This is for instance the case in which a tax payment service sends an e-mail to the citizen before the deadline, specifying also a specific link in which the user will find all the necessary information to proceed with the payment. Certainly proactive control does not make sense for all different kinds of services. There are services which are inherently reactive. Nevertheless if possible proactive control can greatly foster service usage.

**Creative** is the capability of a service delivery with reference to activation policies. This characteristic refers to the presence of activities related to the promotion of related, and maybe relevant, services. In such case the PA implements services that inform the citizens of all the other services in which he/she may be interested in (or that the citizen has to activate). For instance a citizen that starts the procedure for getting married may be interested in services related to the provisioning of low rate mortgages, sustained by the municipality. Clearly in case related services have been implemented following the proactive paradigm the citizen will also receive information on the corresponding access point.

Finally it is worth noting that reactive and proactive control can also be applied to classify intermediate interactions with the citizen. So for instance when some documents are ready the administration could wait for the citizen or could send a message to him/her.

### 4.5.3 Sharing

With the term “sharing” we identify the way in which the PA handles and shares citizen data with other administrations in order to participate in the delivery of a
specific service. This aspect seems to us particularly relevant in those countries in which to store and maintain citizen data it is necessary to receive an explicit authorization from the citizen, in particular when the related data are not those explicitly foreseen by the law. For instance birth related data can be stored only by the municipality of residence. In such a case the “car register” should ask authorization to car holders to store data in its databases if it is not able to retrieve the information from the municipality. Citizens generally feel uncomfortable when they use a service that asks for authorization to store citizen data. They suspect perceives that their data will not be in the “right” place.

**No Sharing** is the lower level of the digital service sharing and is observable when the administration take trace of all citizen data and does not try to retrieve it from the right sources. Among the various issues that this way of organizing a service brings we should certainly mention data redundancy and possible misalignment among PA data storages.

**Data Sharing** this level the service implemented by the PA to fulfill a request never includes requests for authorization to store data. Instead each citizen related data are retrieved interacting with the specific PA that is in charge of maintaining the needed information. It is worth noting that it is possible that authorization requests are included in the process since the PA authorized to manage needed data is not able to provide them in a fast and efficient way.

### 4.5.4 Transparency

With the term “transparency” we mean the ability of the administration to make citizens aware of the delivery process, improving citizens’ perceived trust. Clear and reliable view on the service execution activities typically makes the citizens more satisfied by the provided digital services. Different aspects have to be properly set up by the administrations, we refer to the complete specification of the service activities and the link between activities and the civil servant in charge of control service execution. In the context of transparency we refer to the following levels.

**No Transparency** is the lowest level of e-service transparency and is observable when the activities of a given service are not visible outside of the administration. Therefore a PA implementing this level of transparency makes the citizens completely unaware of the process execution. The citizen can just activate the service and wait for its end. Clearly the citizen, in particular in case of a long lasting service, could feel frustrated given the lack of feedbacks on the execution.
Activity Aware is observed when the administration implements service tracking mechanisms. In general it is highly desirable to make the citizen aware of the activities that have been already carried on and of the activities that need to be completed. So in general the activities composing the service and their organization are made visible to the citizens. Obviously the granularity of visible activities can be variable and a right balance should be found in order to not submerge the citizen with “not so relevant” information. The citizen will certainly feel much more comfortable with such a kind of digital service delivery and will not feel so much affected by long lasting service if it periodically receives “still alive” communications.

Role Aware at this level activity aware transparency is implemented and enriched by the specification of an activity responsible. This is a civil servant that is in charge of monitoring and controlling the valid execution of the service (and of executing it in case the process is not fully automated). In this way if the citizens feel that something is going not as it expects he can directly contact the responsible asking for information.

Transparency policies can be supported by one or more administration according to the complexity of the service. On the local site process view transparency allows high trust between citizens and PA. It is worth noting that transparency could require high levels of cooperation among participating PA. Information on business activities within a PA should be visible to the citizen through the service access point.

4.5.5 Inclusion

With the term “inclusion” we mean the ability of the administration to provide service to the citizens considering the diversity. Different capabilities, economic and social condition as well as geographical diversity has to be rethinks in the Information Society that is influencing the e-government service use. In our investigation we underline three major source of diversity that mainly impact on the delivery process.

Channel Inclusion refers to different ways that can be implemented to access the service. Users may interact with service via many and heterogeneous devices, such as PC, wap phone, PDA, ...(see [164] [53] for a review of e-government access devices). We points out service capabilities to support multiple service delivery channels within the delivery process (Web, mobile computing, ...).
Profile Inclusion refers to the service capabilities to support citizens physical diversities. It is very relevant for disadvantaged people that are often excluded and marginalized by the introduction of Information Society. The e-government has to consider all the citizens and support service distribution without making distinction. So introduce such feature in the delivery process represent a suitable starting point enabling the wide adoption of digital service. In some case the same consideration made with people that are physically diverse should be done also in relation to the experiences and the education level of the target users to make sure that service are for all of them.

Language Inclusion refers to the ability of the service to be used by people with different nationalities and switch among languages during the service delivery. This is particularly interesting toward the trans-national service delivery enabling a fruitful provision on the European e-government pan.

4.6 Validation of the Model

In this section we briefly introduce the evaluation done on the proposed model. Such evaluation was also enriched by the results of meetings with domain experts enabling the definition of the proposed quality model.

Two different approaches were conducted. The first one is based on quantitative evaluation and it was applied to subjective and ICT related parameters. The second one is based on a qualitative evaluation and it was applied to service related parameters.

4.6.1 Subjective and ICT Parameters

According to subjective and ICT related parameters we request to a sample group of people to validate our quality model. The results of such evaluation return the discussed quality parameters. We choose 140 people, most of them are skilled in e-government domain.

The forty-four questions proposed to the citizens has been structured as below. The questions (1-5) refer to the interviewee personal information (i.e. gender, nationality, age, instruction level, occupation, geographic localization, etc.). The questions (6-12) introduce information related to interviewees cultural level and e-government awareness. These questions are also used to mark the questionnaire measuring the goodness and the reliability of the answers. The questions (13-17) support a first definition of the macro-area weight starting from the percentages specified by the user to implement the impact of the specific parameters on quality.
They are able to introduce users expectations and define the PAs communication capability observed by the user. The questions (18-42) are used to really validate the considered parameters and metrics and enable the definition of the proposed model. The questions (43-44) ask about the completeness of the proposed quality approach.

Related to the interviewees we report the following results. 83.3% of them are man and 16.7% are woman. Related to the age: 59% are less than 30 years old, 37.9% are between 31 and 45 years old, and 3.1% are more than 45 years old. Focusing on the instruction levels: 45% of the interview have primary and secondary school certification, meanwhile 55% have an university degree. Our interviews are: students 43.9%, public employee 34.8%, and firms employee 21.3%. About e-government aware: 95.4% of the interviews know e-government, meanwhile 4.6% of them do not know it. About the 95.4% of users that know e-government 43.9% have already used its e-services, meanwhile 56.06% didn’t use them. Related to the ICT experience: 60.6% are users with low experience, 28.78% are with a medium level, and 10.61% have an height level skills. Focusing on the people with experience we ask about their opinion on the introduction of adapted services: 54.54% of them agree, and 45.46% disagree. In particular, only 4.5% of interviews are prone to distribute personal information. About macro-area specification we refer to Table 4.2 and Table 4.3. Table 4.2 introduces the classification observing citizens ICT experience. We observe an importance role of e-government parameters for low and medium users experience. At the same time users with high ICT experience give more importance to front-end, back-end and infrastructural parameters. Table 4.3 focuses on the distribution of macro-areas importance in relation to users age. We observe that front-end parameters are relevant for people with less of 30 years old and infrastructural parameters are relevant for people between 30 and 45 years old. At least, people with more than 45 years old focus on back-end parameters.
### 4.6.2 Service Parameters

In this sub-section we introduce a first evaluation in applying the framework to the current situation in the most of Italian local government. We conducted a survey according to the proposed level of service delivery parameters.

The results of the survey have shown that in small municipalities the current implementations of e-government digital services are not used as proved by access log files of service delivery systems. It also confirms a technological oriented services design and delivery via web interfaces that are not integrated in the administration. In particular according with our framework we recognize lack of coordination for what concerns the capabilities of the administrations to work together and only few of them implement communication policies for documents exchange. So, for instance the moving service is independently managed by registration and deregistration municipalities. For what concern control it is never observed even if recognized as an important way to proceed. Municipalities guaranteed shared registry data in line with the regulation on National Registry Office Index. Unfortunately, it is not valued as it should be and registry updating is just a time consuming activity due to lack of integration in e-government service delivery. Finally the general lack of process specification hold process transparency that should be guaranteed according to national law. Right now transparency is implemented by informative services without impact on the processes itself. The scenario is not so different in big municipalities where internal competencies try to support the promotion of e-government actions and the definition of service delivery process with poor results.

### 4.7 Considerations

After having noticed the huge gap existing between the availability of e-government digital services and their real usage we believe that that an effective e-government solution has to clearly consider quality requirements suitable to improve the quality of the services at different level and at the same time the perception of PA within the citizen.
To do that we propose a quality model that permits to judge how good is an e-government digital service. The model refers to citizens satisfaction, technology infrastructure quality and service delivery quality. In our hypotheses obtaining good marks according to the framework should result in higher usage by the citizens. Indeed we conducted a first evaluation of the model on the Italian scenario and it seems that the model fits with the reality.
Part III

A Formal Methods in e-Government
Chapter 5

Background: Business Process Management

Business Process Management is a management approach focused on aligning all aspects of an organization with the needs of customers [88]. It is a holistic management approach that promotes business efficacy and efficiency while striving for innovation, flexibility, and integration with technology [187]. Business Process Management attempts to improve Business Processes continuously. It could therefore be described as a way of process optimization. It is argued that BPM enables organizations to be more efficient, more effective and more capable of change than functionally and traditional hierarchical management approach.

The recent progress of BPM is reflected by the growth of the related industry. Wintergreen Research estimates that the international Business Process Management and services oriented to the architecture engine markets in 2007 was $1.3 billion for licenses, maintenance, and services and it is expected to reach $4.6 billion by 2014 [171]. The relevance of business modeling and business intelligence to the general BPM initiatives has been considered by Gartner among the Top 10 Strategy technologies for 2008 and 2009 respectively.

In this Chapter we provide a brief overview on the adoption of Business Process Management on the base characterization regarding process and process modeling. Quality issues in Business Process Management are also summed up as well as Business Process Management Notation (BPMN) and enabling technologies. The Chapter is concluded by a characterization upon BPM on e-government.
5.1 History of Business Process Management

One of the first people to describe processes was Adam Smith in his famous example of a pin factory \[3\]. Inspired by an article in Diderot’s Encyclopdie, Smith described the production of a pin in the following way: “One man draws out the wire, another straightens it, a third cuts it, a fourth points it, a fifth grinds it at the top for receiving the head: to make the head requires two or three distinct operations: to put it on is a particular business, to whiten the pins is another ... and the important business of making a pin is, in this manner, divided into about eighteen distinct operations, which in some manufactories are all performed by distinct hands, though in others the same man will sometime perform two or three of them.” Subdivision of labor, however, requires coordination between subtasks. Business Process Management is concerned with coordination mechanisms in order to leverage the efficient creation of goods and services in a production system based on such subdivision of labor. The individuals tasks and the coordination between them are, therefore, subject to optimization efforts. Frederick Taylor advocated the creation of an optimal work environment based on scientific methods to leverage the most efficient way of performing individual work steps \[198\]. In the optimization of each step, he proposed to select the quickest way to eliminate all false movements, slow movements, and useless movement and to collect into one series the quickest and best movements. The efficient coordination of Business Process is demonstrated by the innovation of the assembly line system. Henry Ford proudly praised the production cycle of only 81 hours from the mine to the finished machine in his factories to illustrate the efficiency of the concept \[65\].

Since the seminal work of Nordsieck, German management science has further differentiated organization into structural and process organization. In this context, Nordsieck was the pioneer in such distinction \[145\]. He identifies the order of work steps and the temporal relationship of tasks as the subject of process organization whose overall concern is the integration of these steps. Different levels of automation where distinguished by Nordsieck: free course of work, concerning the contents bound course of work, concerning the order bound course of work, temporally bound course of work, and concerning the beat bound course of work. The focus on structural organization governed the research in organizational field in the decade after World War II at least in German speaking countries.

At the same time the discussion about office automation as an excellent opportunity to greatly improve information systems. In the 1950 such scenario seams quite visionary, but later in 1970 it was very popular as a new design dimension in an organizational setting. Research in office automation, which flourished between 1975 and 1985, laid the groundwork for the development of industrial workflow applications through the analysis of technology support for administra-
tive processes. The first approaches towards the automation of Business Processes were part of the office automation prototypes at Xerox Parc (Officetalk, developed by Skip Ellis and Gary Nutt) and Wharton (SCOOP, developed by Michael Zisman). The focus of office automation research was to reduce the complexity of the user’s interface to the office information system, control the flow of information, and enhance the overall efficiency of the office [57]. The design of both systems started in the middle of the 1970s, but the idea of process automation through information technology can be traced back as far as 1968, when Fritz Nordsieck wrote: “Think about a modern data processing system. It represents a perceptible process, that is [...] connected with the Business Process and accompanies or even controls this process during various segments.” [144]. An overview of further work on office automation is provided in [57].

Following a schematization of the BPM evolution already presented in this section.

- Adam Smith - subdivision of the work - 1776.
- Frederick Taylor - optimality of single steps - 1911.
- Henry Ford - efficiency concept - 1926.
- Fritz Nordsieck - structural organization - 1934.

At this point, we could distinguish the evolution of Business Process Management following two main focuses: management and workflow areas.

### 5.1.1 Evolution of Management Area

Perhaps it is worthwhile taking a few moments to understand a very brief recent history of management’s focus on Business Processes. In the 1980’s there was a considerable focus on Total Quality Management (TQM). This was followed in the early 1990s by Business Process Reengineering (BPR) as promoted by Hammer and Champy in the Manifesto of business revolution [87] and process innovation as promoted by Devemport in the “Process Innovation” book [46].

Following BPR in the mid- and late 1990s, Enterprise Resource Planning (ERP) systems gained organizational focus and became the next big thing. These were supposed to deliver improved ways for organizations to operate, and were sold by many vendors as the “solution to all your problems”. The ERP systems certainly did not solve an organization’s process issues, nor make the processes as efficient and effective as they could have been. Towards the end of the 1990s and in the early 2000s, many Customer Relation Management (CRM) systems were
CHAPTER 5. BACKGROUND: BUSINESS PROCESS MANAGEMENT

rolled out with extensive focus on the customer view and customer experience. While this provided focus on the front office, it did not improve the back-office processes. More recently, Six Sigma has started to come into its own. The BPM hype cycle in Figure 5.1 shows a summarized view of how the process cycle has progressed over the last two decades. Six Sigma was invented in 1986, and created an awareness of processes. This was followed in July 1990 by the Hammer review article title “Don’t automate, obliterate” [86], and the BPR movement started. It’s worth noting that the BPM has been around for some time and created significant interest and discussion, as proved by the book publication of Howard Smith and Peter Fingar title “Business Process Management: The Third Wave” [188]. So, it could be argued that BPM is the most important topic on the management agenda and it provides instant gratification and return of investment.

5.1.2 Evolution of Workflow Area

It was only in the early 1990’s that workflow management prevailed as a new technology to support Business Processes. An increasing number of commercial vendors of workflow management systems beneficiated from new business administration concepts and ideas such as process innovation and Business Process reengineering as showed in Figure 5.2.

The commercial exploitation of workflow technology began between 1983 and 1985, fostered by advances in imaging and document management technology on the one side, and enhanced e-mail systems that extended traditional point-to-point mail routing with a predefined process map on the other side [197]. From this

Figure 5.1: BPM Hype Cycle (Source [105]).
first generation of workflow systems, only few vendors like are still active [246]. Figure 5.3 shows a history of commercial workflow systems.

In the 1990’s, the application of workflow systems, in particular those supporting information systems integration processes, profited from open communication standards and distributed systems technology that both simplify interoperability with other systems [75]. The Workflow Management Coalition (WMC) founded in 1993 is of special importance for this improvement [95]. The historical overview of office automation and workflow systems given in [246] nicely illustrates this breakthrough. Up to the late 1990’s intra-enterprise processes remain the major focus of Business Process Management [48].

Since the advent of the eXtended Markup Language (XML) and web services technology, application scenarios for Business Process integration have become much easier to implement in an inter-enterprise setting. Current standardization efforts mainly address interoperability issues related to such scenarios (see for instance [132]). The common interest of the industry to facilitate the integration of inter-organizational processes leverages the specification of standards for web service composition like the Business Process Execution Language for Web Services (BPEL), for web service choreography like the Web Service Choreography Description Language (WS-CDL), or for inter-organizational processes based on ebXML and related standards (see [94] for an overview).
Figure 5.3: History of Commercial Workflow Systems (Source [246]).
5.2 Understanding Business Process Management

Today, Business Process Management is an important research area that combines insights from business administration, organization theory, computer science, and computer supported cooperative work. Furthermore, it is a considerable market for software vendors, IT service provider, and business consultants.

Business Process Management supports Business Process providing methods, techniques, and software to model, implement, execute and optimize BPs involving humans, software applications, documents and other sources of information. The terms process, Business Process and Business Process Management are seemingly self-explanatory enough that they are used most of the time without an explicit definition.

In this section we provide some Business Process definitions and we try to contextualize them in BP life cycle.

5.2.1 Business Process Definitions

Business Process is a collection of related and structured activities undertaken by one or more organizations in order to pursue some particular goals. Within an organization a BP results in the provisioning of services or in the production of goods for internal or external stakeholders. The execution of a BP often results in the activation of related BPs within the same or other organizations. Further discussion on BP and related topics can be found in [120].

Business Processes operate in the context of constraints, regulations, and defined roles and relationships. Everyday examples of task-level Business Processes include registering to vote, buying a book from a web site, or changing your address. Activity-level processes might include preparing your taxes or applying for a mortgage. Process-level processes have many steps, for example, buying a house or selecting and enrolling in a university. The highest level processes are ongoing, complex activities such as marketing a small business or building a shopping center. Sometimes the value created by a process is easy to see. Other times a process indirectly creates value by performing a task related to accounting, administration, and other management functions. There are many other definitions of Business Processes, but they all come down to this simple level. The challenge then becomes to describe how the inputs are transformed into outputs.

5.2.2 Business Process Life-Cycle

One of the major ingredients of BPM governance is life-cycle management. According to Merriam Webster’s on-line dictionary definition life-cycle is “a series
of stages through which something (as an individual, culture, or manufactured product) passes during its lifetime”.

Process Management life-cycle has to deal with bringing a new Business Process to life, modifying or optimizing an existing Business Process, and continually innovating a Business Process involving a similar set of phases and activities. The Business Process life-cycle visually depicts this ongoing circle of design, implementation, run/monitoring and analysis as show in Figure 5.4. The cycle starts with the analyze step, in which the new or existing processes are studied so that the requirements for the process are understood. The next step is design, in which the process is defined in detail, followed by implement, in which the process is supported or automated to the greatest extent possible. In these steps, the process comes off the drawing board and comes to life, first as a Business Process model and then as some form of technology that assists with implementation. Finally, the process is run and monitored. If there is room for improvement, the cycle starts all over again. It is not uncommon for process simulations to be employed in the design and implement steps to improve the understanding of how a process will work in production. This cycle represents a common-sense approach to viewing processes.

Going into detail and referring to a BPMN life-cycle model showed in Figure 5.5 we can observe a methodology based on an iterative and incremental process for implementing Business Process Management.
CHAPTER 5. BACKGROUND: BUSINESS PROCESS MANAGEMENT

83

Figure 5.5: BPMN Diagram of Business Process Life-cycle.

- Process modeling defines the process models using the selected methodology and notation (below in this Chapter we will focus on notation suitable for BP modeling such as BPMN).

- Process implementation sustains the end-to-end IT support for the process. At this stage Service Oriented Architecture provides technologies and tools to make the implementation phase quick and efficient.

- Process execution and monitoring guaranties to gather the Key Performance Indicators.

- Process simulation makes an execution of the process with the objective gathering KPIs and identifying optimization points.

- Process optimization improves the process efficiency, effectiveness, agility, flexibility and transparency.

Notable is the relationship between design and modeling where the former refers to the overall design process involving multiple steps and the latter refers to the actual representation of the Business Process model using a process language [207]. To this end, the term Business Process modeling is used to characterize the identification and (typically rather informal) specification of the Business Processes at hand.
5.3 Business Process Design

Business Process design can be observed, in a broad sense, from a number of different perspectives that are below discussed [208] [100].

- The control-flow perspective (or process) describes tasks and their execution ordering through different elements, which permit flow of execution control, such as sequence, choice, parallelism and join synchronization. Tasks in elementary form are atomic units of work, and in compound form modularizes an execution order of a set of tasks.

- The data perspective deals with business and processing data. This perspective is layered on top of the control perspective. Business documents and other objects which flow between activities and local variables of the workflow, qualify pre- and post-conditions of task execution.

- The resource perspective provides an organizational structure anchor to the workflow in the form of human and device roles responsible for executing tasks.

- The operational perspective describes the elementary actions executed by tasks, where the actions map into underlying applications. Typically, (references to) business and workflow data are passed into and out of applications through activity-to-application interfaces, allowing manipulation of the data within applications.

Different approaches have been investigated focusing on one or more of these perspectives. Some are general purpose some others are domain dependent. Notable is that the selection of an appropriate process modeling notation is critical for the success of the process analysis.

In the rest of the section we present an overview on modeling notations, we give details on one of these Business Process Modeling Notation and a comparison between BPMN and UML Activity Diagram.

5.3.1 Business Process Modeling

Recent works show that BP modeling has been identified as a fundamental phase in BPM. The quality of BPs resulting from the BP modeling phase is critical for the success of an organization. However, modeling BPs is time-consuming and error-prone. Therefore, how to help organizations to implement high-quality BPs, and increase process modeling efficiency has become one of the topics attracting a lot of attention from industry and academy. Many different commercial tools
have been developed to support BPM but what concerns modeling phase they just provide support for BP editing and syntactical analysis.

Different classes of languages to express BPs have been investigated and defined. There are general purpose and standardized languages, such as Business Process Modeling Notation [218], Event-Driven Process Chain [131] and UML Activity Diagrams [152], and there are also more academic related languages such as Yet Another Workflow Language (YAWL) [206], based on Petri Nets, and Communicating Sequential Processes [93] Calculus of Communicating Systems [135] Process Algebras. Finally there are also domain specific languages such as PICTURE in e-government [16].

5.3.2 Business Process Modeling Notation

Business Process Modeling Notation was developed by Business Process Management Initiative. It is currently maintained by the Object Management Group. The primary goal of BPMN is to provide a notation that is easily understandable by the business analysts that create the initial drafts of the processes, the technical developers responsible for implementing the technology that will perform those processes, and the business people who will manage and monitor those processes.

BPMN will be constrained to support only the concepts of modeling that are applicable to Business Processes. This means that other types of modeling done by organizations for business purposes will be out of scope for BPMN. For example, the modeling of the organizational structures and resources, functional breakdowns, data and information models, strategies and business rules will not be a part of BPMN. BPMN supports three basic types of sub-models as below reported.

- Private (internal) Business Processes are those internal to a specific organization.
- Abstract (public) processes represents the interactions between a private Business Process and another process or participant.
- Collaboration (global) processes depicts the interactions between two or more business entities. These interactions are defined as a sequence of activities that represent the messages exchange between the involved participants.

A BPMN process is made up of BPMN elements. Figure 5.6 provides an overview of a set of BPMN elements related to control-flow specification. BPMN includes objects, sequence flows, and message flows. An object can be an event, an activity, or a gateway. A sequence flow links two objects in a process diagram.
and denotes control flow (i.e. ordering) relations. Message flows are used to capture the interaction between processes. The Figure 5.6 does not show BPMN elements that do not have a control-flow semantics such as lanes, artifacts, groups, and associations.

An event may signal the start of a process (start event), the end of a process (end event), and may also occur during the process (intermediate event). A message event is used to send or receive a message. A timer event indicates that a given time instant has been reached, and an error event signals a fault or exception raised during the process. There are other types of events in BPMN, namely link events, rule events and terminate events. Link events are a notational convenience to spread a model into several pages and therefore they do not affect the semantics of a model. Rule events are similar to message events. They only differ in the way they are triggered. Rule events are triggered by data updates while message events are triggered by arrival of messages. Similarly, terminate events can be treated as a special type of error events.

An activity can be a task or a subprocess. A task is an atomic activity, standing for work to be performed. A subprocess defined as a flow of other activities. It can be invoked via a subprocess invocation. There are embedded and independent sub-processes. An embedded subprocess is part of a process while an independent one can be called by different processes. Also, an activity may have attributes specifying its additional behavior, such as looping and parallel multiple instances.

A gateway is defined as a routing construct. There are: parallel fork gateways (AND-split) for creating concurrent (sequence) flows, parallel join gateways (AND-join) for synchronizing concurrent flows, data/event-based XOR decision gateways for selecting one out of a set of mutually exclusive alternative flows where the choice is based on either the process data (data-based, i.e., XOR-split) or external event (event-based, i.e., deferred choice), XOR merge gateways (XOR-join) for joining a set of mutually exclusive alternative flows into one flow, and inclusive OR decision gateways (OR-split) for selecting any number of branches among all its outgoing flows. In particular, an event-based XOR decision gateway must be followed by either receive tasks or intermediate events to capture race conditions based on timing or external triggers (e.g., the receipt of a message from an external partner).

An intermediate message, timer, or error event attached to the boundary of an activity signals an exception. The occurrence of the activity will be interrupted upon the occurrence of the exception, and the process execution along the normal sequence flow will switch to the exception flow at the point when the exception occurs. Note that an error event on a normal sequence flow models throwing an error, while one attached on the boundary of the activity models catching an error. This is similar to the strictly hierarchical throw-catch mechanism used in most programming languages.
A message flow is used to show transmission of messages between two interacting processes via communication actions such as send/receive task or message event. The two processes are located, respectively, within two separate pools, representing two participants. In graphical representation, a message flow is drawn as a dashed line with an open arrowhead connected to the target process and a circle connected to the source process, and a pool is drawn as a rectangle labeled with the process name.

Finally, a BPMN model is composed of a set of BPMN processes which are related to each other via subprocess invocation activities or message flows.

According to BPMN specification releases we observe the following steps.

August, 2001  Formation of Notation Working Group.

November, 2002  BPMN 0.9 draft specification was released to the public.

August, 2003  the BPMN 1.0 draft specification was released to the public.

May, 2004  BPMN 1.0 specification was released to the public.


June, 2007  BPMN 2.0 RFP: Request for Proposals for version 2.0 of BPMN.
5.3.3 BPMN vs. UML Activity Diagram

The root difference between BPMN and UML Activity Diagrams can be summarized as follows. UML Activity Diagrams are an execution-oriented language. Their execution semantics is defined in quite some level of details, and various formal semantics of UML activity diagrams have been defined, so it is feasible to build an execution engine for UML Activity Diagrams. On the other hand, BPMN has been designed with the aim of being a notation for high-level modeling. As a result, several constructs in BPMN do not have a fully defined execution semantics. BPMN models are not intended to be directly executed. They need to be refined (e.g. into BPEL processes) prior to execution. The rationale for this is that BPMN will be used by domain analysts whose goal is not to produce a system implementation, but rather a set of requirements to be handed over to ICT analysts and software developers. At the most, BPMN might be used to generate templates of BPEL code that developers will have to refine to obtain a running implementation. For good or for bad, it is possible that this state of affairs will change in the future though, since BPMN has been sort of taken over by the OMG, and the OMG is likely to put some pressure to obtain a more precise specification of BPMN. Such effort is going to be introduced in BPM2.0 that is in the request for proposals status. At a more detailed level, the main differences between BPMN and UML activity diagrams are following reported.

- Minor lexical differences and differences in nomenclature (e.g. the symbols used for AND-split and AND-join differs between BPMN and UML Activity Diagrams).

- BPMN has a notion of “event-driven choice” whether UML activity diagram does not. BPMN probably got this feature from BPEL which provides a construct called PICK with the same intended semantics.

- UML Activity Diagrams relies on “signals” where BPMN relies on “events”. Some people consider that signals are a lower-level concept than events. Also, BPMN offers a “zoology” of predefined event types, whereas UML activity diagrams does not specify any signals with special semantics a priori (they are all user-defined).

- BPMN offers a larger number of “control-flow constructs” (i.e. gateways) than UML activity diagrams. In particular BPMN has “OR-splits”, “OR-joins”, and so-called “complex gateways”. The OR-split and especially the
OR-join are basically taken from the workflow patterns and can also be found in YAWL.

5.4 Quality in Business Process Management

Quality in Business Process is the main driver to achieve continual organizational improvement. The ISO underlines the importance of quality or performance management principles to guide organizations towards improved performance [42]. Among others International Standardization Organization refers to process approach principle: “a desired result is achieved more efficiently when activities and related resources are managed as a process”. An interesting discussion on the potential of quality metrics in Business Process Management can be found in [30].

In the next subsection we present benchmarking, such as the main driver of quality in BPM and we provide few metrics to measure process quality.

5.4.1 Benchmarking

In the last decades, the word benchmarking has become relevant within the business management community [4]. The most well-known and comprehensive definition in this area is given by Spendolini [190]. He defines benchmarking as “a continuous systematic process for evaluation of the product, services and work for organizations that are recognized as representing best practices for the purpose of organizational improvement” [190].

Different classifications of benchmarking are discussed in literature. The main one was presented by Camp in 1989 [29], it refers to internal, external, competitive, industrial and generic benchmarking. Another classification categories benchmarking according to the nature of the objects under analysis refers to process, performance and strategic benchmarking [114]. In particular the process benchmarking, supports to the comparison between discrete work process and systems and it is appropriate for overhead cost areas and intra- and cross-organizational approaches [21] [7]. Several studies have illustrated various aspects of process benchmarking [2] [91] [214]. However, only Gleich et al. present the firs attempt of a cross-organizational process benchmarking [76]. They develop and present a performance measurement tool for recording activity and sub-process related cost and performance.

During benchmarking a clear reference model has to be addressed to drive the analysis, we cite Balance Scorecard [107] and SERVQUAL [241] as possible reference models. They support quality assessment in organization both in a single and integrated fashion [4].
5.4.2 Metrics in Business Process

In the past, performance management has focused on measuring results, mostly at the end of the process and usually in financial terms. The modern practice of performance management applies measurements at various steps in processes. By applying performance management to BPM, it becomes possible to identify problems much earlier in the operational cycle while there is still time to do something about it. In addition, as the relationships between the inputs and outputs of each process are better understood, the models become more predictive, making possible to create better forecasts of business activity. The following aspects are critical in applying performance management in organizations.

- Measurements take place at key control points at the beginning, middle, and end of the process.

- Operational measures that indicate the pace and quality of processes become more prominent. These operational measures track non-financial inputs and outputs to processes, allowing more detailed analysis. Metrics become better aligned with business objectives because they measure quantities that can be connected to specific value-creating processes.

- Due to the fact that the inputs and outputs of processes are being measured, it becomes possible to make Business Process models predictive tools. Inputs such as pipeline coverage may be eventually related to the number of sales closed. It means that a drop in pipeline coverage could be an important warning sign.

- The granularity of measurement increases, providing not only metrics at the aggregate level at the end of a process, but also figures that break down metrics according to the product, geography, department, customer, supplier, and so on.

- When a process is too complex and ad hoc to be defined by a Business Process model, the numerical model of the inputs and outputs provides a way to track and monitor performance of the process.

According to such general overview the problem is to define or to choose the best indicators for the process. Unfortunately, there is not much research on Business Process metrics as mentioned in [211]. Interesting approaches in quality evaluation can be found in literature with focus on different quality aspects such as process understandability [134], [133], complexity [30] and more general software related quality concepts [6], [35], [89], [211], [83], [192]. In particular coupling, complexity, modularity and size that are well known indicators in software architecture are discussed in business management field [211]. Generally, complexity...
and related understandability represent the main aspects that are investigated in line with Business Process measurement (see for instance [134] [79] [133]). More recently quality aspects of Business Process are discussed [217] [113] [113] [83]. Wetzstein et al. introduce a framework for performance management based on Key Performance Indicators in the semantic Business Process [217]. Lam et al. proposed a quantitative approach using an activity model for business modeling and analysis, in which adjacent matrixes can be applied to provide explicit performance indicators for the enterpriser to identify the inefficient and ineffective activity looping, and the Business Process flow can then be improved [113]. Finally, Guceglioglu and Deminors propose process quality attributes to measure Business Process quality based on ISO/IEC 9126 software product quality model [83].

5.5 Applying the Technology in Business Process Management

While a clear understanding of the general theory of how BPM helps manage complexity, advances alignment with corporate strategy, empowers staff to act in intelligent ways, and improves efficiency and flexibility of operations in a company is vital to success, all roads to BPM must eventually employ technology. One of the most daunting challenges for executives, technologists, and staff involved in any program of BPM adoption is understanding just what the technology presented to them does.

The purpose of BPM technology is to automate, manage, and improve productivity for the tasks involved in the description and automation of Business Processes. Through BPM technology, companies can manage their Business Processes explicitly, using tools built for the task. The main challenge of understanding BPM technology comes from the fact that it can be applied in a wide variety of ways at many different stages of BPM adoption, during which the perspective can range from encompassing the entire enterprise to a single task.

- The first purpose of BPM technology is to help describe the structure of Business Processes.
- The second purpose of BPM technology is to put a Business Process model to use in automating or execution of the described Business Process.
- The third purpose of BPM technology is to provide some supporting functions for the first two purposes. A wide variety of components can be used when describing or automating Business Processes, including the follow-
CHAPTER 5. BACKGROUND: BUSINESS PROCESS MANAGEMENT

ing: process repository, user interface widgets, process and business activity monitoring simulations.

- The ultimate purpose of BPM technology is to allow processes to be described, automated, monitored and improved as a part of a cycle of continuous innovation.

In the world of BPM, both the purpose and functionality of technology offerings overlap in vexing ways. Most BPM technology starts by allowing the person using it to describe a Business Process model. As we know from our previous analysis, this Business Process model may describe how the departments of a company interact or it could describe how a team creates a purchase order. It is not uncommon for different technology to be used for different scopes of modeling. Once that business model has been created, it can be used for the automatic assembly or creation of applications that can help execute Business Processes. It is the distinction between these two ways of using BPM technology that define the three main categories below reported.

- Technology for Business Process design and analysis is used to express the structure of Business Processes and evaluate its quality.

- Technology for Business Process simulation where the flow of work can then be analyzed to determine the optimal design for a process.

- Technology for Business Process automation allows Business Process applications to be generated based on a description of the process.

Not surprisingly, BPMN has enjoyed widespread adoption in practice, for example by tool vendors (e.g., Pega, Sparx Systems, Telelogic, Intalio, itp-commerce), education providers (e.g., Widener University, Queensland University of Technology and Howe School of Technology Management) or modeling coaches and consultants (e.g., Object Training, BPM-Training.com and BPMInstitute.org). A wider discussion on the topic can be found in [170].

5.6 Business Process in e-Government

In e-government service delivery the process components play a fundamental role as we have already presented in the e-government Chapter 2. Business process via an organized collection of business behaviors satisfies the purpose of the administration to deliver added value services.

According to Business Process Management, in e-government the main task is to find a balance between stakeholders and administrative resources driving the
administration towards a qualitative service delivery. This is confirmed by the scenario that characterizes PA where (i) services portfolio is much diverse and complex, (ii) municipality presents a lot of interconnected and inter-dependence services, (iii) an overview on the current process has to be followed, (iv) new administrations are profit oriented, and (v) there are a lot of stakeholders [16][155].

Focusing in e-government benchmarking, it introduces a further step in the services evaluation and supports measurement via comparison, the continuous improvement of the service, and systematic procedure in carrying out the overall administration. Benchmarking plays a fundamental role both for the citizen and the administration, awareness on the service capabilities and support services improvement providing a way to objectively quantify their quality.

e-Government seems to be an interesting application domain to explore and test the potentialities of BPM particularly concerning the quality, efficiency and efficacy of service delivery.
Chapter 6

Background: Process Algebras and Formal Verification

This chapter is devoted to introduce a survey of well known and well established concepts of process algebras and formal verification. In particular we provide a brief discussion on Communication Sequential Process language and model checking.

6.1 Process Algebra

The term process algebra was coined in 1982 by Bergstra Klop [22]. It refers to a family of specification techniques particularly well-suited to describing concurrent communicating components systems. More than mere documentation methods, however, they also incorporate equivalent theories that define algebraic laws such as formal reasoning systems with variables representing processes.

There is a wide set of process algebras, and related dialects, that are used in formally modeling concurrent systems and provides a tool for the high-level description of interactions, communications, and synchronization between a collection of independent agents or processes. They also provide algebraic laws that allow process descriptions to be manipulated and analyzed, and permit formal reasoning about equivalences between processes. Leading examples of process calculi include Communicating Sequential Processes (CSP) [93], Calculus of Communicating Systems (CCS) [135], Algebra of Communicating Processes (ACP) [23], Language of Temporal Ordering Specification (LOTOS) [27], p-calculus [136], the ambient calculus, PEPA and many others. An interesting discussion about the history of process algebra as an area of research in concurrency theory and the theory of parallel and distributed system in computer science can be found
CHAPTER 6. PROCESS ALGEBRA AND FORMAL VERIFICATION

in [9]. It presents a lot of interesting points towards a more detailed investigation on the topic.

There are, of course, many rivals to the process algebras as illustrated in [64]. Languages such as Z and VDM allow specifications to be expressed non-constructively, often at a much higher level of abstraction than it is possible with the process algebras. However, they have no notation to express concurrency or communication. On the other hand Petri Nets can model concurrent behavior. Moreover they have true concurrency semantics and can model causality, concepts lacking in the process algebras. Nevertheless, there is no satisfactory algebraic theory for them, and they offer a more primitive notation than the process algebras. Regular expressions and finite-state automata are simple and familiar concepts; at first glance, process algebras have much in common with them. However equivalence models for the process algebras include extensive consideration of nondeterministic choices, a feature not normally found in, e.g., regular expressions.

6.2 Communication Sequential Process

A very important contributor to the development of process algebra is Tony Hoare. He was born in 1934 and published the influential paper [93] as a technical report in 1976. Hoare’s CSP is an event based notation, primarily aimed at describing the sequencing of behavior within a process and the synchronization of behavior (or communication) between processes. In CSP, a process is a pattern of behavior and a behavior consists of events, which are atomic and synchronous between the environment and the process. Events can be constructed using the dot operator ‘.’ to form compound events; often these kind of events are used to implement channels permitting to represent a more structured communication schema among processes. Below we report the grammar in Backus Normal Form for the CSP language, where P represents a generic process.

\[
P ::= \text{STOP} \mid \text{SKIP} \mid e \to P \mid P \\& Q \mid P \parallel Q \mid P Q
\]

\[
e ::= x \mid x.e
\]

The process STOP is a deadlocked process where the process SKIP is used to represent a successful termination. Process \( e \to P \) denotes a process capable of performing event \( e \), after which it will behave as process \( P \). Process \( P \\& Q \) denotes the external choice between processes \( P \) and \( Q \); the process is ready to behave as either \( P \) or \( Q \) and external factors will make the choice among this two possibilities. Process \( P \parallel Q \) denotes the interleaved parallel composition of
processes \( P \) and \( Q \). Process \( P \parallel_A Q \) denotes the partial interleaving of processes \( P \) and \( Q \) which share events listed in the event set \( A \). Process \( P; Q \) denotes a process ready to behave as \( P \) and after that \( P \) will successfully terminate, the process will behave as \( Q \). \( x \) is a variable ranging over a set of processes. CSP processes are closed terms built up out of actions and other processes using some operators. The original CSP process algebra includes other operators that we do not present here since not strictly useful for the purpose of our work.

Finally, the operational semantics of CSP is typically given by a set of inference rules which define a mapping from CSP terms to transition system. We do not report such rules here and the interested reader can found them in [93].

### 6.3 Formal Verification

In the context of software systems, formal verification is the act of proving or disproving the correctness of a system with respect to a certain formal specification or property, using methods based on sound mathematical tools. Many different formal approaches can be applied to software system verification. Our interest is mainly in model checking techniques [37], which consist in a systematic and when possible exhaustive exploration of an operational model of the system to verify, to check if the given model satisfies a set of given properties. Implementation techniques include state space exploration, symbolic state space enumeration, abstract interpretation, symbolic simulation, abstraction refinement and others. The properties to be verified are described as goals to reach or conditions that systems states have to satisfy. Reachability analysis, deadlock-freeness analysis, and generic temporal logics properties, such as those expressed using Linear Temporal Logic [58] are typical properties that is necessary to verify on a complex system.

As described by Clarke model checking has a number of advantages compared to other verification techniques such as automated theorem proving or proof checking [36]. We refer to model checking that without constructing a correctness proof provides diagnostic counterexamples. No problems are observable with partial specifications and temporal logics. It can easily express many of the properties that are needed for reasoning about concurrent systems. Objections on the use of model checking can also be mentioned. Some papers report that temporal logic specifications are ugly, writing specification is hard and state explosion [205] is a major problem.

Since its first inception many tools have been proposed and developed. We refer to SPIN [96], UPPAAL [20], SMV [31] and PAT [196] just to cite a few. In our work we integrate the PAT model checking due to its flexibility and since it uses a variant of the CSP formalism as input language.
6.4 Formalizing Business Process

Modeling of Business Process and workflow is an important area in business and software engineering. It is desirable that a Business Process model can be understood by the various stakeholders involved in an as straightforward manner as possible [207]. As already mentioned this could be achieved through the use of graphical representations. At the same time, these stakeholders should assign the same meaning to such a model, there should not be any scope for alternative interpretations. Business Process models can be quite complex and the use of a formal language for their specification is the only sure way to guarantee that alternative interpretations are ruled out. After consensus among the stakeholders has been reached, a Business Process model can be deployed and if a formal language is used, its behavior can be explained in terms of the formal semantics of that specification language. Careful and formal analysis of process model at design time can greatly improve the reliability of such system. The lack of a formal semantics has resulted in different interpretations by vendors of even basic control flow constructs, definitions in natural language such as provided by the Workflow Management Coalition are not precise enough [109].

From a review of the current Business Process modelling/management tools we observe that the support in the design phase is limited to providing an editor and syntactical analysis lack of formal verification features. We believe that performing formal verification at design time, it is possible to identify potential problems, and if so, the model can be modified before it is used for execution. Although one would expect verification functionality to be present in any Business Process modeling tool, workflow management system, or Business Process management suite, this is not the case [232]. At best these systems do some basic syntactical checks, but allow for the modeling of processes with deadlocks, live-locks, and other anomalies.

Considering the advantages of a formal approach in Business Process we report them here below making a distinction on the stakeholders prospectives.

- Formal representation of Business Process: (i) alleviate error-prone (no guidance), significant training needed and lends itself badly to analysis and change (ii) execution model (computers can execute process) (iii) static and runtime consistency checks (iv) lends itself well to outsourcing, service-oriented architecture, and partial automation.

- Users (i) can easily adhere to established best practice (ii) know what tasks can be dealt with now/later (iii) receive help to delegate tasks appropriately (iv) need only local knowledge about the tasks they solve (as opposed to global knowledge about the entire workflow).
Designers/planners (i) can more easily map out and change processes (ii) can introduce structure along the way (ad hoc) (iii) can perform formal analysis on process (iv) can partially automate out-sourcing etc.

Controllers (i) gain finer registration of resource consumption (e.g. time) and thus costs (get activity-based costing for free) (ii) can carry out performance analysis more easily.
Quality of Digital Services in e-Government

This part of the thesis was published in [R3] [C11].

7.1 Introduction

In this Chapter we intend to introduce a methodology and a tool permitting to formally and automatically assess the quality of a designed Business Process with respect to defined quality requirements. As it will be detailed in the following the proposed approach foresees that a BP designer expert will design a process permitting to fulfill specific objectives stated in the requirements. At the same time domain experts will describe quality properties that a BP applied in the given domain should fulfill. The business process and the properties are transformed, through specific mappings we have defined, in formal specification in order to assess if the BP actually satisfy the required properties or not. In case some properties are violated the BP designer will be asked to review the process definition. The property assessment step is based on well established techniques in the area of formal verification of temporal properties for state based models. The approach has been codified in a plug-in for the Eclipse platform resulting in an integrated environment for BP specification and verification.

The approach has been also applied to the evaluation of processes in the service delivery domain for which we had previously defined a quality framework according to the process requirement introduced in Chapter [4]. In particular the defined framework provides a five dimensions evaluation schema derived considering several process delivery related characteristics influencing citizens perception and usage of an service. At the same time each defined dimension provides
design guidelines to service developers that, in order to derive a highly usable and used service, should structure the delivery process according to corresponding suggestions.

As far as we know this is the first tentative to bind user friendly environment with formal verification techniques to guarantee quality e-government requirements. Our work is in line with the recommendations discussed by Davies et al. [47] strongly supporting the application of formal methods to e-government scenarios.

Our focus is on inter-administrative services delivery because they provide interesting scenarios in line with the complexity of the Public Administrations relationships [189]. Anyway, considering the services delivery that involve only one administration, the complexity and the hierarchical structure of the administration [26] in term of departments and offices with different competencies, responsibilities can easily reduced on the inter-administrative services delivery.

In the rest of the Chapter we describe the defined approach and methodology and we discusses the prototypical implementation of the framework. Finally, e preset some interesting case studies and we discuss useful approaches on applying formal methods.

### 7.2 From Specification to Verification

In this section we outline the elements composing the BP formal verification approach we have defined. The use of formal mechanisms to verify properties of complex BPs has been already advocated by other authors (see for instance [232]). Our work aims at providing to BP and domain experts the power of formal verification techniques still allowing the use of graphical notation with which they are already acquainted. The approach, which is sketched in Figure 7.1 relies on the following three steps:

i) Business Process and quality requirements specification via a user-friendly notation;

ii) Mapping of a process specification and of a set of quality requirements to a CSP like notation and a to set of goals, respectively;

iii) Formal verification of defined processes with respect to specified set of properties (goals).

In case the verification phase ends highlighting some problems, i.e. at least one of the property defined by domain experts results to be violated, the process should be restarted.
The remaining part of this section is structured as follows. Section 7.2.1 describes the mapping from BPMN to a CSP model, Section 7.2.2 introduces issues related to requirements specification and finally Section 7.2.3 details some aspects of the verification.

### 7.2.1 BPMN2CSP

The BPMN 1.1 specification does not define a precise semantic for the provided constructs, so one of the objective of the upcoming version of this specification is to clarify the semantic of BMPN constructs. Nevertheless, even if a huge effort has been spent toward this objective, the draft version still contains descriptions in natural language. On the other hand the Request For Comment for BPMN 2.0 is still open, so it is possible that some updates will be included before the final release. In our work we have defined a precise semantic for BPMN constructs through a mapping to CSP processes. Providing a formal semantic to a semi-formal language results in the definition of a unique interpretation for each construct which could possibly be different from that intuitively given by some BP developer. We do not think that this is a big issue for our purpose given that the objective is mainly on property verification. So the BP developer should be in any case alerted by a possible interpretation leading to a low quality process. Nevertheless, in order to reduce such a risk, we have derived design rules which
impose some restrictions on the usage of the available BPMN constructs. This is particularly important when BPMN constructs miss to specify details for verification purpose or when they could lead to a particularly ambiguous interpretation.

Our mapping covers all the core BPMN constructs and almost all the constructs introduced by the OMG notation. Few constructs dealing with transactions, such as compensation events and cancel events, or time have been kept outside of our mapping. Main reason for this choice is that they are seldom used in practice at least in the e-Government domain. Another reason relates to the fact that time properties cannot be verified within the proposed approach.

In order to apply our approach, and the tool-set we have provided, the BP developer has to abide by the following constraints.

- Tasks have to be typed to support specific domain-dependent characterizations.
- Tasks can include at most one type of communication (send or receive), they can not include both. In this way we ask the BP developer to explicitly provide the order in a sequence of messages exchange. Delivery of messages are assumed to happen before exiting from the task where message reception are assumed to happen while entering the task.
- Messages have to be typed to support specific domain-dependent characterizations.
- Pools have to be typed to address the role they play in the process.
- Loops have to be explicitly represented as loop-task or loop-subprocess. No implicit cycles are admitted in the process design. This constraints permits to have more structured BPs avoiding the presence of loop generated by unconditional jump. Besides making the verification step more difficult, the presence of such a kind of loops make the specification more complex and less understandable.
- Collapsed sub-process are not supported. Moreover for each sub-process BPMN end and start events have to be explicitly provided since they support the trigger of elements inside the sub-process.

When a BP has been modeled according to the constraints the approach permits to derive a CSP model (input format for the selected model checking tool in the real implementation of the approach) from the BPMN model. The mapping has been defined according to the following general principles:
CHAPTER 7. QUALITY OF DIGITAL SERVICES IN E-GOVERNMENT

- Each BPMN graphical object included within a pool is formally represented by a CSP process or a parallel execution of generated CSP processes - we will name such process Element CSP.

- Each pool is mapped to a parallel composition of Element CSP processes with barrier synchronization. In this case no message exchange will be observable - we will name such process Private CSP.

- The whole process results from the parallel execution of the Private CSP processes including their interactions implemented via messages exchange - we will name such processes Abstract CSP.

Due to lack of space we report few mapping rules. All the rules we have defined permits to give a denotational semantic to the various BPMN elements and to their composition in term of a complex CSP process. A wider discussion on the mapping can be found in Appendix B.

- The rule to transform the BPMN pool elements produces a CSP global constant (Figure 7.2 (a)). So the general idea is that each participant will be identified by such a constant value.

- The rule to transform the BPMN sequence flow elements produces a CSP process (Figure 7.3 (b)). Such CSP process is able to perform an event esc after which it will perform the event enter. Both events are characterized by the identifiers of the flow and of the pool that contains the flow itself. The general idea is that the CSP process related flow is started by an interaction with the environment. Firstly it is requested the synchronization of the event esc with the corresponding esc event generated by another BPMN elements where such flow is outgoing. After that synchronization with the environment via the event enter is requested. Also in this case the environment is represented by the CSP process generated by a BPMN elements where the same control flow is incoming. When the CSP process output of the mapping from BPMN flow elements returns the BPMN flow is fired and the whole CSP process terminates with success. The CSP process can also terminates in case a BPMN event termination occurs according to the process and sub-process where the flow is placed. A similar mapping has been done for the rule related to the conditional flow.

- The rule to transform the BPMN start event elements produces a CSP process (Figure 7.4 (c)). Such CSP process is able to perform the event enable after which it will execute a set of esc events. The first event is characterized by the identifiers of the BPMN start and of the pool that contains such
element. Each esc event is characterized by the identifier of the outgoing flow and by the pool identifier that contains such a flow. The general idea is that the CSP process is immediately enabled without any interaction with the environment. Than the synchronization of the events esc with the corresponding esc event generated by BPMN flow elements is requested. When the CSP start event process returns the event related to the outgoing flows are fired and the CSP process is successfully terminated. Also in this case the process can be terminated when a BPMN event termination occurs. A similar mapping rule has been produced for the events typed with conditions. As well as for the BPMN start also the end event is considered in our mapping. In this case the general idea is that the CSP process enables the incoming flows on the end event implemented via other CSP process related to flows BPMN element and than consume itself.

- The rule to transform the BPMN simple task elements produces a CSP process (Figure 7.5 (d)). Such CSP process is able to perform the enter event after that it will perform the enable, work and esc events. The first event is characterized by the identifiers of the incoming BPMN flows and of the pool that contains such elements. The second and the third events are characterized by the identifier of the task and of the pool that contains the task itself. Finally the fourth event is characterized by the identifier of the outgoing BPMN flows and of the pool that contains such flows. The general idea is that the CSP process firstly interacts with the environment (with the CSP processes related to the incoming flows), then the main task is enabled and executed, and finally the process implements another interaction with the environment (with the CSP process related to the outgoing flows). More specifically first the events enter are synchronized with the correspondent set of enter events generated by BPMN incoming flows, then the enable and the work events are consumed without interacting, and finally the synchronization of the events esc with the correspondent set of esc events generated by BPMN outgoing flows elements is requested. A similar behavior is observable for the rules related to tasks characterized with loops, multi-instance both in parallel and sequence and messages (Figure 7.6 Figure 7.7). For what concerns tasks sending and receiving messages we introduced a CSP dedicate channel enabling the message exchange.

### 7.2.2 Domain Related Quality Requirements and Mapping

Domain related quality requirements, that generally characterize all the process in a given domain, should be defined by domain experts. Nevertheless domain
# define poolName poolID;

Figure 7.2: Mapping Rules From BPMN Pool sending to CSP.

FlowID(flowID, subProcID, poolID) =
(esc.flowID.poolID → enter.flowID.poolID → Skip;)
☐ (terminate.poolID → Skip;)
☐ (terminate.poolID.subProcID → Skip;)

Figure 7.3: Mapping Rules From BPMN Flow to CSP.
PoolID = "poolID";
SubProcID = "subProcID";
StartID = "startID";
OutgoingEdges = "flowID_1, flowID_2, ..., flowID_m";

\[
\begin{align*}
\text{startID}(\text{startID}, \text{flowID}_1, ..., \text{flowID}_n, \text{subProcID}, \text{poolID}) = \\
(\text{enable.startID.poolID} \rightarrow \\
(\text{esc.flow}_1, \text{poolID} \; || \; ... \; || \; \text{esc.flow}_n, \text{poolID}) \rightarrow \text{Skip};)
\end{align*}
\]
\[
\begin{align*}
\text{□ (terminate.poolID} \rightarrow \text{Skip;}) \\
\text{□ (terminate.poolID.subProcID} \rightarrow \text{Skip;})
\end{align*}
\]

Figure 7.4: Mapping Rules From BPMN Start Event to CSP.

PoolID = "poolID";
SubProcID = "subProcID";
TaskID = "taskID";
IncomingEdgesIDs = "flowInID_1, ..., flowInID_n";
OutgoingEdgesIDs = "flowOutID_1, ..., flowOutID_m";
TaskIDTransparancy = "traspValue";

\[
\begin{align*}
\text{var TaskIDTransparancy = "traspValue";} \\

\text{TaskID}(\text{taskId}, \text{flowInID}_1, ..., \text{flowInID}_n, \\
\text{flowOutID}_1, ..., \text{flowOutID}_m, \text{subProcID}, \text{poolID}) = \\
(\text{|| enter.flowInID}_1, \text{poolID} \; || \; ... \; || \; \text{enter.flowInID}_n, \text{poolID}) \rightarrow \\
\text{enable.taskID.poolID} \rightarrow \text{work.taskID.poolID} \rightarrow \\
(\text{esc.flowOutID}_1, \text{poolID} \; || \; ... \; || \; \text{esc.flowOutID}_m, \text{poolID}) \rightarrow \text{Skip;}) \\
\text{□ (terminate.poolID} \rightarrow \text{Skip;}) \\
\text{□ (terminate.poolID.subProcID} \rightarrow \text{Skip;})
\end{align*}
\]

Figure 7.5: Mapping Rules From BPMN Task to CSP.
Figure 7.6: Mapping Rules From BPMN Task Sending to CSP.
PoolID = "poolID";
SubProcID = "subProcID";
TaskID = "taskID";
IncomingEdges = "flowInID_1, ..., flowInID_n";
OutcomingEdges = "flowOutID_1, ..., flowOutID_m";
IncomingMessages = "msgInID";
CreTypeMsgID = "CreEIID";
ProTypeMsgID = "ProEIID";
ReTypeMsgID = "ReEIID";
TaskIDTransparancy = "traspValue";

var TaskIDTransparancy = "traspValue";

ReceiveTaskID(taskID, flowInID_1, ..., flowInID_n,
flowOutID_1, ..., flowOutID_m, subProcID, poolID) =
( (enter.flowInID_1.poolID || ... || enter.flowInID_n.poolID)
→ enable.taskID.participantID
→ channelTypesmsgInID?msgIn { CreTypeMsgID = CreEIID,
ProTypeMsgID = ProEIID, ReTypeMsgID = ReEIID }
→ work.taskID.participantID
→ (esc.flowOutID_1.poolID || ... || esc.flowOutID_m.poolID) → Skip; )
□ (terminate.poolID → Skip; )
□ (terminate.poolID.subProcID → Skip; )

Figure 7.7: Mapping Rules From BPMN Task Receiving to CSP.
experts may not have enough skills in formal languages so in general they could
not be able to describe such requirements using a given formal notation. Here
our contribution is on the codification of such domain knowledge within a tool
defining a set of property templates that should be satisfied by any process in the
given domain. Such templates constitute also a checklist that can be used during
the design of any BP.

Certainly the codification of a quality framework, i.e. the derivation of the
property templates, requires the collaboration among domain and IT experts. De-
pending on the domain requirements, on the mapping rules, and on the used model
checking tools the codification of the framework can be exploited according to dif-
ferent approaches. In order to actually verify some properties, it is possible that
it could be necessary to directly intervene on the mapping rules to add statements
expressly related to the domain requirements or to verification functionalities.

This is what we did in our case. Thus the derived checklist hides a set of
assertions that can be assessed thanks to the addition of global variables within
the mapping rules. Each mapping rule influences the verification of a property
redefining a global variable that is successively combined with other global vari-
ables to check the whole assertion.

7.2.3 Verification

The verification phase is based on model checking techniques. Reachability anal-
ysis is applied in order to assert whether the goals that are generated from the
properties specification are fulfilled or not. The model checker will apply a search
algorithm to repeatedly explore unvisited states until a state at which the condition
is true is found or all the states will have been visited.

It is worth mentioning that for the BPs that can be typically found in the e-
government domain the approach does not suffer from the state explosion phe-
nomenon as will be shown by some experimental data we already collected in
Section 7.4. In very general terms this problem relates to the huge, possibly in-
finte, number of states that even a simple model could generate. The result is
that when this phenomenon appears an exhaustive exploration of the state space
becomes infeasible or too much expensive. To the reduction of such a risk can ob-
viously contribute the definition of the mapping. In our case one of the choice that
we took to mitigate such a risk refers to the fact that for the purpose of reachability
analysis the data can be often ignored or mapped to small finite sets. Moreover
we also introduced some design constraints and we restricted the expressiveness
of BP diagrams. Nevertheless in different domains this could not be the case and
the BP developer will have to take into account this possible hurdle to the appli-
cability of this kind of formal verification technique.
7.3 BP4PA: A Tool Chains for Business Process Verification in the e-Government Domain

As underlined in the introduction one of the main problem in the e-government field is the scarce usage of services. We are aware that different reasons lead to such a situation but we believe that to promote services usage, delivery related processes should be carefully evaluated in line with specific quality requirements. Given the complexity and the heterogeneity of e-government applications a only human-based investigation on BP characteristics is not enough to guarantee high quality level. Instead this domain seems to have reached a maturity level in which more systematic and standardized techniques can be deployed. The approach we propose fits in such a scenario and it can contribute to improve the effectiveness of e-Government Digital Services. In particular it provides a tool-set for the application of formal verification techniques, such as model checking, to assess quality of services delivery related processes.

The remaining part of this section is structured as follows. Section [7.3.1] describes our tools chain where Section [7.3.2] describes the BP4PA framework focusing on issues related to properties specification in the specific domain of the service delivery processes.

7.3.1 Implementation Details for the BP4PA Plug-In

The formal verification approach illustrated in this paper is supported by a plug-in available for the Eclipse Framework that can be freely downloaded at the BP4PA web page (http://bp4pa.sourceforge.net/index.html). The plug-in permits to have a fully integrated and user friendly environment which supports domain experts both in the BP specification phase, and in the verification phase. In particular our plug-in is integrated in an Eclipse extensions such as the BPMN modeler and use the functionalities of the PAT model checker [196]. Thanks to the scalability of the Eclipse platform and the facilities provided by the Eclipse Modeling Framework we enrich BPMN modeler with novel properties and we also developed specific functionalities needed to implement the mapping from BPMN to CSP and from quality requirements to assertions. The PAT model checking that is available with our plug-in is automatically invoked to support the verification phase. Figure [7.8] reports the various components included in the BP4PA tool putting them in relation to the various elements described in Figure [7.1].

Potential users of our plug-in will start specifying the BP they need using the Eclipse BPMN modeler. At this point they will be able to start the process verification step selecting the quality requirements they would like the specification would satisfy. In order to carry on the verification BPMN2CSP will generate at
first a text file containing the CSP processes derived from the BPMN specification applying the mapping we have defined. The CSP processes will also contain the necessary information enabling the verification of the specified quality properties according to the defined mapping.

After the BPMN2CSP mapping is applied the tool will generate a text file containing the description of the target CSP processes representing the BPMN specification. Such file input of the PAT model checker via the mapping we have codified and the selected quality requirements will be mapped in assertion that enrich the CSP input file. After the CSP model is generated it is successively maintained synchronized with the corresponding BPMN model even when this is edited by the user. Quality requirements, defined by a PA domain expert, can be checked one after another or all at the same time. The resulting specification (related CSP and selected goals) will be than automatically provided to the PAT model checker that perform the verification and returns the results of one or more assertions. For each assertion the result is parsed by our plug-in that provides it back to the users. In particular, for each quality requirements the reached level is notified to the user as well as a flag implements an user-friendly alert.

Following we report some details about BPMN modeler and PAT model checking.
CHAPTER 7. QUALITY OF DIGITAL SERVICES IN E-GOVERNMENT

BPMN Modeler

The BPMN Modeler is a Business Process diagram editor for business analysts it was founded in 2006 by Intalio, Inc. It was developed as a component of the SOA Tools Platform and evolved as a subproject in 2008. It is based on an EMF object model bound to a graphical notation via the Eclipse Graphical Modeling Framework project.

The BPMN Modeler uses a light and flexible object model. It strives to achieve the look and feel of the BPMN visual notation rather than force a schema that fully describes a specification. The simplicity of the object model minimizes the impact when the specifications evolves. It also keeps the size of the generated code maintainable.

Possible usage and extensions of the modeler.

- Create BPMN diagrams to document process orchestration or workflow. Generate org.eclipse.stp.bpmn EMF objects.
- Traverse, annotate, transform to generate BPEL or other object models. Extend the editor to support drag and drop and other application specific usage.
- Implement a particular version of the BPMN specification: add the properties, validation services and generation algorithms.
- Create another domain model and map it to the notation already provided.

PAT model checking

Process Analysis Toolkit (PAT) is design to apply state-of-the-art model checking techniques for system analysis [196]. It supports reachability analysis, deadlack-freeness analysis, full Linear temporal logic model checking, refinement checking as well as a powerful simulator. It is a user-friendly model checker for Windows users.

Starting from PAT 2.0, the authors applied a modularized design to support the analysis of the different systems/languages. The Figure [7.9] shows the architecture design of PAT. Each language is encapsulated as one module with predefined APIs, which identify the (specialized) language syntax, well-form rules as well as (operational) formal semantics, and loaded at run time according to the input model. After parsing, the input model is built into the internal representations of the target module. The internal representation’s execution (operational semantics) are based on Labeled Transition Systems, which can be automatically explored by the verification algorithms (shared by all modules) or used for simulation. If there is any counterexample is identified, then it can be animated in the simulator.
This architecture allows new languages to be developed easily by providing the syntax rules and semantics. Till now, three modules have been developed, namely Communicating Sequential Processes module, Real-Time System module and Web Service module. In the future, PAT’s targeted systems include distributed systems, sensor network, UML (state chart and sequence diagrams), security domain (security protocols) and so on.

The main functionalities of PAT are listed as follows.

- User friendly editing environment for introducing models.
- User friendly simulator for interactively and visually simulating system behaviors; by either random simulation, user-guided step-by-step simulation, complete state graph generation, trace playback, counterexample visualization, etc.
- Easy verification for deadlock-freeness analysis, reachability analysis, state/event linear temporal logic checking (with or with fairness) and refinement checking.
- A wide range of built-in examples ranging from benchmark systems to newly developed algorithms/protocols.

7.3.2 Specification and Verification Issues

Imagine being in a local administration where promoting citizens inclusion is one of the main goal. e-Government business analysts should continuously review
processes to be sure that they are in line with quality requirements. Nevertheless it is often the case that within these offices the employees do not have high skills in ICT methodologies and techniques and they will carry on the quality checking mostly by hands. In this context our approach supports e-government business analysts, with an easy to use BP development environment including quality verification. Moreover we drive analysts in the process specification providing suggestions and defining constraints for the design phase. In particular for what concerns the BPMN modeler we have introduced a set of constraints, in line with those described in Section ??, that impose some additional rules on the Business Process modeling phase. As said this constraints aim at making the verification phase more effective and at making the specification less ambiguous. The inclusion of such additional constraints is made possible by the extensible structure of EMF which permits to semantically enrich the managed data structures corresponding to the developed models. In particular such constraints will permit to specify that:

- Every time a message is introduced in the process we request, both to set up the message type, and to specify if the message informs the citizen on the availability of additional useful services. The first constraint is needed to check coordination level and the second one for control level.

- Every pool implementing the role of the citizen has to be typed with user characterization. This is requested to check the control requirement. At the same time pool can be typed with the mediator role, to implement process tracing and activity aware transparency.

- It is possible to characterize a pool element as a database, such pool will receive messages from other pool/task supporting the storage of the information enabling their reuse

- All tasks have to be enriched with the transparency characterization. This is needed to verified the activity aware transparency checking.

With reference to quality requirements modeling we decided to provide a list of properties, in the form of a checklist, based on the BP4PA framework which has been introduced and illustrated in [44]. The BP4PA framework permits to specify different level of quality for characteristics such as those of coordination, control, sharing, transparency and inclusion. Having defined the quality requirement framework our effort has been focused on providing a mapping to goals to be satisfied by Business Processes related to e-government services delivery. In the following we discuss how the goals are used and checked at design time on BP under development.
CHAPTER 7. QUALITY OF DIGITAL SERVICES IN E-GOVERNMENT

The coordination quality requirement predicates over the interactions among Public Administrations involved in the delivery process. In particular with coordination we mean the capability of two or more Public Administrations to work together in order to accomplish a common goal and thought the usage of ICT technologies. To check coordination levels we assume that message exchanges are explicitly typed. Three different levels of coordination have been identified: communication, collaboration and semantic integration. Our verification on coordination characteristics will conclude that the service reach the quality goal linked to the coordination level specified by the user if interactions among the involved administrations are implemented via ICT technologies and the sequence of messages exchange fits or is higher than the specific coordination goal. In other words we require interactions among participants and we check the type of messages exchange. If different part of a process satisfy a different type of interactions the lower level of quality is considered and used to rank the Business Process. Lack of coordination is observed only in case of more than one participant contribute to the Business Process without implementing messages exchange. Figure 7.10 shows how to check the coordination level on a simple Business Process sung the BP4PA interface. In such case two interactions are implemented via messages exchange typed with communication and collaboration levels respectively. Results of the quality check show that the process fit in the collaboration level respect to the coordination quality requirement.

Figure 7.10: Coordination Check: Collaboration.

The control quality requirement predicates over the first interaction in the BP among administrations and citizens. In particular with control we refer to the paradigm applied to drive the GDS delivery from its start to its final fulfillment.
In the analysis phase the resulting control level can be reactive or proactive. Our check on control concludes that the service reaches the quality goal linked to the specific reactive control level if the first message goes from the citizens toward the PA. Otherwise, if the first message is incoming to the citizens pool the service is recognized as proactive. There is also the case where services are both reactive and proactive (two parallel process are implemented in the citizen pool with separate start and end event), so in this condition no control is observable. In other word we check the order in which the interactions among PA that participate to the services delivery and citizens are implemented. We also consider the possibility to implement a creative service distribution. This characteristic refers to the promotion of related, and maybe relevant, services. Creativeness can be provided as an enrichment on the reactive and proactive way of implementing services delivery control. In this case we check if there is a message characterized with creative feature going from PA to citizen. To provide a simple examples on the control level we refer to the cases in Figure 7.11, Figure 7.12 and Figure 7.13. In such cases we show a proactive, a reactive, and a process which does not satisfy any level of control respectively.

**Figure 7.11: Proactive Control Level**

The *sharing quality* requirement predicates over the way in which the PA handles and shares citizen data with other administrations in order to participate in the delivery of a specific GDS. To check sharing levels we assume that a specific participant typed with knowledge repository role is present in the service delivery. Our verification on sharing concludes that the service reaches the quality goal linked to the specific sharing level if such role is included in the process and other participants interact with it. In other words we check that the type of participants included in the delivery matches this specific role and we request interactions with it via messages exchange. Figure 7.14 shows how sharing is checked on a sim-
CHAPTER 7. QUALITY OF DIGITAL SERVICES IN E-GOVERNMENT

Figure 7.12: Reactive control level

Figure 7.13: No Specific Control Level
ple Business Process via BP4PA user interface. In this case the requested pool is included and interactions are observable.

The transparency quality requirement predicates over the visibility of the business process put in place by the Public Administrations involved in the service delivery. With transparency we mean the ability of the administration to make citizens aware of the delivery process and of its execution state, improving in this way citizens’ perceived trust. To check coordination level we assume that activities involved in the process delivery are typed with transparency characteristic when they are observable. Lane can be also included in the process specification when roles in charge to complete such activity are clearly defined. This two requirements fit with the the levels of transparency recognized in the analysis phase, that are activity aware and role aware respectively. Our verification on transparency conclude that the service reach the quality goal linked to the activity aware transparency level if all tasks involved in the process delivery in charge of the involved administration are traced and visible to the citizen. At the same time if also lanes are included in pools that are not citizen, mediator or knowledge repository, then the role aware specification is observable after checking. In our modeling we consider a further way of implementing and recognizing activity aware transparency. In this case we assume the availability of a particular pool called mediator specifically devoted to trace the state of the executed process. Our verification on transparency concludes that the service reach the quality goal linked to the activity aware transparency if the mediator pool is included into the BP and tasks give specific feedbacks to the mediator via messages exchange. Nothing may be assumed on the role aware transparency in case the mediator is included. To provide simple examples on the transparency level we refer to the
The *inclusion quality* requirement predicates over the process of the involved Public Administrations in the delivery. In particular with inclusion we mean the ability of the administration to provide service to citizens also considering their possible disability. To check inclusion levels we assume that gateway typed with level of inclusion are observable in the process. Such gateway concerning possible interaction channels, user profiles and internationalization can be identified during the analysis phase by domain experts. Our verification on inclusion concludes that the service reach the quality goal linked to the specific inclusion level if in the process is allowed to switch between profiles, languages and channels respectively. To provide a simple examples on the inclusion level we refer to the cases in Figure 7.16 where all the three inclusion levels are introduced.
7.4 BP4PA in Use: Case Studies

This section illustrates how the proposed approach can be used in practice on a real case study. This is the result of a close cooperation between our research group and a local Public Administration where the proposed approach is currently applied in practice and tested on real processes. As for any approach using model checking techniques it is worth mentioning that so far we experimented with 35 different processes and all of them have generated relatively small state sets. In particular the experiments we have conducted using a desktop PC equipped with a Core 2 Duo 2.20GHz and 4GB RAM, have highlighted that a process can be checked with respect to the properties included in the framework in less than 3 hours, for the most complex BP scenarios. Moreover the most complex BP we have analysed so far generated a state space of around three millions states. This data seems to support the idea that in the current status (i.e. complexity of BP processes in the e-Government domain, mapping we have defined and quality properties to be checked) the approach is applicable in real scenarios and can be a useful support for the BP designer.

The services under analysis refers to the newborn registration service and moving service. The participants involved in such process are following reported and shown in Figure 7.17.

- The municipality where the baby has to be registered.
- The Home Affairs Minister (Ministero degli Interni) is deputed to collect and to maintain up to date the information related to citizens. To carry on this task it implements two different infrastructures:
  - SAIA is the technological infrastructure used by all the Italian Public Administrations to support information exchange concerning citizens data;
  - INA is a national knowledge base system which contains information related to Italian citizens. In particular it contains family name, first name, fiscal code, gender, place of birth, date of birth and code of the municipality where the citizen lives.
- Tax Office, in Italian “Agenzia delle Entrate”, is the national organization in charge of issuing the Italian tax code card, officially known in Italy as *Codice Fiscale* (similar to a Social Security Number card in the United States). The card serves to identify, unambiguously for tax related purposes, individuals residing in Italy
- CISIS (Italian inter-regional center for information, statistical and geographical systems) is the association of regional authorities which, among a list
CHAPTER 7. QUALITY OF DIGITAL SERVICES IN E-GOVERNMENT

of several activities, has to collect all the information requested for statistical purposes.

- INPS is the national institute for social insurance. In Italy the welfare insurance is organized by the state.
- Transport Office, in Italian “motorizzazione civile”, is the national organization in charge to issues the driving licence.

In the following we discuss about specific process and their evaluation.

7.4.1 Newborn Registration Service

The GDS under analysis refers to the newborn registration service, which is part of the wide area of cooperation among civil registration services (which are managed locally by municipalities). The service intends to permit to new born citizens to be registered, to get certificates and any other service regardless of their geographical location. In particular the service under analysis supports in the most comfortable way the registration of a baby’s birth delivering at the same time the request for the birth certificate and for the fiscal code number. Such service supports the alignment of the information in all the Public Administration offices dedicated to trace new born babies. The participants involved in such process are the municipality where the baby has to be registered, Home Affair Minister, the Tax Office and CISIS. The service is implemented as a six-steps process, in particular:

- As first step the parents ask for activating and accessing the new born registration service. It is worth mentioning that the access can be provided at the municipality office or via Web when suitable authorizations and authentications mechanisms are set.

- The municipality collects the birth registration information from the parents.

- The municipality sends such data to the Minister via the SAIA infrastructure.

- The Minister communicates the necessary data to the Tax Office that generates and returns a new Tax code.

- The Minister stores all the received information, concerning the citizen, within the INA repository.

- The Minister communicates the data, relevant for statistical purpose, to the CISIS.
In order to use the proposed approach a domain expert will have to codify a BP for the GDS he/she wants to put in place and to verify. In our case we considered the national specification for the New born registration service as it was codified in BPMN by the local PA as shown in Figure 7.18. Successively the BP4PA tool can be run and it will return the different quality levels the current implementation of the BP fulfills. In particular for the new born registration service BP4PA provided the results as below.

- Concerning the coordination dimension the communication level was reported. The BP does not foresees any semantic integration among the partners and the back offices where able to interact requiring some level of human intervention. This is consequence of the usage of the e-mail system as communication channel among civil servants.

- Concerning the control dimension a reactive level was reported. No integration with other services concerning new births are implemented.

- Concerning the sharing dimension the data sharing level was reported. This level is reached thanks to the facilities provided by the INA-SAIA infrastructure.

- Concerning the transparency dimension no mechanisms were implemented. The citizens are not in any way notified about the proceeding of the process.
and they cannot observe the execution of any step in the process.

- Concerning the inclusiveness dimension no condition were checked regarding specific capabilities or disabilities of the citizen.

The analysis conducted using BP4PA permitted to systematically discover many issues hided within the BP specification, that would have been resulted in possible low service usage. At the same time the issues highlighted by the tool have been considered by the design team and domain experts in order to revise and improve the service delivery process trying to fulfill, when possible, higher quality level.
CHAPTER 7. QUALITY OF DIGITAL SERVICES IN E-GOVERNMENT

Figure 7.18: Case Study: Newborn Registration Service.
7.4.2 Moving Service

The service under analysis refers to the change of address service. It is involved in the wide area of cooperation among civil registration services (which are managed locally by municipalities) in order to allow citizens moving from an address to another.

In particular the service under analysis support the easiest way to register the new address and guarantee the change on other documents such as drive license. The service supports the alignment of the information in all the Public Administration offices dedicated to trace the citizens. The participants involved in such process are the municipality where the citizen want to move, Tax Office, Home Affair Minister, CISIS, INPS and Transport Office. There is also the case that the sanitary office has to be involved to enable compensation about sanitary data when the new address is in a different area of competence respect to the original regional sanitary system.

The service is implemented as a six-steps process, in particular:

- As first step the citizen asks for activating and accessing the moving service. It is worth mentioning that the access can be provided at the municipality office or via Web when suitable authorizations and authentications mechanisms are set.
- The municipality collects the information from the citizen.
- The municipality sends such data to the Minister via the SAIA infrastructure.
- The Minister communicates the necessary data to the Tax and Transport Office and INPS so that citizens data are updated.
- The Minister stores all the received information, concerning the citizen, within the INA repository.
- The Minister communicates the data, relevant for statistical purpose, to the CISIS.

According to our approach first we design the service business process using BPMN Modeler as shown in Figure 7.19. Once the verification is started the different requirements are observable in the process. In this case communication level of communication, reactive level of control, data sharing level of sharing are reached and no transparency is implemented. So also in this case many improvements can be implemented to the defined process as below.

- For what concern the coordination, the administrations involved in the service delivery should implement an on-line system of coordination or implement policies enabling the fully interaction of the back-end. Also in this
case the delivery of the service using the national framework of cooperation SPCoop has to be considered.

- For what concern the control level the reactive service level is good enough. In this case the administration can not know in advance when the citizen want to move.

- According to the sharing level the national initiative of INA-SAIA implement in a good way the ability of Public Administration to contribute to the citizens needs and satisfied the request using shared knowledge bases for what concern sensible data of the citizen.

- Transparency should be implemented according to the process of service delivery, provide to the citizens check points on the service.

Feedbacks resulting from the application of the proposed approach show its goodness in practices even if the constraints imposed during the BP design make the modeling a non trivial task.
Figure 7.19: Case Study: Moving Service.
7.5 Useful Approaches on Applying Formal Methods

In Business Process Management Business Process modeling has to provide sound behavioral structure and consider constraints like domain dependent quality requirements. To do that considerable attention has to be given to model checking as practical approach of systems verification. It concerns with determining in advance whether a process model exhibits certain desideratum behaviors. By performing this verification at design time, it is possible to identify potential problems, and if so, the model can be modified before it is used for execution. Interesting aspects of BPM such as BP verification are still well un-explored as needed. This is described by Jan Mendling despite a lot of work on the topic proved by popular and academic textbooks as well as international professional and academic conferences such as BPM [131].

In this section we introduce the application of formal modeling and verification in e-government and BPM area. We present a detailed survey of existing proposal for formal verification techniques of Business Process and we compare them with respect to the formal semantics required to specified Business Process. The last Section is dedicated to service oriented architecture verification as an influent area of research.

7.5.1 Formal Models in e-Government

Although formal methods for development and design information systems are fully available for a long time in term of languages, proof and tools, few of them have significant impact in practice. A major barrier to the use of formal model in practices is that the user finds formal methods difficult to understand and apply. However their potentiality is huge and an appropriate application could be a suitable solution toward the efficient and effective Information Systems.

To drive the development of the e-government, the application domain of our interest, it is essential that governments, or their agents, should be able to produce un-ambiguous specifications of functionalities. This kind of specification requires an approach to modeling known as formal engineering techniques. Informal, imprecise use of graphical notations is needed and should be as much user friendly as possible to make such environment and approach accessible. However, an user friendly notation is not enough. The meaning of the specification must be clear and irrefutable.

The first tentative to introduce formal techniques as result of the application of mature areas of the science and technology to e-government is discussed in [47]. It is the result of a tutorial-workshop event on Technological Foundations
of Electronic Governance. The event explores the relevance and opportunities for the application of mature formal techniques based on mathematical theories and supported by industry-ready tools and methods to build technical solutions for e-government.

In the workshop different formal techniques are discussed as different scenarios to build clear understanding of domains and to generate implementations from abstract specifications. They refer to domain description, systems specification, specification-based testing, behavioral specification, verification and generating implementation as possible areas to explore formal techniques. So each of this techniques can be fruitful implemented and used in e-government to reach specific domain dependent objectives and solve e-government challenges such as privacy protection, identity management, collaboration and interoperability of government IT ecosystem just to cite a few.

In the workshop four papers which provide concrete examples of how formal techniques are able to address specific challenges in e-government domain have been discussed. They refer to transparency verification [33], risk assessment [56], automatic form generation [191] and semantic frameworks for e-government [45].

Workshop concludes that the main issues to solve in such domain is making formal techniques accessible to the wide amount of users and systems designers providing an user friendly environment able to demonstrate the goodness and the efficacy of such techniques.

### 7.5.2 User Friendly Model Checking

From the user-friendly point of view few attempts of model checking integration in the Business Process modeling are already discussed. They are interesting approaches on making model checking accessible to a large audience even for people that are not trained in formal techniques. A recent survey on Business Process verification provides an interesting classification [139]. Here below we report the main efforts made in such direction enriching the result of the cited survey. Such papers mostly refer to a user-friendly process specification languages such as BPMN and UML activity diagram. Concerning the formalization they can be mainly classified in three categories: automata [69], Petri Nets [227] [50] [142] [85] [236] [232] and process algebras [177] [62] [193] [245] [229] [104] [67].

For what concern automata Fu et al. propose a framework to analyze and verify properties of BPMN diagrams converted into the BPEL format that communicate via asynchronous XML messages [69]. The framework first converts the processes to a particular type of automata whose every transition of which is equipped with an XPath format guard, and then these guarded automata are translated into Process or Protocol Meta Language for the SPIN model checker [96]. Consequently, SPIN can be used to verify whether Business Process models
satisfy properties formalized in LTL.

More interesting works have been done with Petri Net that often become hot topic in BPM concerning capturing process control flows [227]. Petri Net can specially detect the dead path of Business Process models which preconditions are not satisfied. Dijkman shows how to correspond all BPMN diagrams constructs into labeled Petri Net [50]. This output can subsequently be used to verify BPEL processes by the open source tools BPEL2PNML and WofBPEL. Another contribution is given by Hamadi and Banatallah. They apply a Petri Net based algebra to modeling Business Processes based on control flows [85]. However the most interesting approach in this area is given by the YAWL community. Yet et al. propose a formal semantics of BPMN in terms of a mapping to YAWL nets, for which efficient analysis techniques already exists [236]. The proposed mapping has been implemented as a tool that generate code in the ProM. Recently a similar work has been published by Wynn et al. [232]. It focuses on two main features common in any modern process modeling language such as cancelation and or-join. The paper demonstrates that process verification is mature enough to be used in practices. Other interesting mappings from BPMN to Petri Net are discussed in the literature, see for instance [237].

In the area of process algebra interesting works have been done applying different approaches. Notable is the work of Salaun et al. that discusses the application of process algebras to describe, compose, and verify Business Processes, with a particular focus on their interactions [177]. The authors show an example in which they use CCS to specify and compose Business Processes. They also use the Concurrency Workbench to validate properties such as correct Business Process composition. It may solve real issues, e.g., the exchange of messages during Business Process interactions. Others attempts has been investigated to use CCS as the most natural, and intuitive, way in which it represents process dynamics [62] [193] [245]. However, the most comprehensive approach related to our approach is discussed by Wong and Gibbons in [229]. The authors demonstrate how the process algebra CSP also can be applied to model complex workflow systems. The authors have given a formal semantics for BPMN in CSP using Z notation [231], and use it to formally check compatibility of BPMN process [230]. Unfortunately, the discussion does not consider the messages exchange and participants involved in the process neither implement domain dependent properties verification.

Other interesting work have been already done in a wide area of Business Process verification whit a mapping on process algebra. In the same area Janssen et al. show how model checking can be applied in the context of business modeling and analysis by people that are not trained for informal techniques [104]. SPIN is used as the model checker underlining a graphical modeling language such as UML. In this case requirements are specified using business requirements pat-
terns, which are translated to LTL. A more recent work published by Forster et al. describes how model checking can be employed for formal verification of Business Process against process patterns based on specialized activities [67]. UML is considered as a single language for specifying both the Business Process and the corresponding constraints. In this case, the transition system generated by GROOVE is automatically translated into the input language of NuSMV model checker.

As already showed literature proposes some interesting approaches that tried to put together Business Process and formal verification. All of them focus on Business Process behavior observing structural and compositional problems but skipping peculiar challenges presented by specific application domain generally absent or less preeminent in other kind of applications.

### 7.5.3 Services Composition Verification

To complete our discussion related works in the area of workflow modeling and verification has to be referred. These approaches use model checking as verification technique. Later we cite the most interesting papers in this areas of automata, Petri Net and process algebra.

In the automata context the following proposals are notable. The work done by Diaz et al. is an interesting case, the authors provide a case study to convert automatically Business Processes written in BPEL-WSCDL to timed automata and to verify subsequently them by the UPPAAL [49]. The authors are currently implementing a tool for the automatic translation. Moreover, Dong et al. propose a framework to verify automatically Business Processes that are modeled in Orc [51]. The authors define a formal timed-automata semantics for Orc expressions, which verifies to the Orc’s operational semantics via UPPAAL model checking. The paper also shows a simple case studies. Koehler et al. discuss a pattern-based modeling [110]. Starting from a Business Process model, which emphasizes the underlined structural process pattern and its associated requirements, the proposed approach map this model into a corresponding IT model based on nondeterministic automata with state variables. Model checking techniques are used to automatically verify elementary requirements on a process such as the termination and reachability of states. Finally, Pu presents an operational semantics for a subset of the BPEL4WS language, which is then mapped onto a network of timed automata, verified using UPPAAL [165].

Concerning Petri Net we present some works below. Yi and Kochut proposes a Petri Net based design and verification tool for web service composition [238]. The tool can visualize, create, and verify Business Processes. Another approach is discussed by Zhang et al. They introduce a Petri Net based architectural description language, named WS-Net, in which Web Service oriented systems can be modeled [244]. In the same area Hinz et al. propose a Petri Net semantics for
BPEL which assures exception handling and compensations [92]. Moreover, the authors present the parser which can automatically convert BPEL specification into Petri Net. Consequently, the semantics enabled many Petri Net verification tools to automatically analyze Business Processes. Petri Nets are also used by Schlingloff et al. to give a semantic for BPEL4WS [181]. The net resulting from the translation is then validated with the LoLA model checking tool. Finally, Dun et al. present an approach to model and verify BPEL based on ServiceNet (special class of Petri Net) [52].

Concerning the process algebra oriented service verification a lot of interesting works can be cited. Here we report just few of them. Ferrare defines correspondence between BPEL and LOTOS [63]. The advantage of this proposal is that it includes compensations and exception handling. Thus, it enables the verification of temporal properties with the CADP model checker. Bianculli et al. present an approach for the formal verification of workflow based composition of web service, described in BPEL4WS [25]. Workflow process can be verified in isolation, assuming that the external services invoked are known only through the interface. Deadlock freedom, properties expressed as data-bound assertion written in WS-CoL are verified as well as LTL temporal properties. In this case Bogor model checker has been used by the authors. Process algebras are used also by Koshkina [111]. The BPE-calculus, is used to abstract BPEL4WS control flow. This calculus is used as input for a process algebra compiler to produce a front-end for the Concurrency Work Bench, in which equivalence checking, preorder checking and model checking of processes are performed.

Finally, focusing on model checking approaches we can refer to the following contributions. Nakajima proposes to use the software model-checking technology for the verification of the Web service flow descriptions [141]. The paper adapts Web Services Flow Language as the language to describe the Web service flows, and uses the SPIN model checker for the verification engine. Moreover, Web Service Analysis Tool a framework for analyzing interaction of composite web services is presented [69] [70]. In this case the interactions of composite web services are modeled as conversations, keeping track of exchanged messages. BPEL4WS specifications of web services are translated into an intermediate representation, an XPath-guarded automaton augmented with unbounded queues for incoming messages. This model is then translated into Promela and LTL properties, which can be also derived from XPath expressions, are checked with the SPIN model checker. Finally, the Verbus verification framework is a modular and extensible framework for the verification of Business Processes [8]. Thanks to an intermediate formalism, the framework is not tied to specific process definition languages or verification tools. The support for the BPEL4WS language is partially complete. In this case missing constructs are the compensation activity and event handlers.
CHAPTER 7. QUALITY OF DIGITAL SERVICES IN E-GOVERNMENT

7.6 Considerations

In this Chapter a formal contribution in e-government has been discussed. We have shown how it is possible to use formal techniques to conduct the analysis and then to assess the e-government digital service. In particular the approach we propose permits to automatically check if a designed delivery process satisfy defined quality properties or instead if it suffer of some degrading property. The general idea has been also concretely implemented in a tool set for the eclipse platform. This will permit to have an integrated development environment in which to model and evaluate an under development business process.
Part IV

Quality Aggregation
Chapter 8

Aggregation Model

This part of the thesis was published in [R1] [L1] [C2] [C4].

8.1 Introduction

In this Chapter we introduce our mathematical model for quality assessment of e-government digital services. After three normalization phases, the model provides the assessment of the quality level. Starting from a set of quality parameters the model estimates the quality as value in the $[0...100]$ range. The main advantages of using a mathematical model are that we have a description of the problem at a high level of abstraction and that we have a formal background on which the service development can be based, so to avoid possible structural mistakes and inaccurate descriptions. Moreover, the model constitutes a firm basis on which objective parameters are placed and on which subjective parameters can be treated in a controlled way.

This model is inspired to that of [122], but we introduce further elements like data homogenization and interaction between parameters. In more detail, (i) homogenization of the input is useful to reason over different e-government parameter metrics and measurement techniques. The homogenization takes also into account whether a given parameter grows in a proportional or inverse proportional way with respect to the overall quality measurement. For example, infrastructure related parameters measured against time needs to be aggregated with security parameters measured with boolean values (or some other metrics). At the same time, we introduce (ii) interaction among parameters to measure dynamic relationships. Using this, we can take into account how parameters influence each other (for instance, usability influences service trust).
8.2 Terminology

Let $S$ be a service. The mathematical model uses the following input parameters.

- $Q = (q_1, ..., q_n)$ is an array of $n$ natural numbers representing the measured value of parameters related to the service $S$. Each $q_i$, $1 \leq i \leq n$, is collected during a measurement process and represents a specific view of the service.

- $Z = (z_1, ..., z_n)$ is an array of $n$ boolean values used in the normalization phase. Each $z_i$, $1 \leq i \leq n$, takes its value as follows:

$$
z_i = \begin{cases} 
1 & \text{if the } q_i \text{ parameter in } Q \text{ grows in proportional way with respect to the overall quality value} \\
0 & \text{otherwise} 
\end{cases}
$$

- $C = (c_1, ..., c_n)$ is an array of $n$ positive natural numbers used during the homogenization phase. Each $c_i$ represents the upper bound of the $q_i$ parameter in the $Q$ array. The elements of $C$ are related to the measurement of parameters: they depend on the specific metrics used to express the parameters and on the methodology of the measurement.

- $I$ is an $n \times n$ matrix of values in the range $[0..1]$. It shows the interaction level between the parameters in $Q$. Each $m_{j,k}$, $1 \leq j, k \leq n$, takes its value as follows:

$$
m_{j,k} = \begin{cases} 
\text{a value in } [0..1] & \text{if } q_j \text{ and } q_k \text{ interact} \\
0 & \text{otherwise} 
\end{cases}
$$

Note that all the diagonal values of $I$ must be 0, i.e., each parameter has not relevant interaction with itself.

- $D$ is a $n \times l$ matrix of boolean values where $n$ is the number of parameters and $l$ is the number of quality groups. $D$ is used to group parameters with similar features. Each parameter can belong to one and only one group, i.e., the matrix must satisfy the following constraint.

$$
\forall i \ 1 \leq i \leq n, \sum_{j=0}^{l} d_{i,j} = 1
$$

We use $h_i$ to refer the cardinalities of the groups. Moreover, we use $H$ to denote the array of these $l$ values.
• \( W = (w_1, \ldots, w_l) \) is an array of \( l \) natural numbers in which each \( w_i, 1 \leq i \leq l \), is the weight of the group \( i \). The array must satisfy the following constraint: \( \sum_{i=1}^{l} w_i = 100 \). The weights can also be associated to parameters if and only if the groups are all singletons.

![Figure 8.1: QoS Assessment Model Phases](image)

### 8.3 Measurement and Derivation of Input Values

In this section we discuss the way in which the input values of the model should be obtained.

The values of objective parameters of array \( Q \) of the quality assessment model are estimated using (i) subjective parameters evaluation obtained by anonymous survey, (ii) run-time measures taken during the execution of the service, (iii) process related parameters evaluation obtained via service design verification. All the values of array \( Q \) are the average values of the measurements of each parameter in the set of data of the considered quality assessment. The values of array \( C \) are the maximum measured values of each parameter.

The values of array \( Z \) (that is the trends of parameters with respect to the overall quality), the values of matrix \( D \) (grouping) and the values of array \( W \) (weights of groups) are derived from the opinion of domain experts.

The values in \( \mathcal{I} \) are derived from the measurements as the absolute values of the statistical correlation coefficients between each pair of different parameters. These values describes the strength of the association between the two parameters.
and allows the definition of the interaction factors that will be used in the model. The use of the absolute value of the correlation is compensated, in the model, by the specification of the trends in array $Z$.

### 8.4 Aggregation Model

Now we present in detail the three phases of our model to determine the quality assessment. Figure 8.1 shows the phases to evaluate the overall service quality.

#### Phase 1: Data Homogenization

As a first step, let $Q$, $Z$ and $C$ be the input arrays. We introduce a function $f_1$ that normalizes the values, which are measured with different metrics. The function takes triples of the form $(q_i, z_i, c_i)$ – where $q_i$, $z_i$ and $c_i$ are the $i$-th elements of the arrays $Q$, $Z$ and $C$ respectively – and returns a value in the range $[0...100]$. Using this function, we obtain a new array $Q'$ of elements $q'_i = f_1(q_i, z_i, c_i)$. The formal definition of the function $f_1$ is as presented below.

$$f_1(q_i, z_i, c_i) = z_i \left( \frac{q_i \times 100}{c_i} \right) + (1 - z_i) \left( 100 - \frac{q_i \times 100}{c_i} \right)$$

#### Phase 2: Parameters Interaction

In the second phase we introduce the interaction factors of the quality parameters. The interaction factor increases the importance of the relative parameter in the overall quality assessment. We obtain the interaction factor $\varphi_k$, $1 \leq k \leq n$, as the mean of the values in the column $k$ of the matrix $I$.

$$\varphi_k = \frac{\sum_{j=1}^{n} m_{j,k}}{n-1}$$

The proposed interaction factor does not take into consideration recursive impact on parameters since $I$ is a matrix with null diagonal elements.

Each element $q'_i$ obtained in the first phase must be normalized again to obtain a new array $Q''$ whose elements $q''_i$ are calculated as presented below.

$$q''_i = \varphi_i q'_i$$

$\varphi_i$ and $q'_i$ are the $i$-th interaction factor and the $i$-th element of the array $Q'$ respectively.

The normalized values $q''_i$ encapsulate the information about the interaction between parameters. This allows the model to determine a quality assessment in
which the parameters are not considered as single items, but they influence each other in the particular experimentation context. This makes the quality assessment more accurate than a model in which the parameters are considered independent.

**Phase 3: Grouping and Group Weight**

At this point, we introduce the possibility of grouping the parameters in order to manage them as groups with different importance. We use the matrix $D$ to obtain a new array $G$ (its elements will be denoted by $g_1, g_2, \ldots$) of values for each group as presented below.

$$G = Q''D$$

Note that each element of $G$ is the sum of the values, in $Q''$, of parameters in the same group.

Finally, to give the overall quality value for the service we use the $QoSLevel$ function defined and reported below.

$$QoSLevel(G, H, W) = \frac{\sum_{i=1}^{l} g_i \cdot h_i}{\sum_{i=1}^{l} w_i}$$

$g_i$ and $w_i$ are the $i$-th elements of the arrays $G$ and $W$ respectively, and $h_i$ is the cardinality of $i$-th group. The function gives a value in the range $[0...100]$ since the denominator is always 100 and the average value of each group is weighed by the corresponding value in $W$.

**8.5 Experimentations**

To validate the goodness of the model we observe its behavior according to the different scenarios that can be more or less real.

**8.5.1 Quality Behavior**

First of all we introduce random values as input in the case proposed below.

- (a) We analyze the behavior of quality function observing the variation of quality parameters by fixing all the other model entities.

- (b) We analyze the behavior of frequency distribution of the quality starting from random values both for parameters and for other inputs of the model.

- (c) We analyze the quality frequency distributions underlying the role of parameters interactions (we observe bound behaviors).
CHAPTER 8. AGGREGATION MODEL

After several experiments we were able to assess the QoS trend. Experiments in item (a) show a linear trend of QoS value. It increases or decreases steadily with respect of the parameters trend. We take into account the properties of the parameters; some of them are proportional whereas others are inversely proportional with respect to the quality value (i.e., the quality increases if the execution time decreases and/or the usability increases). The behavior of frequency distribution of the quality values follows a normal trend. We observe this kind of behavior starting from random values both for parameters and for other inputs of the model in item (b). Taking into account the central limit theorem, the sum of large and independent quality observations has an approximate normal distribution (Gaussian Distribution) under certain general conditions. Finally, in item (c) it is clear that parameters interactions affect quality upper bound. If the interaction decreases the quality level assume low values (Figure 8.2), while with height interaction also the QoS values increase (Figure 8.3). Finally, we observe with low interactions a close quality frequency distribution, while it is stretched to a normal trend with high interactions. The increase of parameters interaction support the goodness of our approach; as matter of fact, the e-government process is influenced by different dependent factors.

![Figure 8.2: Quality Distribution Low interaction.](image)

Focusing on a more real scenario we analyze the quality frequency distribution based on 100 times quality evaluation consulting service at no regular time interval. The service under analysis is the residence certificate in the TecUt portal and
the experimentation is based on run-time measurement. TecUt is an ASP.NET Web Application running on IIS6 (Windows Server 2003 Environment). The Server is a Pentium IV 3.0GHz with 1Gb of RAM located inside the Marche Region Demilitarized Zone. In this case we explore the general behavior of the quality distribution. We calculate quality value 100 times consulting the data at no regular time interval. The frequency distribution of the obtained discrete overall quality values is reported in Figure 8.4. The average value $\mu$ is 52.20, the minimum value is 20 and the maximum value is 85. The standard deviation $\sigma$ is 11.49. We observe that about 82% of the quality values of our service are in the interval $[\mu - \sigma, \mu + \sigma]$. This means that the quality of the service is fairly stable. This result may be justified by the fact that the architecture in which the service is deployed is reliable and the network infrastructure supporting the service delivery guarantees constant performances. The remaining 18% more dispersive values depend on subjective parameters and on users’ skills. Values under $\mu - \sigma$ are determined by the evaluations of non-skilled users, while values above $\mu + \sigma$ are determined by the evaluations of expert users.

8.5.2 Dependences Among Parameters

According to the experimentation done in the real scenario we report below some experimental parameters dependences. They are based on mathematical correla-
Figure 8.4: Frequency Distribution of Quality

- In Figure 8.5 we provide the linear regression between time processing and time data retrieval. The correlation coefficient in this case is 0.0589. We conclude that time data processing is independent by time data retrieval.

- In Figure 8.6 we provide the linear regression between time execution and time data retrieval. The correlation coefficient in this case is 0.8875. We conclude that time execution impact on the time retrieval.

- In Figure 8.7 we provide the linear regression between time processing and time execution. The correlation coefficient in this case is 0.4909. We conclude that time processing influences time data retrieval.

8.6 Considerations

In this Chapter we have defined a formal assessment models useful to provide a description of a complex environment where the interactions between parameters play an important role. For our purposes, the defined model is satisfactory. It allows to carefully assess the quality of e-government services focusing on all
the different involved aspects. We recognize that the accuracy of the model depends on the accuracy of the validation and evaluation done by domain experts. It depends also on the accuracy of the subjective answers given by the users to the questionnaires. However, this is a general characteristic of the e-government domain, in which the subjective component is important as well as the flow of activities done by the Public administration. Positively, the choice of putting together a formal model and subjective evaluations allowed us to fix a clear basic formal frame for objective aspects of the quality assessment on which subjectivity can be introduced in a controlled way.
CHAPTER 8. AGGREGATION MODEL

Figure 8.6: Dependence Between Time Execution and Time Data Retrieval

Figure 8.7: Dependence Between Time Processing and Time Execution
Part V

Conclusions
Conclusions and Further Work

In this thesis we have provided a quality oriented view of e-government service delivery focusing on the application of formal techniques as a suitable way to support quality assessment. The quality model seems to be suitable to enable the improving of citizens service.

Hereunder we summarize the main achievement and than we discuss further possible extensions.

- In Chapter 4 we discuss the definition of a systematic view of quality in e-government services. Our contribution supports the definition of a quality model considering customer satisfaction, site quality, technical performance and process performance. Starting from a literature review and thanks to experience on this field we collect a set of quality requirements that refers to (i) how the quality parameters perceived by the customers against their expectations, (ii) the web site usability and interface quality characteristics, (iii) technical aspects of web site and (iv) quality aspects typically related to traditional government services.

- In Chapter 7 we introduce a methodology and a tool allowing to formally and automatically assess the quality of a designed Business Process respecting the defined quality requirements. This is a tentative to bind user friendly environment with formal verification techniques.

- In Chapter 8 we define an aggregation model suitable to have an unique view on quality. The main advantages of using a mathematical model are the description of the problem at a high level of abstraction and the introduction of formal background on which the service development can be based in order to avoid possible structural mistakes and inaccurate descriptions. Moreover, the model constitutes a firm basis on which objective parameters
are placed and on which subjective parameters can be treated in a controlled way.

Even though this thesis presents significant contributions to the field of quality of e-government service, there are still some open research problems that have not been contemplated, or that have come out as a consequence of the advances proposed in it. We conclude this section by mentioning some of them.

Concerning Chapter 4 more quality parameters can be considered according to quantitative and qualitative evaluation of e-government domain.

Concerning Chapter 7 the addition of new quality property to be verified on the processes is easily managed due to the flexibility of the current implementation of the approach. At the same time we will work to support the analysis of other PAs related processes. Given the organization and the resources availability in a specific public office other PA dimensions concerning process in place can be verified.

Finally our work will have to consider the influences that in the next future will have the upcoming release of the BPMN 2.0 specification and we will certainly have to adapt our approach to that notation version.

Concerning Chapter 8 first we would evaluate the proposed approach with a different and larger experimentation. At the same time, we should study complexity reduction algorithms for the proposed mathematical model. This kind of algorithm will be able to maintain the models’ expressiveness.

Finally, we intend to propose our approach as a mean to make a comparison among services (with the same functionalities) provided by several Public Administrations in term of benchmarking. In line with the guidelines discussed in [178] our proposal can be a first tool for practical benchmarking of PA digital services delivery process facing some important issues still unexplored by the current benchmarking studies [24]. Project out-sourcing is often the adopted solution within Public Administrations due to limited staff and budget. Our approach represents a suitable way to enrich contracts with explicit, clear and strict specifications that have to be guaranteed by software producers. As matter of fact it enables the enrichment of Service Level Agreement specification.
Bibliography


[82] Åke Grönlund and Annika Andersson. e-Government research quality improvements since 2003: more rigor, but research (perhaps) redefined. In *Wimmer et al. [225]*, pages 1–12. 14


[112] Mateja Kunstelj, Tina Jukic, and Mirko Vintar. Analysing the demand side of e-government: what can we learn from slovenian users? In Wimmer et al. [224], pages 305–317. 4


166

BIBLIOGRAPHY


[189] Maddalena Sorrentino and Enrico Ferro. Does the answer to egovernment lie in intermunicipal collaboration? An exploratory Italian case study. In Wimmer et al. [223], pages 1–12. [102]


[227] Petia Wohed, Wil M. P. van der Aalst, Marlon Dumas, and Arthur H. M. ter Hofstede. Analysis of web services composition languages: the case of


[245] Li Zhao, Qing Li, Xiaoli Liu, and Nanshan Du. A modeling method based on CCS for workflow. pages 376–384, 2009. [131][132]

Appendix A - e-Government Availability and Use

Figure 9.1: e-Gov Service Availability - Part I (Source Eurostat).

Figure 9.2: e-Gov Service Availability - Part II (Source Eurostat).
Figure 9.3: e-Gov Service Use by Citizens - Part I (Source Eurostat).

Figure 9.4: e-Gov Service Use by Citizens - Part II (Source Eurostat).

Figure 9.5: e-Gov Service Use by Citizens - Female - Part I (Source Eurostat).

Figure 9.6: e-Gov Service Use by Citizens - Female - Part II (Source Eurostat).
Figure 9.7: e-Gov Service Use by Citizens - Male - Part I (Source Eurostat).

Figure 9.8: e-Gov Service Use by Citizens - Male - Part II (Source Eurostat).

Figure 9.9: e-Gov Service Use by Enterprise - Part I (Source Eurostat).
Figure 9.10: e-Gov Service Use by Enterprise - Part II (Source Eurostat).
Appendix B - Mapping from BPMN and CSP

In this Appendix we presents the mapping from BPMN to CSP. We use the CSP syntax input of the PAT model checking.

Mapping Pool and Lane

The rule to transform the BPMN pool construct produces a CSP global constant. So the general idea is that each participant will be identified by such constant value.

Figure 9.11 provides the detailed transformation rule for a pool. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- PoolName contains the name of the pool.

Given the premises the resulting CSP global constant is named with the name of the pool and it has the value of the pool identifier.

The rule to transform the BPMN lane construct produces a CSP global variable, the value associated to such variable is set according to the number of task included in the lane. So the general idea is that each participant will be identified by such constant value. By default the lane variable value is null and it is incremented during the execution of the task.

Figure 9.12 provides the detailed transformation rule for a lane. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attribute:

- LaneID contains the identifier of the lane.

Given the premises the resulting CSP global constant is named with the ID of the lane and its value is null.
PoolID = "poolID";
PoolName = "poolName";

#define poolName poolID;

Figure 9.11: Mapping Pool.

LaneID = "laneID";

var laneID = 0;

Figure 9.12: Mapping Lane.
Mapping Sequence Flow

The rule to transform the BPMN sequence flow construct produces a CSP processes. So the general idea is that the process related flow is started by the trigger of an outgoing flows given by other CSP process related to other BPMN constructs and than it activates the incoming flows in other CSP process related to other BPMN constructs. When this process returns the outgoing flows are fired and the whole flow is terminate with success. The process can also be terminate in the case a termination occur in the BPMN process.

Figure 9.13 provides the detailed transformation rule for a sequence flow. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- **PoolID** contains the identifier of the pool;
- **SubProcID** contains the identifier of the subprocess, it could be null if the element is not contained in a subprocess;
- **FlowID** contains the identifier of the flow.

Given the premises the resulting CSP processes is defined in the following way. The process is named with the value assumed by FlowID. Three parameters are defined for this process the first one representing a flow Identifier, the second one the subprocess identifier and than the pool identifier.

Such CSP process is able to perform an event `esc` after which it will perform the event `enter`. Both events are characterized by the identifier of flow and pool that contain the flow itself. So the general idea is that the CSP process related flow is started by an interaction with the environment. Firstly it is requested the synchronization of the event `esc` with the corresponding `esc` event generated by another BPMN element where such flow is outcoming. After that synchronization with the environment via the event `enter` is requested. Also in this case the environment is represented by the CSP process generated by a BPMN element where the flow is incoming in the same pool with the same flow identifier. When the CSP process output of the mapping from BPMN flow construct returns the BPMN flow is fired and the whole CSP process terminates with success. The process can also terminates in case a BPMN event termination occur according to the process and sub-process where the flow is placed. A similar behavior is observable for the rule related to the conditional flow.

The BPMN simple sequence flow can be enriched by a condition that govern the flow execution implementing the abstraction discussed in the sequence flow mapping. The rule to transform the BPMN conditional flow construct produces
PoolID = "poolID";
SubProcID = "subProcID";
FlowID = "flowID";

\[
\text{FlowID}(\text{flowID}, \text{subProcID}, \text{poolID}) = \\
(\text{esc.flowID.poolID} \rightarrow \text{enter.flowID.poolID} \rightarrow \text{Skip}; ) \\
\quad (\text{terminate.poolID} \rightarrow \text{Skip}; ) \\
\quad (\text{terminate.poolID.subProcID} \rightarrow \text{Skip}; ) \\
\]

Figure 9.13: Mapping Sequence Flow.

two separate CSP processes. The first one is a CSP process permitting to represent
the flow as discussed in the simple flow. The second process bind the flow to the
condition. So the general idea is that the second process check the condition and
govern the process related to the flow.

Figure [9.14] provides the detailed transformation rule for conditional flow. In
the rule premises we show the graphical construct representation together with
the related attributes. In particular the construct is characterized by the following
attributes:

- PoolID contains the identifier of the pool;

- SubProcID contains the identifier of the subprocess, it could be null if the
element is not contained in a subprocess;

- FlowID contains the identifier of the flow;

- FlowCondition contain the expression that has to be check.

Given the premises the resulting two CSP processes are defined in the follow-
ing way. The first process is exactly the same of the flowID process described
previously. The second process is named with the value assumed by flowID pre-
fixed with the string “cond”. Then a list of parameters is defined for this process
the first one representing a flow identifier, the second one the subprocess identifier
and than the pool identifier. Internally the process check the condition and if it is
satisfied the FlowID process is invoked otherwise this process skip it flow.
PoolID = “poolID”;
SubProcID = “subProcID”;
FlowID = “flowID”;
flowConditionType = “expression”;

\[
\begin{align*}
&FlowID(flowID, subProcID, poolID) = \\
&\quad (esc.flowID.poolID \rightarrow enter.flowID.poolID \rightarrow \text{Skip};)
\quad \Box (terminate.poolID \rightarrow \text{Skip};)
\quad \Box (terminate.poolID.subProcID \rightarrow \text{Skip};)
\end{align*}
\]

\[
\begin{align*}
&\text{condFlowID(flowID, subProcID, poolID) =}
\quad \text{if (Expression) \{ FlowID(flowID, subProcID, poolID) \} else \{ Skip \}}
\end{align*}
\]

Figure 9.14: Mapping Conditional Sequence Flow.

Mapping Events

According to our study we propose a mapping from BPMN events to CSP process. Different characterizations are proposed for start, end and intermediate events. In the following we provide specific rules for each of them.

Start Event

The rule to transform the BPMN start event element produces a CSP processes. So the general idea is that the CSP process related to the start event is enabled and than it activates the outcoming flow implemented via other CSP process related to flow BPMN construct. When the start event process returns the outcoming flows are fired and the whole start event process is successfully terminated. The process can also be terminated without firing the outcoming flows in case a BPMN event termination occur.

Figure 9.15 provides the detailed transformation rule for a start event. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess, it could be null if the
element is not contained in a subprocess;

- StarID contains the identifier of the start event;
- FlowID contains the identifier of the outcoming flow.

Given the premises the resulting CSP processes is defined in the following way. The process is named with the value assumed by StarID. Parameters are defined for this process the first one representing a start event identifier, the second one the outcoming flow identifier, the third the subprocess identifier when the element is included in a subprocess, and the last one the pool identifier. Internally the CSP process is able to perform the event `enable` after which it will perform the event `esc`. The first event is characterized by the identifier of the BPMN start and the pool that contain such element. The second event is characterized by the identifier of the outcoming flow and pool that contain such flow. The general idea is that the CSP process is immediately enabled without any interaction with the environment. Than it is requested the synchronization of the events `esc` with the correspondent `esc` events generated by BPMN flow elements. When the CSP start event process returns the event related to the outcoming flows are fired and the CSP process is successfully terminated. Also in this case the process can be terminated when a BPMN event termination occur.

\[
\begin{align*}
\text{PoolID} &= \text{"poolID";} \\
\text{SubProcID} &= \text{"subProcID";} \\
\text{StartID} &= \text{"startID";} \\
\text{OutgoingEdges} &= \text{"flowID";} \\
\end{align*}
\]

Figure 9.15: Mapping Empty Start Event (one Output Flow)

In our study we provide a more general version of the transformation rule. As matter of fact the BPMN element can also be connected each other by more than one control flow. In Figure 9.16 we propose rules related to the general case with a generic number of input output flows. The other considerations done in the mapping of start event with one output flow are valid.
According to the BPMN specification the start event can be enriched by specializations. In the following we report the mapping related to start event typed with condition and signal. The mapping of timer type can be reduced to conditional one, where the condition predicate on the time. Moreover, the mapping related to message type start event will be discussed in the following (Section 9) and the mapping of multiple start event is trivial implemented putting the others in exclusive choice.

The BPMN start event can be enriched by a condition that govern the start execution implementing the abstraction discussed in the start event mapping. The rule to transform the BPMN conditional start event element produces two separate CSP processes. The first one is CSP process permitting to represent the start as discussed in the simple flow. The second process bind the start to the condition. So the general idea is that the second process check the condition and govern the process related to the flow. Figure 9.17 provides the detailed transformation rule for a start event with condition. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;
- StartID contains the identifier of the start event;
- StartCond contains the expression that has to be check;

Figure 9.16: Mapping Empty Start Event (n Output Flows).
• OutcomingEdgeID contains the identifiers of the outgoing flows.

Given the premises the resulting CSP processes are defined in the following way. The first process is exactly the same of the simple start process described previously. The second CSP process is named with the value assumed by StarID prefixed with the string “cond”. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the outgoing flows identifiers, than a parameter is introduced to identified the subprocess, and a last parameter is used to provide the pool identifier to the process.

Internally the process check the condition and if it is satisfied the CSP process related to the simple start event is invoked otherwise the conditional process skip its flow.

\[
\begin{align*}
\text{PoolID} & = \text{"poolID"}; \\
\text{SubProcID} & = \text{"subProcID"}; \\
\text{StartID} & = \text{"startID"}; \\
\text{StartCond} & = \text{"expression"}; \\
\text{OutgoingEdges} & = \text{"flowID_1, flowID_2, etc."};
\end{align*}
\]

\[
\text{startID}(\text{startID}, \text{flowID_1}, \ldots, \text{flowID_n}, \text{subProcID}, \text{poolID}) = \\
(\text{enable}.\text{startID}.\text{poolID} \rightarrow \\
(\text{esc}.\text{flow}_1.\text{poolID} \mid \ldots \mid \text{esc}.\text{flow}_n.\text{poolID}) \rightarrow \text{Skip}; ) \\
\Box (\text{terminate}.\text{poolID} \rightarrow \text{Skip}; ) \\
\Box (\text{terminate}.\text{poolID}.\text{subProcID} \rightarrow \text{Skip}; )
\]

\[
\text{CondstartID}(\text{startID}, \text{flowID_1}, \ldots, \text{flowID_n}, \text{subProcID}, \text{poolID}) =
\text{if} (\text{expression})
\{ \text{startID}(\text{startID}, \text{flowID_1}, \ldots, \text{flowID_n}, \text{subProcID}, \text{poolID}) \} \text{ else } \{ \text{Skip} \}
\]

Figure 9.17: Mapping Start Event with Conditional Type.

The rule to transform the BPMN start event construct typed with signal produces a CSP processes. So the general idea is a bit different in this case signals are exchanged using dedicate channels. Figure 9.18 provides the detailed transformation rule for a start event with signal. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:
PoolID contains the identifier of the pool;

SubProcID contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;

StartID contains the identifier of the start event;

OutcomingEdgeID contains the identifiers of the outcoming flows;

ChannelID contains the identifier of the channel where the signal is sended.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by StarID prefixed by the string “Sign”. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the outcoming flows identifiers, than a parameter is introduced to identified the subprocess, and a last parameter is used to provide the pool identifier to the process. Internally the process enable the start of the process flow represented by the CSP event named enable and characterized by the value of StarID and PoolID. The process prosecutes waiting for a message (the signal) on the signalID channel, allocated by the sender, and than it fires the events named esc and characterized by the value of FlowIDs and PoolID. After all the outcoming flows are started the process terminate successfully with the Skip. The process can also be terminated without be fire in the case a BPMN event termination occur.

\[
\begin{align*}
 PoolID = "poolID"; \\
 SubProcID = "subProcID"; \\
 StartID = "startID"; \\
 OutgoingEdges = "flowID1,flowID2,...,flowID_n"; \\
 ChannelID = "signalID";
\end{align*}
\]

\[
\begin{align*}
 \text{SignStartID}(startID, flowID_1, ..., flowID_n, subProcID, poolID) = \\
(\text{enable}.startID.poolID \rightarrow \text{signalID?sign}) \rightarrow \\
(\text{esc}.flow_1.poolID \ || \ ... \ || \ \text{esc}.flow_n.poolID) \rightarrow \text{Skip}; \\
\quad \Box (\text{terminate}.poolID \rightarrow \text{Skip};) \\
\quad \Box (\text{terminate}.poolID.subProcID \rightarrow \text{Skip};)
\end{align*}
\]

Figure 9.18: Mapping Start Event with Signal Type.
End Event

The rule to transform the BPMN end event construct produces a CSP processes. So the general idea is that the process related event enable the incoming flows implemented via other CSP process related to flow BPMN construct and than enabled itself. When the end event process returns the incoming flows are fired and the whole flow is successfully terminated. The process can also be terminate without be fire in the case a BPMN event termination occur.

Figure 9.19 provides the detailed transformation rule for an end event. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;
- EndID contains the identifier of the end event;
- IncomingEdgeID contains the identifiers of the incoming flows.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by EndID. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the incoming flow identifiers, than a parameter is introduced to identified the subprocess, and a last parameter is used to provide the pool identifier to the process. Internally the CSP process is able to perform the event enter after which it will perform the event enable. The first event is characterized by the identifier of the incoming flows and pool that contain such flow. The second event is characterized by the identifier of the BPMN end and the pool that contain such element. The general idea is that the CSP process request the synchronization of the events enter with the correspondent enter event generate by the mapping of BPMN flow elements. Than the end event is consumed via the event enable without any interaction with the environment. When the CSP end event process returns the CSP process is successfully terminated. Also in this case the process can be terminated when a BPMN event termination occur.

The end event can be enriched by elements that specialize it. In the following we report the mapping related to end event typed with signal and termination. Moreover, the mapping related to message type end event will be discussed in Section 9 and the mapping of multiple end event is trivial implemented putting the others in exclusive choice.
PoolID = “poolID”;
SubProcID = “subProcID”;
EndID = “endID”;
IncomingEdges = “flowID1, flowID2, ..., flowIDn”;

endID(endID, flowID1, ..., flowIDn, subProcID, poolID) =
((enter.flow1.poolID || ... || enter.flown.poolID)
→ enable.endID.poolID → Skip; )
□ (terminate.poolID → Skip; )
□ (terminate.poolID.subProcID → Skip; )

Figure 9.19: Mapping Empty End Event.

The rule to transform the BPMN end event construct typed with signal produces a CSP processes as previously discussed. So the general idea is a bit different in this case signals are exchanged using dedicate channels. Figure 9.20 provides the detailed transformation rule for the BPMN end event with signal. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;
- EndID contains the identifier of the end event;
- IncomingEdgeID contains the identifiers of the incoming flows;
- Sign contains the signal to send on the channel.
- ChannelID contains the identifier of the channel where the signal will be send.

Given the premises first of all a dedicate channel for the signal is allocated than the resulting CSP process is defined in the following way. The process is named with the value assumed by EndID prefixed by the string “signal”. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the incoming flow identifiers, than the parameters related to the sign that will be send
and a parameter to identified the subprocess are introduced, and a last parameter is considered to provide the pool identifier to the process.

Internally the process activates all the incoming flows with the parallel execution of $n$ enter events characterized by corresponding value of FlowIDs and PoolID. Then the process enable the end of the process flow represented by the CSP event named enable and characterized by the value of EndID and PoolID. The process prosecutes sending a message (the signal) on the signal channel. Finally the process terminate successfully. Also in this case the behavior of the BPMN flow construct can also be governed by an external event produced by a termination BPMN event. According to the element position, sub-process or in the main flow, the CSP process can be properly terminated.

$$\begin{align*}
\text{PoolID} &= \text{"poolID"}; \\
\text{SubProcID} &= \text{"subProcID"}; \\
\text{EndID} &= \text{"endID"}; \\
\text{IncomingEdges} &= \text{"flowID}_1, \text{flowID}_2, \ldots, \text{flowID}_n"; \\
\text{Sign} &= \text{"signal"}; \\
\text{ChannelID} &= \text{"signalID"};
\end{align*}$$

**Figure 9.20:** Mapping End Event with Signal Type.

The rule to transform the BPMN end event construct typed with terminate produces a CSP processes. In particular we distinguish two cases termination in the main process and termination in a subprocess. Generally speaking according to the position of such event the termination is propagated at the elements that are at the same level of the event.

In the first case the general idea is terminate all the process current active producing a termination of the pool related elements. Figure 9.21 provides the

$\begin{align*}
\text{Channel signalID} 0; \\
\text{signalEndID}(\text{endID}, \text{flowID}_1, \ldots, \text{flowID}_n, \text{signal}, \text{subProcID}, \text{poolID}) = \\
((\text{enter.flow}_1, \text{poolID} \parallel \ldots \parallel \text{enter.flow}_n, \text{poolID}) \\
\rightarrow \text{enable.endID}.\text{poolID} \\
\rightarrow \text{signalID}.\text{signal} \rightarrow \text{Skip}; )
\end{align*}$

$\blacklozenge (\text{terminate.poolID} \rightarrow \text{Skip});$

$\blacklozenge (\text{terminate.poolID}.\text{subProcID} \rightarrow \text{Skip});$

**Figure 9.20:** Mapping End Event with Signal Type.
detailed transformation rule for an end event with termination. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;
- EndID contains the identifier of the end event;
- IncomingEdgeID contains the identifiers of the incoming flows.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by EndID prefixed by the string “Terminate”. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the incoming flow identifiers and a last parameter is used to provide the pool identifier to the process.

Internally the process activates all the incoming flows with the parallel execution of n enter events characterized by corresponding flowIDs and PoolIDs. Then the process enable the end of the process flow represented by the CSP event named enable and characterized by the value of EndID and PoolID. The process prosecute invoking the terminate event on all the element of the pool (such event will following synchronized with all the other terminate event bind to different BPMN element according to the related poolID). Finally the process terminate successfully. After termination the end event can no be consumed anymore. The behavior of the BPMN flow construct can also be governed by an external event produced by a termination BPMN event. This is observable in the case that other terminations are introduced at the same level and they are executed before the termination under analysis has been fired.

In the second case when the BPMN termination end event is included in a subprocess, the transformation rule is a bit different. In such case only the elements included in the subprocess has to be terminated and the process control return to the main process flow. In this case the identifier of the subprocess has to be introduced as parameter of the terminate event. Also in this case the termination can be a possible behavior of the process in the case other termination are observable at the same level before the termination under analysis has been fired. Figure 9.22 shows the rule to map the end event with termination in a subprocess.

Before going into detail with the mapping of end event with error we introduce a brief overview of such BPMN element. The error thrown by the event will be caught by an intermediate event at a higher level. Error are not broadcast
PoolID = "poolID";
EndID = "endID";
IncomingEdges = "flowID_1, flowID_2, ..., flowID_n";

\[
\text{TerminateEndID}(\text{endID}, \; \text{flowID}_1, \; ..., \; \text{flowID}_n, \; \text{poolID}) = \\
((\text{enter}\_\text{flow}_1, \text{poolID} \parallel ... \parallel \text{enter}\_\text{flow}_n, \text{poolID}) \rightarrow \text{enable}\_\text{endID}, \text{poolID} \rightarrow \text{terminate}\_\text{poolID} \rightarrow \text{Skip} ; ) \\
\Box (\text{terminate}\_\text{poolID} \rightarrow \text{Skip} ; )
\]

Figure 9.21: Mapping End Event with Terminate.

PoolID = "poolID";
SubProcID = "subProcID";
EndID = "endID";
IncomingEdges = "flowID_1, flowID_2, ..., flowID_n";

\[
\text{terminateEndID}(\text{endID}, \; \text{flowID}_1, \; ..., \; \text{flowID}_n, \; \text{subProcID}, \; \text{poolID}) = \\
((\text{enter}\_\text{flow}_1, \text{poolID} \parallel ... \parallel \text{enter}\_\text{flow}_n, \text{PoolID}) \rightarrow \text{enable}\_\text{endID}, \text{poolID} \\
\rightarrow \text{terminate}\_\text{poolID}, \text{subProcID} \rightarrow \text{Skip} ; ) \\
\Box (\text{terminate}\_\text{poolID}, \text{subProcID} \rightarrow \text{Skip} ; ) \\
\Box (\text{terminate}\_\text{poolID} \rightarrow \text{Skip} ; )
\]

Figure 9.22: Mapping End Event with Terminate in a Subprocess.
throughout or across process. Error have a specific scope of visibility. An error can only be seen by a parent process. Other process at the same level or within different pools cannot see the error. Errors only move upward in the process hierarchy. If there happens to be more than one process level higher than the error end event, than first level that has a catch error intermediate event attached to its boundary will be interrupted even if there are higher levels that could possibly catch the same error. To guarantee the goodness of our mapping we request to the modeler to bind the sender and the receive of the error introducing a specific channel identifier.

The rule to transform the BPMN end event construct typed with error produces a CSP processes as previously discussed. So the general idea is a bit different in this case error are throw using a dedicate channel that govern the proper management of the error. Figure 9.24 provides the detailed transformation rule for an end event with error. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- ErrorID contains the identifier of the error.
- IncomingEdgeID contains the identifiers of the incoming flows;
- EndID contains the identifier of the end event.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by EndID prefixed by the string “error”. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the incoming flow identifiers, the parameters related to the error identifier that will be send, and a last parameter is used to provide the pool identifier to the process.

First of all a dedicate channel for the error is allocated. It is a general channel able to support send and receive of error messages. At the same time, internally the process activates all the incoming flows with the parallel execution of n enter events characterized by corresponding flowIDs and PoolID. Then the process enable the end of the process flow represented by the CSP event named enable and characterized by the value of EndID and PoolID. The process prosecutes sending a specific error on the error channel. Finally the process terminate successfully.

The behavior of the BPMN end event with error in a pool can also be governed by an external event produced by a termination BPMN event. According to the element position the CSP process can be properly terminated.
APPENDIX B - MAPPING FROM BPMN AND CSP

PoolID = “poolID”;
EndID = “endID”;
IncomingEdges = “flowID_1, flowID_2, ..., flowID_n”;
ErrorID = “errorID”;

Channel error 0;

errorEndID(endID, flowID_1, ..., flowID_n, errorID, poolID) =
((enter.flow_1.poolID || ... || enter.flow_n.poolID)
→ enable.endID.PoolID → error!errorID → terminate.poolID → Skip;)
□ (terminate.poolID → Skip;)

Figure 9.23: Mapping End Event with Error.

Notable is the case where the end event is included in a subprocess. The rule to transform the BPMN end event construct typed with error produces a CSP processes as previously discussed. In this case the termination condition observable as external choice is considered in the scope of the subprocess itself. Figure 9.24 provides the detailed transformation rule for such case.

Empty Intermediate Event

The rule to transform the BPMN intermediate event construct produces a CSP processes. So the general idea is that the process related event enable the incoming flows implemented via other CSP process related to flow BPMN construct and after that it enabled itself. Than it activates the outcoming flows implemented via other CSP process related to flow BPMN construct. When the intermediate event process returns the outcoming flows are fired and the whole flow is successfully terminated. There is also the case that such intermediate event is not consumed. This is the case when termination is observable at the same level where intermediate event is placed.

Figure 9.25 provides the detailed transformation rule for an intermediate event. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
PoolID = “poolID”;
SubProcID = “subProcID”;
EndID = “endID”;
IncomingEdges = “flowID1, flowID2,..., flowIDn”;
ErrorID = “errorID”;

Channel error 0:

\[
\text{errorEndID}(\text{endID}, \text{flowID}_1, ..., \text{flowID}_n, \text{errorID}, \text{subProcID}, \text{PoolID}) = \\
( (\text{enter}\_\text{flow}_1, \text{PoolID} \parallel ... \parallel \text{enter}\_\text{flow}_n, \text{PoolID}) \rightarrow \text{enable}\_\text{endID}\_\text{PoolID} \\
\rightarrow \text{error}!\text{errorID} \rightarrow \text{terminate}\_\text{poolID}\_\text{subProcID} \rightarrow \text{Skip}; ) \\
\Box (\text{terminate}\_\text{poolID}\_\text{subProcID} \rightarrow \text{Skip}; ) \\
\Box (\text{terminate}\_\text{poolID} \rightarrow \text{Skip}; )
\]

Figure 9.24: Mapping End Event With Error in a Subprocess.

- SubProcID contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;
- IntermediateID contains the identifier of the end event;
- IncomingEdgeID contains the identifiers of the incoming flows.
- OutcomingEdgeID contains the identifiers of the outcoming flows.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by IntermediateID. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the incoming flows identifiers, followed by a list of m parameters to represent the outcoming flows identifiers than a parameter is introduced to identified the containing subprocess, and a last parameter is used to provide the containing pool identifier.

Internally the process activates all the incoming flows with the parallel execution of n enter events characterized by corresponding flowIDs and PoolID. After all the incoming flows are started the process enable the BPMN intermediate event via the enable CSP event characterized by the value of IntermediateID and PoolID. The process prosecutes firing the events named esc and characterized by the value of FlowIDs and PoolIDs. After all the outcoming flows are started the process terminate successfully. The behavior of the BPMN flow construct can also be
APPENDIX B - MAPPING FROM BPMN AND CSP

governed by an external event produced by a termination BPMN event. According to the element position, sub-process or in the main flow, the CSP process can be properly terminated.

\[
\begin{align*}
\text{PoolID} &= "poolID"; \\
\text{SubProcID} &= "subProcID"; \\
\text{IntermediateID} &= "intID"; \\
\text{IncomingEdges} &= "flowInID_1, ..., flowInID_n"; \\
\text{OutcomingEdges} &= "flowOutID_1, ..., flowOutID_m";
\end{align*}
\]

\[
\text{intID}(\text{intID}, \text{flowInID_1}, ..., \text{flowInID_n}, \\
\text{flowOutID_1}, ..., \text{flowOutID_m}, \text{subProcID}, \text{poolID}) = \\
((\text{enter}.\text{flowInID_1}.\text{poolID} \parallel \ldots \parallel \text{enter}.\text{flowInID_n}.\text{poolID}) \rightarrow \\
\text{enable}.\text{endID}.\text{poolID} \rightarrow \\
(\text{esc}.\text{flowOutID_1}.\text{poolID} \parallel \ldots \parallel \text{esc}.\text{flowOutID_m}.\text{poolID}) \rightarrow \text{Skip}; \\
\Box(\text{terminate}.\text{poolID} \rightarrow \text{Skip}; \\
\Box(\text{terminate}.\text{poolID}.\text{subProcID} \rightarrow \text{Skip};
\]

Figure 9.25: Mapping Empty Intermediate Event.

As already discussed the intermediate event can be enriched by element that specialize it. In the following we report the mapping related to end event typed with signal, time, error and link. Thought and catch modality of such intermediate event has been considered. Moreover, the mapping related to message type end event will be discussed in the following (Section 9) and the mapping of multiple intermediate event is trivial implemented putting the others in exclusive choice.

The rule to transform the BPMN intermediate event typed with time produces a CSP processes guarded by the time conditions. Internally when the condition is verified the process behavior is the same of intermediate event. Figure 9.26 provides the detailed transformation rule for an intermediate event with timer.

For what concern the conditional type of intermediate event it can be easily derived from the flow with condition. In this case the condition is more general and defines a rule that must be satisfied in order to support process execution. Figure 9.27 shows the mapping rule.

The rule to transform the BPMN intermediate event typed with signal in catching modality produces a CSP processes. So the general idea is that the process related event enable the incoming flows implemented via other CSP process related to flow BPMN construct, and after receiving a signal that enabled itself, the outcoming flows implemented via other CSP process related to flow BPMN construct are activated. When the intermediate event process returns the outcoming
PoolID = “poolID”;
SubProcID = “subProcID”;
IntermediateID = “intID”;
IncomingEdges = “flowInID₁,…,flowInIDₙ”;
OutcomingEdges = “flowOutID₁,…,flowOutIDₘ”;
TimeCondition = “expression”;

intID(intID, flowInID₁, ..., flowInIDₙ,
flowOutID₁, ..., flowOutIDₘ, subProcID, poolID) =
(enter.flowInID₁.poolID || ... || enter.flowInIDₙ.poolID) →
enable.endID.poolID →
(esc.flowOutID₁.poolID || ... || esc.flowOutIDₘ.poolID) → Skip; )
☐ (terminate.poolID → Skip; )
☐ (terminate.poolID.subProcID → Skip; )

timerIntermediateID(intID, flowInID₁, ..., flowInIDₙ,
flowOutID₁, ..., flowOutIDₘ, subProcID, poolID) =
[expression] intID(intID, flowInID₁, ..., flowInIDₙ,
flowOutID₁, ..., flowOutIDₘ, subProcID, poolID)

Figure 9.26: Mapping Timer Intermediate Event.
PoolID = “poolID”;
SubProcID = “subProcID”;
IntermediateID = “intID”;
IncomingEdges = “flowInID_1,...,flowInID_n”;
OutcomingEdges = “flowOutID_1,...,flowOutID_m”;
InEventCondition = “expression”;

intID(intID, flowInID_1, ... , flowInID_n,
      flowOutID_1, ... , flowOutID_m, subProcID, poolID) =
      ((enter.flowInID_1.poolID || ... || enter.flowInID_n.poolID) →
       enable.endID.poolID →
       (esc.flowOutID_1.poolID || ... || esc.flowOutID_m.poolID) → Skip;)
      (terminate.poolID → Skip;)
      (terminate.poolID.subProcID → Skip;)

condIntEventID(intID, subProcID, poolID) =
if (Expression) { intID(intID, flowInID_1, ... , flowInID_n,
                     flowOutID_1, ... , flowOutID_m, subProcID, poolID) } else { Skip }

Figure 9.27: Mapping Conditional Intermediate Event.
flows are fired and the whole flow is successfully terminated.

Figure 9.28 provides the detailed transformation rule for an intermediate event with catching signal. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- **PoolID** contains the identifier of the pool;
- **SubProcID** contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;
- **IntermediateID** contains the identifier of the end event;
- **IncomingEdgeID** contains the identifiers of the incoming flows;
- **OutcomingEdgeID** contains the identifiers of the outgoing flows;
- **SignalID** is the identifier of the channel where send the signal.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by intermediate event prefixed by the string “signalc”. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the incoming flows identifiers, followed by a list of m parameters to represent the outgoing flows identifiers than a parameter is introduced to identified the containing subprocess, and a last parameter is used to provide the containing pool identifier.

Internally the process activates all the incoming flows with the parallel execution of n enter events characterized by corresponding flowIDs and PoolID. After all the incoming flows are started the process enable the BPMN intermediate event via the enable CSP event characterized by the value of IntermediateID and PoolID. The process prosecutes waiting for a message (the signal) on the signal channel and than it fires the events named esc and characterized by the value of FlowIDs and PoolID. After all the outcoming flows are started the process terminate successfully. As well as for the other case it is possible observe a termination event that abort the process.

The intermediate event typed with signal can also throwing a signal. Also in this case the rule to transform the BPMN intermediate event typed with signal in throwing modality produces a CSP processes. So the general idea is that the process related event enable the incoming flows implemented via other CSP process related to flow BPMN construct, and after sending a signal that enabled itself, the outcoming flows implemented via other CSP process related to flow BPMN construct are activated. When the intermediate event process returns the outcoming flows are fired and the whole flow is successfully terminated.
PoolID = “poolID”;
SubProcID = “subProcID”;
IntermediateID = “intID”;
IncomingEdges = “flowInID_1, ..., flowInID_n”;
OutcomingEdges = “flowOutID_1, ..., flowOutID_m”;
ChannelID = “signalID”;

\[
\text{signal} = \text{IntID}(\text{intID}, \text{flowInID}_1, ..., \text{flowInID}_n, \\
\text{flowOutID}_1, ..., \text{flowOutID}_m, \text{signal}, \text{subProcID}, \text{poolID}) = \\
(\text{enter} \cdot \text{flowInID}_1 . \text{PoolID} \parallel ... \parallel \text{enter} \cdot \text{flowInID}_n . \text{poolID}) \rightarrow \\
\text{enable} . \text{intID} . \text{poolID} \rightarrow \text{signal} \rightarrow \text{signal} \rightarrow \text{Signal} \\
\big( \text{esc} \cdot \text{flowOutID}_1 . \text{poolID} \parallel ... \parallel \text{esc} \cdot \text{flowOutID}_m . \text{poolID} \big) \rightarrow \text{Skip} ; \) \\
\big( \text{terminate} . \text{poolID} \rightarrow \text{Skip} ; \) \\
\big( \text{terminate} . \text{poolID} . \text{subProcID} \rightarrow \text{Skip} ; \)
\]

Figure 9.28: Mapping Signal Intermediate Event (Catching).

Figure 9.29 provides the detailed transformation rule for an intermediate event with signal. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;
- IntermediateID contains the identifier of the end event;
- IncomingEdgeID contains the identifiers of the incoming flows;
- OutcomingEdgeID contains the identifiers of the outcoming flows;
- ChannelID contains the identifier of the channel where send the signal.

Given the premises a dedicate channel for the signal is allocated and the resulting CSP process is defined in the following way. The process is named with the value assumed by intermediate event prefixed by the string “signalt”. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the incoming flows identifiers, followed by a list of m parameters to represent the outcoming...
APPENDIX B - MAPPING FROM BPMN AND CSP

flows identifiers than a parameter is introduced to identified the signal, this is followed by a parameter used to identified the subprocess, and a last parameter is used to provide the pool identifier to the process.

Internally the process activates all the incoming flows with the parallel execution of n enter events characterized by corresponding flowIDs and PoolID. After all the incoming flows are started the process enable the BPMN intermediate event via the enable CSP event characterized by the value of IntermediateID and PoolID. The process prosecutes sending a message (the signal) on the signal channel and than it fires the events named esc and characterized by the value of FlowIDs and PoolID. After all the outcoming flows are started the process terminate successfully. As well as the other cases it is possible observe a termination event that abort the process.

PoolID = “poolID”;
SubProcID = “subProcID”;
IntermediateID = “intID”;
IncomingEdges = “flowInID1,....,flowInIDn”;
OutcomingEdges = “flowOutID1,....,flowOutIDm”;
Sign = “signal”;
ChannelID = “signalID”;

Channel signalID 0;

signalIntID(intID, flowInID1, ... , flowInIDn,
flowOutID1, ... , flowOutIDm, signal, subProcID, PoolID) =
((enter.flowInID1.PoolID || ... || enter.flowInIDn.PoolID) →
enable.intID.PoolID → signalID?signal →
(esc.flowOutID1.PoolID || ... || esc.flowOutIDm.PoolID) → Skip; )
□ (terminate.poolID → Skip; )
□ (terminate.poolID,subProcID → Skip; )

Figure 9.29: Mapping Signal Intermediate Event (Throwing).

The rule to transform the BPMN intermediate event typed with error in catching modality produces a CSP processes. So the general idea is that the process related event enable the incoming flows implemented via other CSP process related to flow BPMN construct, and after receiving an error that enabled itself, the outcoming flows implemented via other CSP process related to flow BPMN construct are activated. When the intermediate event process returns the outcoming flows are fired and the whole flow is successfully terminated.
Figure 9.30 provides the detailed transformation rule for an intermediate event with catching error. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;
- IntermediateID contains the identifier of the end event;
- ErrorID contains the identifier of the error;
- ErrChannelID contains the channel where send the error;
- IncomingEdgeID contains the identifiers of the incoming flows;
- OutcomingEdgeID contains the identifiers of the outcoming flows.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by intermediate event identifier prefixed by the string err. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the incoming edge identifiers, followed by a list of m parameters to represent the outcoming flow identifiers than a parameter is introduced to identified the error followed by a parameter used to identified the subprocess, and a last parameter is used to provide the pool identifier to the process.

Internally the process activates all the incoming flows with the parallel execution of n enter events characterized by corresponding flow IDs and PoolID. After all the incoming flows are started the process enable the BPMN intermediate event via the enable CSP event characterized by the value of IntermediateID and PoolID. The process prosecutes waiting for a message (the error) on the error channel and than it fires the events named esc and characterized by the value of Flow IDs and PoolID. After all the outcoming flows are started the process terminate successfully with the Skip. As well as the other case it is possible observe a termination event that abort the process.

In the case the intermediate error event is associate to a specific subprocess a specific rule involving the subprocess itself has to be introduced. We discuss in detail such part the section devoted to subprocess.

The rule to transform the BPMN link event construct in catching modality produces a CSP processes as previously discussed. So the general idea is that after catching the link in input the event enable itself and it fires all the outcoming
PoolID = “poolID”;
SubProcID = “subProcID”;
IntermediateID = “intID”;
ErrorID = “errorID”;
IncomingEdges = “flowInID_1, ..., flowInID_n”;
OutcomingEdges = “flowOutID_1, ..., flowOutID_m”;
ErrChannelID = “error”;

Figure 9.30: Mapping Error Intermediate Event (Catching).

flows. Figure [9.31] provides the detailed transformation rule for intermediate link event in catching modality. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the event, it could be null if the element is not contained in a subprocess;
- LinkID contains the identifier of the link event;
- OutcomingEdgeID contains the identifiers of the outcoming flows.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by linkID prefixed by the string “cht”. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the outcoming flows identifiers, following by a parameter that provide the subprocess identifier, and a last parameter is used to provide the pool identifier to the process. Internally the process enable the link at the process flow represented by the CSP event named goto and characterized by the value of linkID and
PoolID. The process prosecutes firing the events named esc and characterized by
the value of FlowIDs and PoolID. After all the outcoming flows are started the
process terminate successfully. The termination of the element, due too process
abort, is also supported.

\[
\begin{align*}
\text{PoolID} & = \text{"poolID"];} \\
\text{SubProcID} & = \text{"subProcID"];} \\
\text{LinkID} & = \text{"linkID"];} \\
\text{OutgoingEdges} & = \text{"flowID}_1, \text{flowID}_2, \ldots, \text{flowID}_n;
\end{align*}
\]

\[
\text{chtlinkID}(\text{linkID}, \text{flowID}_1, \ldots, \text{flowID}_n, \text{subProcID}, \text{PoolID}) = \\
\begin{aligned}
goto.\text{linkID}.\text{PoolID} & \rightarrow (\text{esc.}\text{flow}_1.\text{PoolID} \parallel \ldots \parallel \text{esc.}\text{flow}_n.\text{PoolID}) \\
& \rightarrow \text{Skip; }
\end{aligned}
\]

\[
\begin{array}{l}
\Box (\text{terminate.}\text{poolID} \rightarrow \text{Skip; }) \\
\Box (\text{terminate.}\text{poolID.}\text{subProcID} \rightarrow \text{Skip; })
\end{array}
\]

Figure 9.31: Mapping Link Event (Catching).

The rule to transform the BPMN link construct in throwing modality produces
a CSP processes. Such process allows go to execution during the flow of the
process. So the general idea is allowing the link event after the firing of all the
incoming flows implemented via other CSP process related to flow BPMN con-
struct. The link switch the process control flow to a different CSP process. When
the link event process is consumed all the incoming flows are fired and the whole
flow is successfully terminated.

Figure 9.32 provides the detailed transformation rule for a link event in throw-
ing modality. In the rule premises we show the graphical construct representation
 together with the related attributes. In particular the construct is characterized by
the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the event,
it could be null if the element is not contained in a subprocess;
- linkID contains the identifier of the link event;
- IncomingEdgeID contains the identifiers of the incoming flows.
APPENDIX B - MAPPING FROM BPMN AND CSP

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by linkID prefixed by the string “thr”. Then a list of parameters is defined for this process the first element is used to represent the event identifier, then a list of n parameters is introduced to represent the incoming flows identifiers, following by a parameter that provide the subprocess identifier, and a last parameter is used to provide the pool identifier to the process. Internally the process activates all the incoming flows with the parallel execution of n enter events characterized by corresponding flowIDs and PoolID. After all the incoming flows are started the process enable the link via the goto event characterized by linkID and PoolID. Finally the process terminate successfully. As well in the other case the termination of the element, due too process abort, is also supported.

PoolID = “poolID”;
SubProcID = “subProcID”;
linkID = “linkID”;
IncomingEdges = “flowID_1, flowID_2, ..., flowID_n”;

\[
\text{thrlinkID}(\text{linkID}, \text{flowID}_1, ..., \text{flowID}_n, \text{subProcID}, \text{PoolID}) = \\
((\text{enter.flow}_1.\text{PoolID} \parallel ... \parallel \text{enter.flow}_n.\text{PoolID}) \\
\rightarrow \text{goto.linkID.PoolID} \rightarrow \text{Skip}; ) \\
\Box (\text{terminate.poolID} \rightarrow \text{Skip}; ) \\
\Box (\text{terminate.poolID.subProcID} \rightarrow \text{Skip}; ) \\
\]

Figure 9.32: Mapping Link Event (Throwing).

Mapping Activities

BPMN activity elements are task and subprocess. We provide the mapping for both of them according to the different options proposed by the notation.

Task

The rule to transform the BPMN simple task construct produces a CSP processes. The process bind the main task to the incoming and outcoming process flows. So
the general idea is that the incoming flows has to be activated and then the main task activity is executed. After that the outcoming flows are fired and the whole task is successfully terminated.

Figure 9.33 provides the detailed transformation rule for a task. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- **PoolID** contains the identifier of the pool;
- **SubProcID** contains the identifier of the subprocess that contain the task, it could be null if the element is not contained in a subprocess;
- **TaskID** contains the identifier of the task;
- **IncomingEdgesIDs** contains the identifiers of the incoming flows;
- **OutcomingEdgesIDs** contains the identifiers of the outcoming flows;
- **TaskIDTransparancy** contains the property of the task to be transparent, it can assume boolean value according to the modeler specification.

Given the premises the transparency variable is also included and assume true or false according to the BPMN task transparency property value. The resulting CSP process is defined in the following way. The process is named with the value assumed by TaskID. Then a list of parameters is defined for this process the first element is used to represent the task identifier, then a list of n parameters is introduced to represent the incoming flows identifiers, followed by a list of m parameters to represent the outcoming flows identifiers, than a parameters to identified the subprocess that contains the task, and a last parameter is used to provide the pool identifier to the process. Internally the process synchronize itself with all the CSP processes of the incoming flow. This is done through the parallel execution of an enter event for each incoming flow where each event is characterized by the corresponding FlowID and the PoolID. Then the process prosecutes with an event enable characterized by the TaskID and the PoolID. This event is necessary in case the task needs to synchronize itself with a send/receive message. At this point the process executes the corresponding BPMN task represented by the CSP event named work and characterized by the value of TaskID and PoolID. Then the process prosecutes activating all the outcoming flows with the parallel execution of m esc events characterized by corresponding flowIDs and PoolID. After all the outcoming flows are started the process terminate successfully. Instead of such main behavior the CSP process related to the task BPMN element can also be drastically terminated if abort event happened. In such case terminate event will be fired. The terminate event can characterized by the value of PoolID
or the values of PoolID and SubprocessID according to the position of the task, in a subprocess or in a main process respectively.

\[
\begin{align*}
\text{PoolID} &= \text{"poolID"}; \\
\text{SubProcID} &= \text{"subProcID"}; \\
\text{TaskID} &= \text{"taskID"}; \\
\text{IncomingEdgesIDs} &= \text{"flowInID}_1, \ldots, \text{flowInID}_n"; \\
\text{OutcomingEdgesIDs} &= \text{"flowOutID}_1, \ldots, \text{flowOutID}_m"; \\
\text{TaskIDTransparancy} &= \text{"traspValue"}; \\
\end{align*}
\]

\[
\text{var TaskIDTransparancy} = \text{"traspValue"};
\]

\[
\begin{align*}
\text{TaskID}(\text{taskID}, \text{flowInID}_1, \ldots, \text{flowInID}_n, \\
\text{flowOutID}_1, \ldots, \text{flowOutID}_m, \text{subProcID}, \text{poolID}) &= \\
(\text{enter} \cdot \text{poolID} \mid \ldots \mid \text{enter} \cdot \text{flowInID}_n \cdot \text{poolID}) \to \\
\text{enable} \cdot \text{taskID} \cdot \text{poolID} \to \text{work} \cdot \text{taskID} \cdot \text{poolID} \to \\
(\text{esc} \cdot \text{flowOutID}_1 \cdot \text{poolID} \mid \ldots \mid \text{esc} \cdot \text{flowOutID}_m \cdot \text{poolID}) \to \text{Skip; } \\
\square (\text{terminate} \cdot \text{poolID} \to \text{Skip; }) \\
\square (\text{terminate} \cdot \text{poolID} \cdot \text{subProcID} \to \text{Skip; })
\end{align*}
\]

Figure 9.33: Mapping Task.

As already discussed the task element can be enriched by element that specialize it. In the following we report the mapping related to task that are repeated more than one time. In particular we refer to loop task, multi-instance task in parallel and multi-instance task in sequence. Moreover, the mapping related to task sending or receiving message will be discussed in the following (Section 9).

The rule to transform the BPMN simple loop task construct produces two separate CSP processes. The first one is a CSP process permitting to represent the main task activity to be repeated a fixed number of times. The second process bind the main task to the incoming and outcoming CSP process related flows. So the general idea is that the second process is started by the incoming flows and then it activates the other process. When this process returns the outcoming flows are fired and the whole task terminate successfully.

Figure 9.34 provides the detailed transformation rule for a simple loop (i.e. where the loopingType attribute holds the “simple” value). In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
• SubProcID contains the identifier of the subprocess that contain the loop task, it could be null if the element is not contained in a subprocess;

• TaskID contains the identifier of the task;

• IncomingEdgesIDs contains the identifiers of the incoming flows;

• OutcomingEdgesIDs contains the identifiers of the outcoming flows;

• TaskIDTransparancy contains the property of the task to be transparent, it can assume boolean value according to the modeler specification;

• LoopCondition contains the condition that guard the loop execution;

• MaxLoop contains the maximum number of loops that can be executed.

Given the premises the transparency variable is also included and assume true or false according to the BPMN task transparency property value. The resulting two CSP processes are defined in the following way.

1. The first process is named with the value assumed by TaskID prefixed with the string “workloop”. Three parameters are defined for this process the first one representing a task identifier, the second one the subprocess identifier and the last one the pool identifier. Internally the process executes the corresponding BPMN task represented by the CSP event named work and characterized by the value of TaskID and PoolID. The process executes checking the condition governing the loop and in case it is evaluated to true the process restart after having decremented the loop index. Instead when the condition is evaluated to false the process terminates successfully. Such behavior of the process can be terminated when a termination event append. In such case the termination event characterized by the value of the PoolID and eventually by the value of the SubprocessID will be introduced and fired before that the process successfully terminate.

2. The second process is named with the value assumed by TaskID prefixed with the string “loop”. Then a list of parameters is defined for this process the first element is used to represent the task identifier, then a list of n parameters is introduced to represent the incoming flows identifiers, followed by a list of m parameters to represent the outcoming flows identifiers than a parameters to identified the subprocess that contains the task, and a last parameter is used to provide the pool identifier to the process. Internally the process synchronize itself with all the CSP process related to the incoming flows. This is done thorugh the parallel execution of an enter event for each incoming flow where each event is characterized by the corresponding
FlowID and the PoolID. Then the process prosecutes with an event enable characterized by the TaskID and the PoolID. This event is necessary in case the task needs to synchronize itself with a send/receive message. At this point the process call the other one to execute the loop. When the loop terminates the process activates all the outcoming flows with the parallel execution of m esc events characterized by corresponding FlowIDs and PoolID. After all the outcoming flows are started the process terminates successfully. Also in this case termination is introduced.

\[
\text{PoolID} = \text{"poolID"}; \\
\text{SubProcID} = \text{"subProcID"}; \\
\text{TaskID} = \text{"taskID"}; \\
\text{IncomingEdgesID} = \text{"flowInID}_1, \ldots, \text{flowInID}_n"; \\
\text{OutcomingEdgesID} = \text{"flowOutID}_1, \ldots, \text{flowOutID}_m"; \\
\text{LoopCondition} = \text{"Expression"}; \\
\text{MaxLoop} = \text{"numberofLoop"}; \\
\text{TaskIDTransparancy} = \text{"traspValue"};
\]

\[
\text{workloopTaskID}(\text{taskID}, \text{subProcID}, \text{poolID}) = \\
(\text{work.taskID.poolID} \rightarrow \text{if (MaxLoop > 0 && Expression = true) } \\
\{ \text{minus}\{\text{MaxLoop} = \text{MaxLoop} - 1; \} \rightarrow \text{WorkLoopActivityID(taskID, poolID) } \} \text{ else } \{ \text{Skip} \}) \\
\text{□ (terminate.poolID } \rightarrow \text{ Skip; )} \\
\text{□ (terminate.poolID.subProcID } \rightarrow \text{ Skip; )}
\]

\[
\text{loopTaskID}(\text{taskID}, \text{flowInID}_1, \ldots, \text{flowInID}_n, \\
\text{flowOutID}_1, \ldots, \text{flowOutID}_m, \text{subProcID, poolID}) = \\
(\text{(enter.flowInID}_1,\text{poolID} || \ldots || \text{enter.flowInID}_n,\text{poolID) } \rightarrow \text{enable.taskID.poolID } \rightarrow \text{WorkLoopActivityID(taskID, poolID); } \\
\text{(esc.flowOutID}_1,\text{poolID} || \ldots || \text{esc.flowOutID}_m,\text{poolID) } \rightarrow \text{Skip; )} \\
\text{□ (terminate.poolID } \rightarrow \text{ Skip; )} \\
\text{□ (terminate.poolID.subProcID } \rightarrow \text{ Skip; )}
\]

Figure 9.34: Mapping Task with Standard Loop.

For what concern multi instance task with instance executed in sequence a numeric expression is evaluated only once before the activity is performed. The
rule to transform the BPMN multi instance task with instance execute in sequence construct produces two separate CSP processes. The first one is a CSP process permitting to represent the main task activity to be repeated a fixed number of times. The second process binds the main task to the CSP process related to incoming and outcoming flows. So the general idea is that the second process is started by the incoming flows and then it activates the recursive process. When this process returns the outcoming flows are fired and the whole task is terminated.

Figure 9.35 provides the detailed transformation rule for a multi instances task with instance execute in sequence (i.e. where the loopingType attribute holds the “multiInstanceSequence” value). In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess, it could be null if the task is not contained in a subprocess;
- TaskID contains the identifier of the task;
- IncomingEdgesIDs contains the identifiers of the incoming flows;
- OutcomingEdgesIDs contains the identifiers of the outcoming flows;
- TaskIDTransparancy contains the property of the task to be transparent, it can assume boolean value according to the modeler specification;
- NOfSeqActivity contains the maximum number of task instance that can be instantiated in sequence.

Given the premises the transparency variable is also included and assume true or false according to the BPMN task transparency property value. The resulting two CSP processes are defined in the following way.

1. The first process is named with the value assumed by TaskID prefixed with the string “workMLoop”. Three parameters are defined for this process the first one representing a task identifier, the second one the subprocess identifier and the last one the pool identifier. Internally the process checks the condition governing the loop and in case it is evaluated to true the process executes the corresponding BPMN task represented by the CSP event named work and characterized by the value of TaskID and PoolID. After that the process restart itself after having decremented the loop index. Instead when the condition is evaluated to false the process terminates successfully. Also in this case termination due to terminate event is introduced.
2. The second process is named with the value assumed by TaskID prefixed with the string “MultiInstanceSeq”. Then a list of parameters is defined for this process the first element is used to represent the task identifier, then a list of \( n \) parameters is introduced to represent the incoming flows identifiers, followed by a list of \( m \) parameters to represent the outcoming flows identifiers than a parameters to identified the subprocess that contains the task, and a last parameter is used to provide the pool identifier to the process. Internally the process synchronize itself with all the CSP process related to the incoming flows. This is done through the parallel execution of an enter event for each incoming flow where each event is characterized by the corresponding FlowID and the PoolID. Then the process prosecutes with an event enable characterized by the TaskID and the PoolID. This event is necessary in case the task needs to synchronize itself with a send/receive message. At this point the process call the other one to execute the requested loops. When the loop terminates the process activates all the outcoming flows with the parallel execution of \( m \) esc events characterized by corresponding FlowIDs and PoolID. After all the outcoming flows are started the process terminates successfully. Also in this case termination is introduced.

The rule to transform the BPMN multi instance task with instance execute in parallel construct produces three separate CSP processes. The first one is a CSP process permitting to represent the main task activity. The second process allows to implement parallel executions of the main task. The third process bind the main task to the incoming and outcoming process flows. So the general idea is that the third process is started by the incoming flows and then it activates the parallel execution of the task. When this processes returns the outcoming flows are fired and the whole task is successfully terminated.

Figure 9.36 provides the detailed transformation rule for a multi instances task with instance execute in parallel (i.e. where the loopingType attribute holds the “multiInstanceParallel” value). In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess, it could be null if the task is not contained in a subprocess;
- TaskID contains the identifier of the task;
- IncomingEdgesIDs contains the identifiers of the incoming flows;
- OutcomingEdgesIDs contains the identifiers of the outcoming flows;
APPENDIX B - MAPPING FROM BPMN AND CSP

PoolID = “poolID”;
SubProcID = “subProcID”;
TaskID = “taskID”;
IncomingEdgesIDs = “flowInID_1,...,flowInID_n”;
OutcomingEdgesIDs = “flowOutID_1,...,flowOutID_m”;
NOfSeqActivity = “numberofSeq”;
TaskIDTransparancy = “traspValue”;

\[
\text{var TaskIDTransparancy = “traspValue”};
\]

workMLoopTaskID(taskID, subProcID, poolID) =
(if (NOfSeqActivity > 0)
{ minus{NOfSeqActivity = NOfSeqActivity - 1; } \rightarrow
work.taskID.poolID \rightarrow
workMLoopTaskID(taskID, subProcID, poolID) } else { Skip})
\[\square (\text{terminate.poolID} \rightarrow \text{Skip;})
\[\square (\text{terminate.poolID.subProcID} \rightarrow \text{Skip;})

MultiInstanceSeqTaskID(taskID, flowInID_1, ..., flowInID_n,
flowOutID_1, ..., flowOutID_m, subProcID, poolID) =
((enter.flowInID_1.poolID || ... || enter.flowInID_n.poolID) \rightarrow
enable.taskID.poolID \rightarrow
workMLoopTaskID(taskID, subProcID, poolID);
(esc.flowOutID_1.poolID || ... || esc.flowOutID_m.poolID) \rightarrow \text{Skip})

\[\square (\text{terminate.poolID} \rightarrow \text{Skip;})
\[\square (\text{terminate.poolID.subProcID} \rightarrow \text{Skip;})

Figure 9.35: Mapping Task with Multi Instance in Sequence.
• TaskIDTransparency contains the property of the task to be transparent, it can assume boolean value according to the modeler specification.

• NOfParActivity contains the maximum number of task instance that can be instantiated in parallel.

Given the premises the transparency variable is also included and assume true or false according to the BPMN task transparency property value. The resulting three CSP processes are defined in the following way.

1. The first process is named with the value assumed by TaskID prefixed with the string "simp". Two parameters are defined for this process the first one represents a task identifier and the second one refers to the pool identifier. Internally the process executes the corresponding BPMN task represented by the CSP event named work and characterized by the value of TaskID and PoolID. After that the process successfully terminate.

2. The second process is named with the value assumed by TaskID prefixed with the string “parallel”. Then a list of parameters is defined for this process the first element is used to represent the task identifier, than a parameters to identified the subprocess that contains the task and finally a parameter is used to provide the pool identifier to the process. Internally the process execute in parallel the other process already defined according to the times that are specified. Termination due to BPMN event terminate is alternative considered as behavior of this process.

3. The third process is named with the value assumed by TaskID prefixed with the string “MultiInstancePar”. Then a list of parameters is defined for this process the first element is used to represent the task identifier, then a list of n parameters is introduced to represent the incoming flows identifiers, followed by a list of m parameters to represent the outcoming flows identifiers, than a parameters to identified the subprocess that contains the task, and a last parameter is used to provide the pool identifier to the process. Internally the process synchronize itself with all the CSP process related to the incoming flows. This is done through the parallel execution of an enter event for each incoming flow where each event is characterized by the corresponding FlowID and the PoolID. Then the process prosecutes with an event enable characterized by the TaskID and the PoolID. This event is necessary in case the task needs to synchronize itself with a send/receive message. At this point the process call the second process to execute the activity. When the second process is terminates the process activates all the outcoming flows with the parallel execution of m esc events characterized by corresponding
FlowIDs and PoolID. After all the outcoming flows are started the process successful terminate. Also in this case termination due to BPMN event terminate is alternative considered as behavior of this process.

\[
\text{PoolID} = \text{“poolID”};
\]
\[
\text{SubProcID} = \text{“subProcID”};
\]
\[
\text{TaskID} = \text{“taskID”};
\]
\[
\text{IncomingEdgesIDs} = \text{“flowInID}_{1},...,\text{flowInID}_{n}”;
\]
\[
\text{OutcomingEdgesIDs} = \text{“flowOutID}_{1},...,\text{flowOutID}_{m}”;
\]
\[
\text{NOfParActivity} = \text{“numberofPar”};
\]
\[
\text{TaskIDTransparancy} = \text{“traspValue”};
\]

\[
\text{var TaskIDTransparancy} = \text{“traspValue”};
\]
\[
\text{simpTaskID(taskID, poolID) = work.taskID.poolID} \rightarrow \text{Skip};
\]
\[
\text{parallelTaskID(taskID, subProcID, PoolID) =}
\]
\[
(\parallel x : \{0..\text{numberofPar}\} @ \text{TaskID(taskID, poolID)})
\]
\[
\Box (\text{terminate.poolID} \rightarrow \text{Skip};)
\]
\[
\Box (\text{terminate.poolID.subProcID} \rightarrow \text{Skip};)
\]
\[
\text{MultiInstanceParTaskID(taskID, flowInID}_{1},...,\text{flowInID}_{n},\]
\[
\text{flowOutID}_{1},...,\text{flowOutID}_{m}, \text{subProcID, poolID) =}
\]
\[
((\text{enter.flowInID}_{1}\text{.poolID} \parallel ... \parallel \text{enter.flowInID}_{n}\text{.poolID}) \rightarrow 
\]
\[
\text{enable.taskID.poolID} \rightarrow 
\]
\[
\text{parallelTaskID(taskID, subProcID, PoolID)};
\]
\[
(\text{esc.flowOutID}_{1}\text{.poolID} \parallel ... \parallel \text{esc.flowOutID}_{m}\text{.poolID}) \rightarrow \text{Skip};)
\]
\[
\Box (\text{terminate.poolID} \rightarrow \text{Skip};)
\]
\[
\Box (\text{terminate.poolID.subProcID} \rightarrow \text{Skip};)
\]

Figure 9.36: Mapping Task with Multi Instance in Parallel.

**Subprocess**

The rule to transform the BPMN simple subprocess construct produces a CSP processes. So the general idea is that the incoming flows has to be activated and then the subprocess elements are executed. After that the outcoming flows are fired and the whole subprocess is terminated.
Figure 9.37 provides the detailed transformation rule for a subprocess. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubprocessID contains the identifier of the subprocess;
- SubProcID contains the identifier of the subprocess that contain the subprocess under analysis, it could be null if the element is not contained in a subprocess;
- IncomingEdgesIDs contains the identifiers of the incoming flows;
- OutcomingEdgesIDs contains the identifiers of the outcoming flows;
- SubstartID contains the identifiers of the start events implemented in the subprocess;
- SubendID contains the identifiers of the end events implemented in the subprocess.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by SubprocessID. Then a list of parameters is defined for this process the first element is used to represent the subprocess identifier, than a list of \( z \) elements is used to represent the start events in the subprocess, following by a list of \( y \) elements referring to the end events inside the subprocess, then a list of \( n \) parameters is introduced to represent the incoming edges identifiers, followed by a list of \( m \) parameters to represent the outcoming flows identifiers. Two more parameters are also introduced the identifier of the subprocess that contains the subprocess itself and the last one is the pool identifier where the process lies. Internally the process synchronize itself with all the incoming flow via the respectively CSP processes. This is done through the parallel execution of an enter event characterized by the corresponding FlowID and the PoolID for each incoming flows. Then the process prosecutes with a parallel execution of events enable characterized by the SubstartIDs and the PoolID followed by the parallel execution of the events enable characterized by the SubendIDs and the PoolID. At this point the process executes the BPMN elements inside the subprocess represented by other CSP process which are outside the scope of the subprocessID process in which the events enable occur. Then the process prosecutes activating all the outcoming flows with the parallel execution of \( m \) esc events characterized by corresponding FlowIDs and PoolID. After all the outcoming flows are started the process is successfully terminated. Termination is
APPENDIX B - MAPPING FROM BPMN AND CSP

a possible behavior of the process according to an external event execution bind with CSP process related to BPMN termination event.

Notable is the lack of the event enable on the subprocess. We remove it because of the CSP process related to BPMN subprocess does not request messages exchange. It is in charge of the elements included in the subprocess.

\[
\text{PoolID} = \text{"poolID"}; \\
\text{SubProcID} = \text{"subProcID"}; \\
\text{SubprocessID} = \text{"subprocessID"}; \\
\text{IncomingEdges} = \text{"flowInID}_1, \ldots, \text{flowInID}_n"; \\
\text{OutcomingEdges} = \text{"flowOutID}_1, \ldots, \text{flowOutID}_m"; \\
\text{SubstartID} = \text{"substartID}_1, \ldots, \text{substartID}_z"; \\
\text{SubendID} = \text{"subendID}_1, \ldots, \text{subendID}_y";
\]

\[
\text{subprocessID} (\text{subprocessID}, \text{substartID}_1, \ldots, \text{substartID}_x, \text{subendID}_1, \ldots, \text{subendID}_y, \text{flowInID}_1, \ldots, \text{flowInID}_n, \text{flowOutID}_1, \ldots, \text{flowOutID}_m, \text{poolID}) = \\
((\text{enter}.\text{flowInID}_1.\text{poolID} \mathbin{|} \cdots \mathbin{|} \text{enter}.\text{flowInID}_n.\text{poolID}) \rightarrow \\
(\text{enable}.\text{substartID}_1.\text{poolID} \mathbin{|} \cdots \mathbin{|} \text{enable}.\text{substartID}_z.\text{poolID}) \rightarrow \\
(\text{enable}.\text{subendID}_1.\text{poolID} \mathbin{|} \cdots \mathbin{|} \text{enable}.\text{subendID}_y.\text{poolID}) \rightarrow \\
(\text{esc}.\text{flowOutID}_1.\text{poolID} \mathbin{|} \cdots \mathbin{|} \text{esc}.\text{flowOutID}_m.\text{poolID}) \rightarrow \text{Skip}; ) \\
\square (\text{terminate}.\text{poolID} \rightarrow \text{Skip}; ) \\
\square (\text{terminate}.\text{poolID}.\text{subProcID} \rightarrow \text{Skip}; )
\]

Figure 9.37: Mapping Subprocess.

As already discussed the subprocess element can be enriched by specifications. In the following we report the mapping related to subprocess that are repeated more than one time. In particular we refer to loop subprocess, multi-instance subprocess in parallel and multi-instance task in sequence.

The rule to transform the BPMN simple loop subprocess produces two separate CSP processes. The first one is a CSP process enabling the main subprocess task to be repeated a fixed number of times. The second process bind the main subprocess to the incoming and outcoming process flows. So the general idea is that the second process is started by the incoming flows and then it activates the recursive process. When this process returns the outcoming flows are fired and the whole subprocess is successfully terminated.

Figure 9.38 provides the detailed transformation rule for a simple loop. In the rule premises we show the graphical construct representation together with
the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubprocessID contains the identifier of the subprocess;
- SubProcID contains the identifier of the subprocess that contain the subprocess under analysis, it could be null if the element is not contained in a subprocess;
- IncomingEdgeIDs contains the identifiers of the incoming flows;
- OutcomingEdgesIDs contains the identifiers of the outcoming flows;
- LoopCondition contains the condition that guard the loop execution;
- MaxLoop contains the maximum number of loop that can be executed;
- SubstartID contains the identifiers of the start events implemented in the subprocess;
- SubendID contains the identifiers of the end events implemented in the subprocess.

Given the premises the resulting two CSP processes are defined in the following way.

1. The first process is named with the value assumed by SubprocessID prefixed with the string “workLoop”. Then a list of parameters is defined for this process the first element is used to represent the subprocess identifier, than a list of $z$ elements is used to represent the start events in the subprocess, following a list of $y$ elements refers to the end events inside the subprocess, then a list of $n$ parameters is introduced to represent the incoming edges identifiers, followed by a list of $m$ parameters to represent the outcoming flows identifiers. Two more parameters are also introduced the identifier of the subprocess that contains the subprocess itself and the last one is the pool identifier where the process lies. This process is composed by a parallel execution of events enable characterized by the SubstartIDs and the PoolID and followed by the parallel execution of the events enable characterized by the SubendIDs and the PoolID. The process executes the corresponding BPMN elements inside the subprocess represented by other CSP process which are outside the scope of the subprocessID process in which the events enable occur. The the process prosecutes checking the condition governing
the loop and in case it is evaluated to true the process restart after having decremented the loop index. Otherwise when the condition is evaluated to false the process terminates successfully. Such main behavior the CSP process related to the task BPMN element can also be drastically terminated if abort event happened.

2. The second process is named with the value assumed by SubprocessID prefixed with the string “loop”. Then a list of parameters is defined for this process the first element is used to represent the subprocess identifier, than a list of \( z \) elements is used to represent the start events in the subprocess, following a list of \( y \) elements refers to the end events inside the subprocess, then a list of \( n \) parameters is introduced to represent the incoming edges identifiers, followed by a list of \( m \) parameters to represent the outcoming flows identifiers. Two more parameters are also introduced the identifier of the subprocess that contains the subprocess itself and the last one is the pool identifier where the process lies. Internally the process synchronize itself with the CSP process related to all the incoming flows. This is done through the parallel execution of an enter event for each incoming flow where each event is characterized by the corresponding FlowID and the PoolID. At this point the process call the other one to execute the loop. When the loop terminates the process activates all the outcoming flows with the parallel execution of \( m \) esc events characterized by corresponding FlowIDs and PoolID. After all the outcoming flows are started the process is terminated with success. Termination is considered also in this case.

For what concern multi instances subprocess with instances executed in sequence a numeric expression is evaluated only once before the subprocess is performed. The rule to transform the BPMN multi instances subprocess with instances execute in sequence produces two separate CSP processes. The first one is a CSP process mapping the start and the end events of the subprocess for a fixed number of times. The second process bind start and end events to the CSP process related incoming and outcoming process flows. So the general idea is that the second process is started by the incoming flows and then it activates the recursive process. When this process returns the outcoming flows are fired and the whole task terminate with success. Alternative the termination of the event is requested due to the BPMN process abort.

Figure 9.39 provides the detailed transformation rule for a multi instances task with instances execute in sequence. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
PoolID = “poolID”;
SubprocessID = “subprocessID”;
SubProcID = “subProcID”;
IncomingEdgesIDs = “flowInID1, ..., flowInIDn”;
OutcomingEdgesIDs = “flowOutID1, ..., flowOutIDm”;
SubstartID = “substartID1, ..., substartIDz”;
SubendID = “subendID1, ..., subendIDy”;
LoopCondition = “expression”;
MaxLoop = “numberOfLoop”;

\[
\text{workLoopsubprocessID}(\text{subprocessID}, \text{substartID}_1, \ldots, \\
\text{substartID}_z, \text{subendID}_1, \ldots, \text{subendID}_y, \text{subProcID}, \text{poolID}) = \\
((\text{enable}.\text{substartID}_1.\text{poolID} \parallel \ldots \parallel \text{enable}.\text{substartID}_z.\text{poolID}) \rightarrow \\
(\text{enable}.\text{subendID}_1.\text{poolID} \parallel \ldots \parallel \text{enable}.\text{subendID}_y.\text{poolID}) \rightarrow \\
\text{if} \ (\text{MaxLoop} > 0 \iff \text{expression} = \text{true}) \\
\{ \text{minus}\{\text{MaxLoop} = \text{MaxLoop} - 1; \} \\
\rightarrow \text{workLoopsubprocessID}(\text{subprocessID}, \text{substartID}_1, \ldots, \\
\text{substartID}_z, \text{subendID}_1, \ldots, \text{subendID}_y, \text{poolID}) \} \\
\text{else} \{\text{Skip}\};) \}
\]
\[
\square (\text{terminate}.\text{poolID} \rightarrow \text{Skip};)
\]
\[
\square (\text{terminate}.\text{poolID}.\text{subProcID} \rightarrow \text{Skip};)
\]

\[
\text{loopSubprocessID}(\text{subprocessID}, \text{substartID}_1, \ldots, \text{substartID}_z, \\
\text{subendID}_1, \ldots, \text{subendID}_y, \text{flowInID}_1, \ldots, \text{flowInID}_n, \\
\text{flowOutID}_1, \ldots, \text{flowOutID}_m, \text{subProcID}, \text{poolID}) = \\
((\text{enter}.\text{flowInID}_1.\text{poolID} \parallel \ldots \parallel \text{enter}.\text{flowInID}_n.\text{poolID}) \rightarrow \\
\text{workLoopsubprocessID}(\text{subprocessID}, \text{substartID}_1, \ldots, \\
\text{substartID}_z, \text{subendID}_1, \ldots, \text{subendID}_y, \text{poolID}) \rightarrow \\
(\text{esc}.\text{flowOutID}_1.\text{poolID} \parallel \ldots \parallel \text{esc}.\text{flowOutID}_m.\text{poolID}) \rightarrow \text{Skip};)
\]
\[
\square (\text{terminate}.\text{poolID} \rightarrow \text{Skip};)
\]
\[
\square (\text{terminate}.\text{poolID}.\text{subProcID} \rightarrow \text{Skip};)
\]

Figure 9.38: Mapping Looping Subprocess.
SubprocessID contains the identifier of the subprocess;

SubProcID contains the identifier of the subprocess that contain the subprocess under analysis, it could be null if the element is not contained in a subprocess;

IncomingEdgesIDs contains the identifiers of the incoming flows;

OutcomingEdgesIDs contains the identifiers of the outcoming flows;

NOfSeqSub contains the maximum number of task instance that can be instantiated;

SubstartID contains the identifiers of the start events implemented in the subprocess;

SubendID contains the identifiers of the end events implemented in the subprocess.

Given the premises the resulting two CSP processes are defined in the following way. Termination is considered in both processes.

1. The first process is named with the value assumed by SubprocessID prefixed with the string “workMLoop”. Then a list of parameters is defined for this process the first element is used to represent the subprocess identifier, than a list of z elements is used to represent the start events in the subprocess, following a list of y elements refers to the end events inside the subprocess, then a list of n parameters is introduced to represent the incoming edges identifiers, followed by a list of m parameters to represent the outcoming flows identifiers. Two more parameters are also introduced the identifier of the subprocess that contains the subprocess itself and the last one is the pool identifier where the process lies. Internally the process checks the condition governing the loop and in case it is evaluated to true the process executes in parallel the corresponding BPMN start events followed by a parallel execution of end events represented by two CSP events named enable and characterized on one hand by the values of SubstartIDs and PoolID and on the other hand by the value of SubendIDs and PoolID. After that the process restart itself decrementing the loop index. When the condition is evaluated as false the process terminates successfully.

2. The second process is named with the value assumed by SubprocessID prefixed with the string “MultiIn”. Then a list of parameters is defined for this process the first element is used to represent the task identifier, than a list of z elements is used to represent the start events in the subprocess, following
a list of y elements refers to the end events inside the subprocess, then a
list of n parameters is introduced to represent the incoming edges identifiers,
followed by a list of m parameters to represent the outcoming flows identifiers. Two more parameters are also introduced the identifier of the
subprocess that contains the subprocess itself and the last one is the pool
identifier where the process lies. Internally the process synchronize itself
with all the CSP process related to incoming flows. This is done through
the parallel execution of an enter event for each incoming flow where each
event is characterized by the corresponding FlowID and PoolID. At this
point the process call the other one to execute the loops. When the loops
terminate the process activates all the outcoming flows with the parallel exe-
cution of m esc events characterized by corresponding FlowIDs and PoolID.
After all the outcoming flows are started the process terminate successfully.

The rule to transform the BPMN multi instances process with instances ex-
ecute in parallel produces three separate CSP processes. The first one is a CSP
process permitting to invoke the start and the end event of the subprocess. The
second allows to implement parallel executions of the start and end events. The
third bind the main task to the incoming and outcoming process flows. So the
general idea is that the third process is started by the incoming flows and then
it activates the parallel execution of the task. When this processes returns the
outcoming flows are fired and the whole task is successfully terminate.

Figure 9.40 provides the detailed transformation rule for a multi instance sub-
process with instance execute in parallel. In the rule premises we show the graph-
archical construct representation together with the related attributes. In particular the
construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubprocessID contains the identifier of the subprocess;
- SubProcID contains the identifier of the subprocess that contain the sub-
  process under analysis, it could be null if the element is not contained in a
  subprocess;
- IncomingEdgesIDs contains the identifiers of the incoming flows;
- OutcomingEdgesIDs contains the identifiers of the outcoming flows;
- NumberOfParSub contains the maximum number of task instance that can
  be instantiated;
- SubstartID contains the identifiers of the start events implemented in the
  subprocess;
PoolID = "poolID";
SubprocessID = "subprocessID";
SubProcID = "subProcID";
IncomingEdgesIDs = "flowInID1,...,flowInIDn";
OutcomingEdgesIDs = "flowOutID1,...,flowOutIDm";
NOSeqSub = "numberofSeq";
SubstartID = "substartID1,...,substartIDz";
SubendID = "subendID1,...,subendIDy";

\[
\text{WorkMLoopSubprocessID}(\text{subprocessID}, \text{substartID}_1, \ldots, \text{substartID}_z, \\
\text{subendID}_1, \ldots, \text{subendID}_y, \text{subProcID}, \text{poolID}) = \\
\begin{cases}
\text{WorkMLoopSubprocessID}(\text{subprocessID}, \text{substartID}_1, \ldots, \text{substartID}_x, \\
\text{subendID}_1, \ldots, \text{subendID}_y, \text{subProcID}, \text{poolID}) & \text{if (NumberOfSeqSubProc > 0)} \\
\text{Skip} & \text{else}
\end{cases}
\]

\[
\text{MultiInSubprocessID} \left( \text{subprocessID}, \text{substartID}_1, \ldots, \text{substartID}_x, \\
\text{subendID}_1, \ldots, \text{subendID}_y, \text{flowInID}_1, \ldots, \text{flowInID}_n, \\
\text{flowOutID}_1, \ldots, \text{flowOutID}_m, \text{subProcID}, \text{poolID} \right) = \\
\begin{cases}
\text{WorkMLoopSubprocessID} \left( \text{subprocessID}, \text{substartID}_1, \ldots, \text{substartID}_x, \\
\text{subendID}_1, \ldots, \text{subendID}_y, \text{subProcID}, \text{poolID} \right) & \text{if (NumberOfSeqSubProc > 0)} \\
\text{Skip} & \text{else}
\end{cases}
\]

Figure 9.39: Mapping Subprocess with Multi Instance in Sequence.
- SubendID contains the identifiers of the end events implemented in the subprocess.

Given the premises the resulting three CSP processes are defined in the following way.

1. The first process is named with the value assumed by SubprocessID prefixed with the string “sub”. Then a list of parameters is defined for this process the first element is used to represent the subprocess identifier, than a list of \( z \) elements is used to represent the start events in the subprocess, following a list of \( y \) elements refers to the end events inside the subprocess, and the last one is the pool identifier where the process lies. Internally the process executes a parallel execution of the corresponding BPMN start events followed by a parallel execution of the end events. They are represented by two CSP events named enable and characterized on one hand by the value of SubstartID and PoolID and on the other hand by the value of SubendID and PoolID. After that the process successfully terminate.

2. The second process is named with the value assumed by SubprocessID prefixed with the string “parallel”. Then a list of parameters is defined for this process the first element is used to represent the subprocess identifier, than a list of \( z \) elements is used to represent the start events in the subprocess, following a list of \( y \) elements refers to the end events inside the subprocess, then a list of \( n \) parameters is introduced to represent the incoming edges identifiers, followed by a list of \( m \) parameters to represent the outcoming flows identifiers. Two more parameters are also introduced the identifier of the subprocess that contains the subprocess itself and the last one is the pool identifier where the process lies. Internally the process execute the first process in parallel according to the number of the specified parallel executions. Termination is also supported in such process.

3. The third process is named with the value assumed by TaskID prefixed with the string “MultiInstancePar”. Then a list of parameters is defined for this process the first element is used to represent the subprocess identifier, than a list of \( z \) elements is used to represent the start events in the subprocess, following a list of \( y \) elements refers to the end events inside the subprocess, then a list of \( n \) parameters is introduced to represent the incoming edges identifiers, followed by a list of \( m \) parameters to represent the outcoming flows identifiers. Two more parameters are also introduced the identifier of the subprocess that contains the subprocess itself and the last one is the pool identifier where the process lies. Internally the process synchronize itself with the CSP process related to all the incoming flows. This is done
through the parallel execution of enter events for each incoming flows where each event is characterized by the corresponding FlowID and the PoolID. At this point the process call the second process to enable the subprocess elements. When the second process is terminates the process activates all the outcoming flows with the parallel execution of m esc events characterized by corresponding FlowIDs and PoolID. After all the outcoming flows are started the process terminate successfully. Instead of such main behavior the CSP process related to the task BPMN element can also be drastically terminated if abort event happened. In such case terminate event will be fired. The terminate event is characterized by the value of PoolID or the value of PoolID and SubProcID according to the position of the task, in a subprocess or in a main process respectively.

Subprocess With Intermediate Error Event

As introduced in the event section, it is also possible that the subprocess is characterized by error events. This means that the flow of subprocess can be properly governed.

The rule to transform the BPMN simple subprocess construct enriched by intermediate error event produces a CSP processes. So the general idea is that the incoming flows has to be activated and then the subprocess elements are executed. If after the execution of some elements an error occurs, this means that the elements in the subprocess have to be terminated without be executed and the outcoming flows related to the error are fired. After that the whole subprocess is terminated.

Figure 9.41 provides the detailed transformation rule for a subprocess. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubprocessID contains the identifier of the subprocess;
- SubProcID contains the identifier of the subprocess that contain the subprocess under analysis, it could be null if the element is not contained in a subprocess;
- IncomingEdgesIDs contains the identifiers of the incoming flows;
- errorID contains the identifiers of the error;
PoolID = “poolID”;
SubprocessID = “subID”;
SubProcID = “subProcID”;
IncomingEdgesIDs = “flowInID₁,...,flowInIDₙ”;
OutcomingEdgesIDs = “flowOutID₁,...,flowOutIDₘ”;  
NumberOfParSub = “n SubProc”;
SubstartID = “substartID₁,...,substartIDₓ”;  
SubendID = “subendID₁,...,subendIDᵧ”;  

\[
\text{subID}(\text{subID}, \text{substartID₁}, ..., \text{substartIDₓ}, \text{subendID₁}, ..., \text{subendIDᵧ}, \text{poolID}) = \\
(\text{enable} \cdot \text{substartID₁} \cdot \text{poolID} | | ... | | \text{enable} \cdot \text{substartIDₓ} \cdot \text{poolID}) \rightarrow \\
(\text{enable} \cdot \text{subendID₁} \cdot \text{poolID} | | ... | | \text{enable} \cdot \text{subendIDᵧ} \cdot \text{poolID}) \rightarrow \text{Skip};
\]

\[
\text{ParallelSubID}(\text{subID}, \text{substartID₁}, ..., \text{substartIDₓ}, \text{subendID₁}, ..., \text{subendIDᵧ}, \text{subProcID}, \text{poolID}) = \\
(\| \ x : \{0..\text{NumberOfPar}\} \ @ \text{subID}(\text{subprocessID}, \text{substartID₁}, ..., \text{substartIDₓ}, \text{subendID₁}, ..., \text{subendIDᵧ}, \text{poolID}) )
\]
\[
\square ( \text{terminate} \cdot \text{poolID} \rightarrow \text{Skip}; )
\]
\[
\square ( \text{terminate} \cdot \text{poolID} \cdot \text{subProcID} \rightarrow \text{Skip}; )
\]

\[
\text{MultiInstanceParSubID}(\text{subID}, \text{substartID₁}, ..., \text{substartIDₓ}, \text{subendID₁}, ..., \text{subendIDᵧ}, \text{flowInID₁}, ..., \text{flowInIDₙ}, \text{flowOutID₁}, ..., \text{flowOutIDₘ}, \text{subProcID}, \text{poolID}) = \\
(\langle \text{enter} \cdot \text{flowInID₁} \cdot \text{poolID} | | ... | | \text{enter} \cdot \text{flowInIDₙ} \cdot \text{poolID} )
\]
\[
\rightarrow \text{ParallelSubID}(\text{subprocessID}, \text{substartID₁}, ..., \text{substartIDₓ}, \text{subendID₁}, ..., \text{subendIDᵧ}, \text{subProcID}, \text{poolID})
\]
\[
\rightarrow (\text{esc} \cdot \text{flowOutID₁} \cdot \text{poolID} | | ... | | \text{esc} \cdot \text{flowOutIDₘ} \cdot \text{poolID} ) \rightarrow \text{Skip}; )
\]
\[
\square ( \text{terminate} \cdot \text{poolID} \rightarrow \text{Skip}; )
\]
\[
\square ( \text{terminate} \cdot \text{poolID} \cdot \text{subProcID} \rightarrow \text{Skip}; )
\]

Figure 9.40: Mapping Subprocess with Multi Instance in Parallel.
• OutErrorEdgesIDs contains the identifiers of the outcoming flows binder to the error;

• OutcomingEdgesIDs contains the identifiers of the outcoming flows;

• SubstartID contains the identifiers of the start events implemented in the subprocess;

• SubendID contains the identifiers of the end events implemented in the subprocess.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by subprocessID. Then a list of parameters is defined for this process the first element is used to represent the subprocess identifier, the second is a list of z elements used to represent the start events in the subprocess, the thirds is a list of y elements refers to the end events inside the subprocess, the fourth element refers to the intermediate error event, than a list of j parameters is introduced to represent the outcoming flows identifiers in the case an error occur, then a list of n parameters is introduced to represent the incoming edges identifiers, followed by a list of m parameters to represent the outcoming flows identifiers and a last parameter is used to provide the pool identifier where the process lies. Internally the process synchronize itself with all the incoming flows. This is done through the parallel execution of an enter event for each incoming flow where each event is characterized by the corresponding FlowID and the PoolID. Then the process prosecutes with a parallel execution of the start events enable characterized by the SubstartIDs and the PoolID. At this point the corresponding BPMN elements inside the subprocess will run until the error occur. They are represented by other CSP process which are outside the scope of the subprocessID process. In this case after receiving an error on the error channel all the outcoming flows related to the error with the parallel execution of j esc events characterized by corresponding FlowIDs and PoolID are executed. After all the outcoming flows are fired and the process is successfully terminated. Termination is also considered in this case as well as for the other BPMN elements.

Mapping Gateway

As well as for the other BPMN elements we provide the transformation rules related to BPMN gateways. Notable in such rules the use of the conditions that are base on data or events. We assume them as boolean variables output of the evaluation already done. As matter of fact our mapping does not deal with data.
PoolID = "poolID";
SubProcID = "subProcID";
SubprocessID = "subprocessID";
ErrorID = "errorID";
OutErrorEdgesIDs = "flowOutErrID_1, ..., flowOutErrID_j";
IncomingEdgesIDs = "flowInID_1, ..., flowInID_n";
OutcomingEdgesIDs = "flowOutID_1, ..., flowOutID_m";
SubstartID = "substartID_1, ..., substartID_z";
SubendID = "subendID_1, ..., subendID_y";

subprocessID(subprocessID, substartID_1, ..., substartID_z, subendID_1, ..., subendID_y, flowInID_1, ..., flowInID_n, flowOutErrID_1, ..., flowOutErrID_j)
flowOutID_1, ..., flowOutID_m, poolID) =
((enter.flowInID_1.poolID || ... || enter.flowInID_n.poolID) →
(enable.substartID_1.poolID || ... || enable.substartID_z.poolID) →
((error?errorID → (esc.flowOutErrID_j.poolID || ... ||
esc.flowOutErrID_1.poolID) → terminate.poolID.subProcID → Skip;) →
(esc.flowOutErrID_j.poolID || ... || esc.flowOutID_m.poolID) → Skip;))
→
(terminate.poolID → Skip;)
→
(terminate.poolID.subProcID → Skip;)

Figure 9.41: Mapping Subprocess with Intermediate Error Event at the Border.
In the following section we discuss about data-base exclusive gateway, inclusive gateway and parallel gateway. All of them are observed both in splitting and merging modality. No rules are introduced for exclusive event based gateways. They can be easily derived from exclusive data-base where the condition is governed by an event. The complex gateway can be trivial derived from the other mapping via a combination of the condition that govern the flow.

**Data-based Exclusive Gateway**

The rule to transform the BPMN exclusive data-based gateway in splitting modality produces two CSP processes. The first one is a CSP process that represents the input and output flows. The second process binds the decision to the incoming and outcoming process flows. So the general idea is that the second process governs the flow of the process according to the current state of the condition. Only one of the alternatives will be chosen. As matter of fact it routes the sequence flow to exactly one of the outgoing branches based on conditions. When this process returns the outcoming flow are fired and the whole gateway is successfully terminated.

Figure 9.42 provides the detailed transformation rule for exclusive data-based gateway in splitting modality. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the gateway, it could be null if the element is not contained in a subprocess;
- GatID contains the identifier of the gateway;
- IncomingEdgeID contains the identifier of the incoming flow;
- OutcomingEdgesIDs contains the identifiers of the outcoming flows;
- OutcomingConds contains the conditions evaluation for the output on a specific flow;
- InclusiveTypeGaID contains the property of the task to be inclusive; the associate value depend from the type of inclusiveness that is implemented. It can be null if the gateway don’t implement inclusive choice.

Given the premises the resulting two CSP processes are defined in the following way.
1. The first process is named Trans. Three parameters are defined for this process: the first one represents the incoming flow, the second the outcoming flows identifier, and the last one the pool identifier. Internally the process fires the event enter characterized by the value of IncomingEdgeID and PoolID and sets the value of InclusiveTypeGaID property. This is followed by the event enter characterized by the value of OutcomingEdgesIDs and PoolID.

2. The second process is named with the value assumed by GatID prefixed with the string “dataExS”. Then a list of parameters is defined for this process: the first element is used to represent the gateway identifier, the second parameter is introduced to represent the incoming edge identifier, followed by a list of m parameters to represent the outcoming flows identifiers, than a parameters to identified the subprocess that contains the gateway, and a last parameter is used to provide the pool identifier. Internally the process executes a conditional choice via the case. Case is a key word in PAT input language and \( \text{cond}_1, \text{cond}_2, \ldots, \text{cond}_{m-1} \) are Boolean expressions already evaluated. If \( \text{cond}_1 \) is true, then the first process will be executed and the outcoming flow identifier is the first outcoming flow in the OutcomingEdgesIDs. Otherwise, if \( \text{cond}_2 \) is true, then the outcoming flow choose for the Trans process will be the second element in the OutcomingEdgesIDs. The condition is evaluated one by one until one which is true is found. In case no condition is true, the Trans process will be executed according to the default condition of BPMN. Instead of such main behavior the CSP process related to the gateway BPMN element can also be drastically terminated if abort event happened. In such case terminate event will be fired. The terminate event is putting in exclusive choice with the main process and it is characterized by the value of PoolID or the value of PoolID and SubprocessID according to the position of the gateway, in a subprocess or in a main process respectively.

The rule to transform the BPMN exclusive data-based gateway in merging modality produces two CSP processes. The first one is a CSP process that represents the input and output flows. The second binds the decision to the incoming and outcoming process flows. So the general idea is that the second govern the flow of the process according to the current state of the condition. Only one of the alternatives will be chosen. As matter of fact it routes the sequence flow to exactly one of the incoming branches based on conditions. When this process returns the incoming flows are fired and the whole gateway is successfully terminated.

Figure 9.43 provides the detailed transformation rule for exclusive data-based gateway in merging modality. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:
PoolID = “poolID”;
SubProcID = “subProcID”;
GatID = “gatID”;
IncomingEdgeID = “flowInID”;
OutcomingEdgesIDs = “flowOutID1,...,flowOutIDm”;
OutcomingCond = “cond1,...,condm-1”;
InclusiveTypeGaID = "GatIDType";

Trans(flowInID, out, poolID) =
enter.flowInID.poolID → {InclusiveTypeGaID = GatIDType} → enter.out.poolID → Skip;

dataExSGatID(gatID, flowInID,
flowOutID1, ... ,flowOutIDm, subProcID, poolID) =
case {
cond1 : Trans(flowInID, flowOutID1, poolID)
cond... : Trans(flowInID, ..., poolID)
condm−1 : Trans(flowInID, flowOutIDm−1, poolID)
default : Trans(flowInID, flowOutIDm , poolID)
}

□ (terminate.poolID → Skip; )
□ (terminate.poolID.subProcID → Skip; )

Figure 9.42: Mapping Exclusive (Data-Based) Splitting.
• PoolID contains the identifier of the pool;

• SubProcID contains the identifier of the subprocess that contains the gateway, it could be null if the element is not contained in a subprocess;

• GatID contains the identifier of the gateway;

• IncomingEdgesIDs contains the identifiers of the incoming flows;

• OutcomingEdgeID contains the identifier of the outgoing flow;

• InclusiveTypeGaID contains the property of the task to be inclusive; the associate value depend from the type of inclusiveness that is implemented. It can be null if the gateway don’t implement inclusive choice;

• IncomingCon contains the conditions evaluation for the input flows.

Given the premises the resulting two CSP processes are defined in the following way.

1. The first process is named Trans. Three parameters are defined for this process the first one represents one of the incoming flow choose in the IncomingEdgesIDs set, the second one the outcoming flow identifier and the last one the pool identifier. Internally the process fire the event esc characterized by one of the value of IncomingEdgesIDs and PoolID and set the value of InclusiveTypeGaID peroperty. This is followed by the event esc characterized by the value of OutcomngEdgeID and PoolID.

2. The second process is named with the value assumed by GatID prefixed with the string “dataExM”. Then a list of parameters is defined for this process the first element is used to represent the gateway identifier, followed by a list of m parameters to represent the incoming flows identifiers, the third parameters is introduced to represent the outcoming edge identifier, than a parameters to identified the subprocess that contains the gateway, and a last parameter is used to provide the pool identifier. Internally the process execute a conditional choice via the case. Case is a key word in PAT input language and \( cond_1, cond_2, \ldots, cond_{m-1} \) are Boolean expressions already evaluated. If \( cond_1 \) is true, then the first process will be executes and the incoming flow identifier is the first flow in the IncomingEdgesIDs. Otherwise, if \( cond_2 \) is true, then the outcoming flow choose for the Trans process will be the second element in the IncomingEdgesIDs. The condition is evaluated one by one until one which is true is found. In case no condition is true, the Trans process will be executed according to the default condition of BPMN. Instead of such main behavior the CSP process related to the
gateway BPMN element can also be drastically terminated if abort event
happened. In such case terminate event will be fired. The terminate event
is putting in exclusive choice with the main process characterized by the
value of PoolID or the value of PoolID and SubprocessID according to the
position of the gateway, in a subprocess or in a main process respectively.

\[
\begin{align*}
PoolID &= \text{"poolID";} \\
SubProcID &= \text{"subProcID";} \\
GatID &= \text{"gatID";} \\
OutcomingEdge &= \text{"flowOutID";} \\
IncomingEdgesIDs &= \text{"flowInID1, ..., flowInID}_m; \\
IncomingCond &= \text{"cond1, ..., cond}_m; \\
InclusiveTypeGaID &= \text{"GatIDType";}
\end{align*}
\]

\[
\text{Trans(} \text{in}, \text{flowOutID, poolID) =} \\
\text{esc. in. poolID} \rightarrow \{ \text{InclusiveTypeGaID = GatIDType}\} \rightarrow \text{esc. flowOutID, poolID} \rightarrow \text{Skip;}
\]

\[
\text{dataExgatID(gatID, flowInID}_1, ..., \text{flowInID}_m, \\
\text{flowOutID, subProcID, PoolID) =} \\
\text{case}\{ \\
\text{cond1 : Trans(flowInID}_1, \text{flowOutID, poolID) } \\
\text{cond... : Trans(..., flowOutID, poolID) } \\
\text{default : Trans(flowInID}_m, \text{flowOutID, poolID) } \\
\}
\]

\[
\begin{align*}
\Box (\text{terminate.poolID} \rightarrow \text{Skip}; ) \\
\Box (\text{terminate.poolID, subProcID} \rightarrow \text{Skip}; )
\end{align*}
\]

Figure 9.43: Mapping Exclusive (Data-Based) Merging.

**Inclusive Gateway**

The rule to transform the BPMN inclusive gateway in splitting modality produces
two CSP processes. The first one is a CSP process that represents the input and
output flows. The second process binds the decision to the incoming and outcom-
ing process flows. So the general idea is that the second process govern the flow of
the process according to the current state of the condition. In this case more than
one of the alternatives will be chosen. As matter of fact it routes the sequence flow
to all combinations of the paths based on conditions. When this process returns
the outcoming flow are fired and the whole gateway is successfully terminated.

Figure 9.44 provides the detailed transformation rule for inclusive gateway in
splitting modality. In the rule premises we show the graphical construct representa-
tion together with the related attributes. In particular the construct is character-
ized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the gate-
way, it could be null if the element is not contained in a subprocess;
- GatID contains the identifier of the gateway;
- IncomingEdgeID contains the identifier of the incoming flow;
- OutcomingEdgesIDs contains the identifiers of the outcoming flows;
- OutcomingCon contains the conditions evaluation for the output.

Given the premises the resulting two CSP processes are defined in the follow-
ing way.

1. The first process is named Trans. Three parameters are defined for this pro-
cess the first one represents the incoming flow identifier, the second one of
the outcoming flows identifier choosed in the OutcomingEdgesIDs set and
the last one the pool identifier. Internally the process fire the event enter
characterized by the value of IncomingEdgeID and PoolID, it is followed
by the event enter characterized by the by one of the value of OutcomingEd-
geIDs and PoolID.

2. The second process is named with the value assumed by GatID prefixed
with the string “dataInS”. Then a list of parameters is defined for this pro-
cess the first element is used to represent the gateway identifier, the second
parameter is introduced to represent the incoming edge identifier, followed
by a list of n parameters to represent the outcoming flows identifiers, than
a parameters to identified the subprocess that contains the gateway, and a
last parameter is used to provide the pool identifier. Internally the process
execute a conditional choice via a sequence of if then constructs. For each
condition if it is evaluate to true the related Trans process will be executed
taking in input the specific outcoming edge. Differently from the exclusive
gateway in this case all the condition are evaluated. Instead of such main
behavior the CSP process related to the gateway BPMN element can also
be drastically terminated if abort event occur. In such case terminate event will be fired. The terminate event is putting in exclusive choice with the main process and it is characterized by the value of PoolID or the values of PoolID and SubprocessID according to the position of the gateway, in a subprocess or in a main process respectively.

Figure 9.44: Mapping Inclusive Gateway (splitting).

The rule to transform the BPMN inclusive gateway in merging modality produces two CSP processes. The first one is a CSP process that represents the input and output flows. The second process binds the decision to the CSP process related to the incoming and outcoming flows. So the general idea is that the second process govern the flow of the process according to the evaluation of the condition. In this case more than one of the alternatives will be chosen. As matter of fact it routes the sequence flow to all combinations of the paths based on conditions. When this process returns the outcoming flows are fired and the whole gateway is successfully terminated.

Figure 9.45 provides the detailed transformation rule for inclusive gateway in merging modality. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:
• PoolID contains the identifier of the pool;

• SubProcID contains the identifier of the subprocess that contain the gateway, it could be null if the element is not contained in a subprocess;

• GatID contains the identifier of the gateway;

• IncomingEdgesIDs contains the identifiers of the incoming flows;

• OutcomingEdgeID contains the identifier of the outcoming flow;

• IncomingCon contains the conditions evaluation for the input flows.

Given the premises the resulting two CSP processes are defined in the following way.

1. The first process is named Trans. Three parameters are defined for this process the first one represents one of the incoming flow choose in the IncomingEdgesIDs set, the second refers to the outcoming flow identifier and the last one the pool identifier. Internally the process fire the event esc characterized by one of the value of IncomingEdgesIDs and PoolID, it is followed by the event esc characterized by the value of OutcomingEdgeID and PoolID.

2. The second process is named with the value assumed by GatID prefixed with the string “IncM”. Then a list of parameters is defined for this process the first element is used to represent the gateway identifier, followed by a list of n parameters to represent the incoming flow identifiers, the third parameter is introduced to represent the outcoming edge identifier, than a parameters to identified the subprocess that contains the task, and the last parameter is used to provide the pool identifier to the process. Internally the process execute a conditional choice via a set of if then constructs. For each condition if it is evaluate to true the related Trans process will be executed. It takes in input the specific incoming edge. Differently from the exclusive gateway in this case all the conditions are evaluated. Instead of such main behavior the CSP process related to the gateway BPMN element can also be drastically terminated if abort event occur. In such case terminate event will be fired. The terminate event is putting in exclusive choice with the main process behavior and it is characterized by the value of PoolID or the value of PoolID and SubprocessID according to the position of the gateway, in a subprocess or in a main process respectively.
PoolID = “poolID”;
SubProcID = “subProcID”;
GatID = “gatID”;
OutcomingEdgeID = “flowOutID”;
IncomingEdgesID = “inputFlowID₁, ..., inputFlowIDₙ”; 
IncomingCond = “cond₁, ..., condₙ”;

\[
\text{Trans}(\text{in, flowOutID, poolID}) = \\
\text{esc.in.poolID} \to \text{esc.flowOutID.poolID} \to \text{Skip};
\]

\[
\text{IncMID}(\text{gatID}, \text{inputFlowID₁}, ..., \text{inputFlowIDₙ}, \\
\text{flowOutID}, \text{subProcID}, \text{PoolID}) = \\
\begin{array}{l}
\text{if} \ (\text{cond₁}) \{ \text{Trans}(\text{inputFlowID₁, flowOutID, poolID})\}; \\
\text{if} \ (\text{cond...}) \{ \text{Trans}(\text{inputFlowID..., flowOutID, poolID})\}; \\
\text{if} \ (\text{condₙ}) \{ \text{Trans}(\text{inputFlowIDₙ, flowOutID, poolID})\} \\
\quad \Box \ (\text{terminate.poolID} \to \text{Skip};) \\
\quad \Box \ (\text{terminate.poolID.subProcID} \to \text{Skip};)
\end{array}
\]

Figure 9.45: Mapping Inclusive Gateway (merging).
Parallel Gateway

The rule to transform the BPMN parallel gateway in splitting modality produces a CSP processes. So the general idea is to support more than one parallel control flow as output to the gateway. When this process returns the outcoming flows are fired, more than one token are distributed on the main BPMN process and the whole gateway is successfully terminated.

Figure 9.46 provides the detailed transformation rule for parallel gateway in splitting modality. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the gateway, it could be null if the element is not contained in a subprocess;
- GatID contains the identifier of the gateway;
- IncomingEdgeID contains the identifier of the incoming flow;
- OutcomingEdgesIDs contains the identifiers of the outcoming flows.

Given the premises the resulting CSP processes is defined in the following way. The CSP process is named with the value assumed by GatID prefixed with the string “AndSplit”. Then a list of parameters is defined for this process the first element is used to represent the gateway identifier, the second parameters is introduced to represent the incoming edge identifier, followed by a list of m parameters to represent the outcoming flows identifiers, than a parameters to identified the subprocess that contains the gateway, and a last parameter is used to provide the pool identifier. Internally the process fires the corresponding BPMN input flow represented by the CSP event named enter and characterized by the value of IncomingEdgeID and PoolID. The process prosecutes firing in parallel the event named enter and characterized by the value of OutcomingEdgesIDs and PoolID. After all the process terminate successfully. Instead of such main behavior the CSP process related to the gateway BPMN element can also be drastically terminated if abort event happened. In such case terminate event will be fired. The terminate event is putting in exclusive choice with the main project governed by the case and it is characterized by the value of PoolID or the value of PoolID and SubprocessID according to the position of the gateway, in a subprocess or in a main process respectively.
AndSplitGatID
\[(\text{gatID}, \text{flowInID}, \text{flowOutID}_1, \ldots, \text{flowOutID}_m, \text{subProcID}, \text{PoolID}) \rightarrow \\
(\text{enter}.\text{flowInID}.\text{poolID} \rightarrow \\
(\text{enter}.\text{flowOutID}_1, \text{poolID} \parallel \ldots \parallel \text{enter}.\text{flowOutID}_m, \text{poolID}))
\]
\[\square \ (\text{terminate}.\text{poolID} \rightarrow \text{Skip}; )
\]
\[\square \ (\text{terminate}.\text{poolID}.\text{subProcID} \rightarrow \text{Skip}; )
\]

Figure 9.46: Mapping Parallel Gateway (splitting).

Parallel Gateway Merging

The rule to transform the BPMN parallel gateway in merging modality produces a CSP process. So the general idea is support more than one parallel control flow in input to the gateway. When this process returns the incoming flows are fired, and more than one token is collected from the main BPMN process flow and the whole gateway is successfully terminated.

Figure [9.47] provides the detailed transformation rule for parallel gateway in merging modality. In the rule premises we show the graphical construct representation together with the related attributes. In particular the construct is characterized by the following attributes:

- PoolID contains the identifier of the pool;
- SubProcID contains the identifier of the subprocess that contain the gateway, it could be null if the element is not contained in a subprocess;
- GatID contains the identifier of the gateway;
- IncomingEdgesIDs contains the identifiers of the incoming flows;
- OutcomingEdgeID contains the identifier of the outcoming flow.

Given the premises the resulting CSP process is defined in the following way. The process is named with the value assumed by GatID prefixed with the string “AndJoin”. Then a list of parameters is defined for this process the first element is
used to represent the gateway identifier, followed by a list of n parameters to repre-
sent the incoming flows identifiers, the third parameters is introduced to represent
the outcoming edge identifier, than a parameters to identified the subprocess that
contains the task, and a last parameter is used to provide the pool identifier to the
process. Internally the process firing in parallel the event named esc and charac-
terized by the values of IncomingEdgesIDs and PoolID. The process prosecutes
firing the corresponding BPMN output flow represented by the CSP event named
esc and characterized by the value of OutcomingEdgeID and PoolID. After all the
process terminate successfully. Instead of such main behavior the CSP process
related to the gateway BPMN element can also be drastically terminated if abort
event happened. In such case terminate event will be fired. The terminate event
is putting in exclusive choice with the main process characterized by the value of
PoolID or the values of PoolID and SubprocessID according to the position of the
gateway, in a subprocess or in a main process respectively.

\[
\begin{align*}
PoolID &= "poolID"; \\
SubProcID &= "subProcID"; \\
GatID &= "gatID"; \\
OutcomingEdge &= "flowOutID"; \\
IncomingEdgesIDs &= "flowInID_1,...,flowInID_m"; \\
\end{align*}
\]

Figure 9.47: Mapping Parallel Gateway (merging).

### Mapping Data Object

The rule to transform the BPMN data object construct produces a CSP variable.
So the general idea is that for each data object we refer to such variable that
unambiguously refer to it. Figure 9.48 provides the detailed transformation rule
for a pool. In the rule premises we show the graphical construct representation
 together with the related attributes. In particular the construct is characterized by
the following attributes:
• DataObjID contains the identifier of the data object;

• DataObjName contains the name of the data object.

Given the premises the resulting CSP variable is named with the name of the data object and it has the value of the data object identifier.

\[
\begin{align*}
\text{DataObjID} &= \text{"DataObjID";} \\
\text{DataObjName} &= \text{"DataObjName";} \\
\end{align*}
\]

\[
\text{var dataObjName} = \text{DataObjID} ;
\]

Figure 9.48: Mapping Data Object.

**Mapping Message Flow**

A Message Flow tells us what messages flow across organizational boundaries, so between participants in the service delivery. To map such scenario we introduce CSP processes that communicates through messages passing on channels. In PAT a channel is declared as follows.

\[
\text{channel chname chdimension;}
\]

Where channel is a key word for declaring channels only, chname is the channels names and chdimension is the channel buffer size this means that sends/receives messages are synchronously. Channel input/output is written in a similar way as simple event prefixing. \( c!a \) is the channel output and \( c?x \) is the channel input. Where \( c \) is a channel, \( a \) is expression which evaluates to values (at run time) and \( x \) is (local) variables which take the input values.

Most of the element already discussed, such as events and tasks can be enriched with messages that can be sent or received. We don’t provide a detailed discussion of each of them, but we report a list of such elements and provide related transformation rules.

• Figure 9.49 provides the detailed transformation rule for a send task.

• Figure 9.50 provides the detailed transformation rule for a receive task.
• Figure 9.51 provides the detailed transformation rule for a start event receiving message.

• Figure 9.52 provides the detailed transformation rule for an end event sending message.

• Figure 9.53 provides the detailed transformation rule for an intermediate event sending.

• Figure 9.54 provides the detailed transformation rule for an intermediate event receiving.

• Figure 9.55 provides the detailed transformation rule for a loop task sending.

• Figure 9.56 provides the detailed transformation rule for a loop task receiving.

• Figure 9.57 provides the detailed transformation rule for multiple instance task in sequence sending.

• Figure 9.58 provides the detailed transformation rule for multiple instance task in sequence receiving.

• Figure 9.59 provides the detailed transformation rule for multiple instance task in parallel sending.

• Figure 9.60 provides the detailed transformation rule for multiple instance task in parallel receiving.

For what concern the sending element in the rule premises we show the graphical construct representation together with the related attributes. In particular we enrich the premisses already discussed with the following attributes:

• OutcomingMessages contains the identifiers of the outcoming messages corresponding to message flow in BPMN;

• DataObject contains the identifiers of the outcoming data object corresponding to messages that are exchanged;

• CollaborationTypeMsgID contains the property value of the element that can be communication, collaboration or strong integration.

• TransparencyMedMsgID contains the property value of the element to be transparent according to the interaction with a mediator sending a message.
• SharingDatMsgID contains the property value of the element to be sharing according to the interaction with a knowledge base sending a message.

• chType contains the different type of channel that can be implemented such as communication, collaboration and strong integration.

Given the premises the resulting CSP process is enriched by channel instantiation and the sending of data types specified on the channel instantiated.

For what concern the receiving element in the rule premises we show the graphical construct representation together with the related attributes. In particular we enrich the premises already discussed with the following attributes:

• IncomingMessages contains the identifiers of the incoming messages corresponding to message flow in BPMN.

• CreativeTypeMsgID contains the property of the element to be creative sending a message.

• ProactiveTypeMsgID contains the property of the element to be proactive sending a message.

• ReactiveTypeMsgID contains the property of the element to be reactive sending a message.

Given the premises the resulting CSP process is enriched by receiving of data types on the channel already instantiated with the sending activity.
PoolID = "poolID";
SubProcID = "subProcID";
TaskID = "taskID";
IncomingEdges = "flowInID_1,...,flowInID_n";
OutcomingEdges = "flowOutID_1,...,flowOutID_m";
OutcomingMessages = "msgOutID";
DataObject = "dtObjectName";
chType = "com" or "coll" or "sInt";
CollaborationTypeMsgID = "ColElID";
TransparencyMedMsgID = "TraElID";
SharingDatMsgID = "SharElID";

Channel chTypemsgOutID 0;

SendTaskID(taskID, flowInID_1, ..., flowInID_n, dtObjectName
flowOutID_1, ..., flowOutID_m, subProcID, poolID) =
((enter.flowInID_1.poolID || ... || enter.flowInID_n.poolID) →
enable.taskID.participantID → work.taskID.participantID →
chTypemsgOutID!dtObjectName → {{CollaborationTypeMsgID = ColElID,
   TransparencyMedMsgID = TraElID, SharingDatMsgID = SharElID} →
   (esc.flowOutID_1.poolID || ... || esc.flowOutID_m.poolID) → Skip; )
   (terminate.poolID → Skip; )
   (terminate.poolID, subProcID → Skip; )

Figure 9.49: Mapping Send Task.
var TaskIDTransparancy = "traspValue";

ReceiveTaskID(taskID, flowInID1, ..., flowInIDn, 
flowOutID1, ..., flowOutIDm, subProcID, poolID) =
((enter.flowInID1.poolID || ... || enter.flowInIDn.poolID)
→ enable.taskID.participantID
→ channelType.msgIn?msgIn → {CreativeTypeMsgID = CreElID,
ProactiveTypeMsgID = ProElID, ReactiveTypeMsgID = ReElID}
→ work.taskID.participantID
→ (esc.flowOutID1.poolID || ... || esc.flowOutIDm.poolID) → Skip;)
□ (terminate.poolID → Skip;)
□ (terminate.poolID.subProcID → Skip;)

Figure 9.50: Mapping Receive Task.
PoolID = “poolID”;
SubProcID = “subProcID”;
StartID = “startID”;
IncomingEdges = “flowInID₁, ..., flowInIDₙ”;
OutcomingEdges = “flowOutID₁, ..., flowOutIDₘ”;  
IncomingMessages = “msgInID”;  
CreativeTypeMsgId = “CreElID”;  
ProactiveTypeMsgId = “ProElID”;  
ReactiveTypeMsgId = “ReElID”;  

ReceiveStartID

(trainID, flowInID₁, ..., flowInIDₙ,  
flowOutID₁, ..., flowOutIDₘ, subProcID, poolID) =  
((enter.flowInID₁.poolID || ... || enter.flowInIDₙ.poolID)  
→ enable.startID.participantID  
→ channelType(msgInID,msgIn) → {CreativeTypeMsgId = CreElID,  
ProactiveTypeMsgId = ProElID, ReactiveTypeMsgId = ReElID}  
→ (esc.flowOutID₁.poolID || ... || esc.flowOutIDₘ.poolID) → Skip; )  
□ (terminate.poolID → Skip; )  
□ (terminate.poolID,subProcID → Skip; )  

Figure 9.51: Mapping Receiving Start Event.
PoolID = “poolID”;
SubProcID = “subProcID”;
StartID = “startID”;
OutcomingEdges = “flowOutID₁, ..., flowOutIDᵢ”;
OutcomingMessages = “msgOutID”;
DataObject = “dtObjectName”;
chType = “com” or “coll” or “sInt”;
CollaborationTypeMsgId = “ColElID”;
TransparencyMedMsgId = “TraElID”;
SharingDatMsgId = “SharElID”;

Channel chTypemsgOutID 0;

SendStartID(startID, dtObjectName₁, ..., dtObjectNameᵢ, flowOutID₁, ..., flowOutIDᵢ, subProcID, poolID) =
(enable.startID.poolID → chTypemsgOutID!dtObjectName →
{CollaborationTypeMsgId = ColElID, TransparencyMedMsgId = TraElID, SharingDatMsgId = SharElID,
(esc.flowOutID₁.poolID || ... || esc.flowOutIDᵢ.poolID) → Skip; }
□ (terminate.poolID → Skip; )
□ (terminate.poolID.subProcID → Skip; )

Figure 9.52: Mapping End Event Sending.
PoolID = “poolID”;
SubProcID = “subProcID”;
IntermediateID = “intermediateID”;
IncomingEdges = “flowInID1,...,flowInIDn”;
OutcomingEdges = “flowOutID1,...,flowOutIDm”;
OutcomingMessage = “msgOutID”;
DataObject = “dtObjectName”;
chType = “com” or “coll” or “sInt”;
CollaborationTypeMsgId = “ColElID”;
TransparencyMedMsgId = “TraElID”;
SharingDatMsgId = “SharElID”;

Channel chTypemsgOutID 0;

SendIntermediateID(endlD, flowInID1,...,flowInIDn, dtObjectName1,...,
dtObjectName1, flowOutID1,...,flowOutIDm, subProcID, poolID) =
((enter.flowInID1.poolID || ... || enter.flowInIDn.poolID) →
enable.intermediateID.poolID →
{CollaborationTypeMsgId = ColElID,
TransparencyMedMsgId = TraElID, SharingDatMsgId = SharElID} →
chTypemsgOutID!dtObjectName →
(esc.flowOutID1.poolID || ... || esc.flowOutIDn.poolID) → Skip; )
□ (terminate.poolID → Skip; )
□ (terminate.poolID → SubProcID → Skip; )

Figure 9.53: Mapping Intermediate Event Sending.
PoolID = "poolID";
SubProcID = "subProcID";
IntermediateID = "intermediateID";
IncomingEdges = "flowInID_1, ..., flowInID_n";
OutcomingEdges = "flowOutID_1, ..., flowOutID_m";
IncomingMessages = "msgInID";
CreativeTypeMsgId = "CreElID";
ProactiveTypeMsgId = "ProElID";
ReactiveTypeMsgId = "ReElID";

ReceiveIntermediateID(endID, flowInID_1, ..., flowInID_n, flowOutID_1, ..., flowOutID_m, subProcID, poolID) =
((enter.flowInID_1.poolID || ... || enter.flowInID_n.poolID) →
enable.intermediateID.poolID →
chTypeMsgIn?msgIn → \{ CreativeTypeMsgId = CreElID, ProactiveTypeMsgId = ProElID, ReactiveTypeMsgId = ReElID \}
→ (esc.flowOutID_1.poolID || ... || esc.flowOutID_m.poolID) → Skip; )
\( (\text{terminate.poolID → Skip; } ) \)
\( (\text{terminate.poolID.subProcID → Skip; } ) \)

Figure 9.54: Mapping Receiving Intermediate Event.
APPENDIX B - MAPPING FROM BPMN AND CSP

PoolID = "poolID";
SubProcID = "subProcID";
TaskID = "taskID";
IncomingEdges = "flowInID_1,...,flowInID_n";
OutcomingEdges = "flowOutID_1,...,flowOutID_m";
LoopCondition = "Expression";
MaxLoop = "numberofLoop";
OutcomingMessages = "msgOutID";
DataObject = "dObjectName";
chType = "com" or "coll" or "sInt";
CollaborationTypeMsgId = "ColElID";
TransparencyMedMsgId = "TraElID";
SharingDatMsgId = "SharElID";

var TaskIDTransparancy = "traspValue";

Channel chTypemsgOutID 0;

SendLoopTaskID(taskID, dObjectName, subProcID, poolID) =
({ CollaborationTypeMsgId = ColElID,
  TransparencyMedMsgId = TraElID, SharingDatMsgId = SharElID} →
work.taskID.poolID → chTypemsgOutID!dObjectName → if (MaxLoop > 0 & & Expression = "true")
{ minus{MaxLoop = MaxLoop - 1; } }
→ SendWorkLoopTaskID(taskID, dObjectName, subProcID, PoolID))
else { Skip};

SendLoopStartID(taskID, flowInID_1, ..., flowOutID_n, dObjectName, 
flowOutID_1, ..., flowOutID_m, subProcID, poolID) =
{ enter.flowInID_1.poolID || ... || enter.flowInID_n.poolID }
→ enable.taskID.participantID
→ SendWorkLoopTaskID(taskID, dObjectName, subProcID, poolID)
→ (esc.flowOutID_1.poolID || ... || esc.flowOutID_m.poolID) → Skip; 

Figure 9.55: Mapping Loop Task Sending.
PoolID = "poolID";
SubProcID = "subProcID";
IntermediateID = "intermediateID";
IncomingEdges = "flowInID_1,...,flowInID_n";
OutcomingEdges = "flowOutID_1,...,flowOutID_m";
LoopCondition = "Expression";
MaxLoop = "numberOfLoop";
IncomingMessages = "msgIn";
CreativeTypeMsgId = "CreElID";
ProactiveTypeMsgId = "ProElID";
ReactiveTypeMsgId = "ReElID";
TaskIDTransparancy = "traspValue";

\[
\text{ReceiveWorkLoopTaskID}(\text{taskID}, \text{subProcID}, \text{poolID}) = \\
(\text{chTypeMsgInID?msgIn} \rightarrow \{ \text{CreativeTypeMsgId} = \text{CreElID}, \\
\text{ProactiveTypeMsgId} = \text{ProElID}, \text{ReactiveTypeMsgId} = \text{ReElID} \} \rightarrow \\
\text{work.taskID.poolID} \rightarrow \text{if} (\text{MaxLoop} > 0 \&\& \text{Expression} = \text{"true"}) \\
\{ \text{minus}\{ \text{MaxLoop} = \text{MaxLoop} - 1; \} \\
\rightarrow \text{ReceiveWorkLoopTaskID}(\text{taskID}, \text{subProcID}, \text{poolID}) \} \text{ else } \{ \text{Skip}; \} )
\]
\[\square (\text{terminate.poolID} \rightarrow \text{Skip}; )
\]
\[\square (\text{terminate.poolID.subProcID} \rightarrow \text{ Skip}; )
\]

\[
\text{ReceiveLoopTaskID}(\text{taskID}, \text{flowInID}_1, ..., \text{flowInID}_n, \\
\text{flowOutID}_1, ..., \text{flowOutID}_m, \text{subProcID}, \text{poolID}) = \\
((\text{enter.flowInID}_1.poolID \parallel \ldots \parallel \text{enter.flowInID}_n.poolID) \\
\rightarrow \text{enable.taskID.participantID}) \\
\rightarrow \text{ReceiveWorkLoopTaskID}(\text{taskID}, \text{subProcID}, \text{poolID}) \\
\rightarrow (\text{esc.flowOutID}_1.poolID \parallel \ldots \parallel \text{esc.flowOutID}_m.poolID) \rightarrow \text{Skip;} )
\]
\[\square (\text{terminate.poolID} \rightarrow \text{Skip}; )
\]
\[\square (\text{terminate.poolID.subProcID} \rightarrow \text{ Skip}; )
\]

Figure 9.56: Mapping Receiving Loop Task.
APPENDIX B - MAPPING FROM BPMN AND CSP

PoolID = “poolID”;
SubProcID = “subProcID”;
TaskID = “taskID”;
IncomingEdges = “flowInID1, ..., flowInIDn”;
OutcomingEdges = “flowOutID1, ..., flowOutIDm”;
NumberOfSeqActivity = “numberOfSeq”;
OutcomingMessages = “msgOutID”;
DataObject = “dtObjectName”;
chType = “com” or “coll” or “sInt”;
CollaborationTypeMsgId = “ColElID”;
TransparencyMedMsgId = “TraElID”;
SharingDatMsgId = “SharElID”;

var TaskIDTransparancy = “traspValue”;

Channel chType=msgOutID 0;

SendWorkMLoopTaskID(taskID, dtObject, subProcID, poolID) =
if (NumberOfSeqActivity > 0)
{ minus(NumberOfSeqActivity = NumberOfSeqActivity - 1) } →
work.taskID.poolID → { CollaborationTypeMsgId = ColElID,
TransparencyMedMsgId = TraElID, SharingDatMsgId = SharElID } →
chType=msgOutID!dtObject → SendWorkMLoopTaskID
(taskID, dtObject, subProcID, poolID)
} else { Skip; }

SendMultiInstanceSeqTaskID
(taskID, flowInID1, ..., flowInIDn, dtObject, subProcID, poolID) =
((enter.flowInID1.poolID || ... || enter.flowInIDn.poolID)
→ enable.taskID.poolID
→ SendWorkMLoopTaskID
(taskID, dtObject, subProcID, poolID)
→ (esc.flowOutID1.poolID || ... || esc.flowOutIDn.poolID) → Skip; )
| (terminate.poolID → Skip; )
| (terminate.poolID.subProcID → Skip; )

Figure 9.57: Mapping Multiple Instance Task in Sequence Sending.
PoolID = "poolID";
SubProcID = "subProcID";
TaskID = "taskID";
IncomingEdges = "flowInID_1, ..., flowInID_n";
OutcomingEdges = "flowOutID_1, ..., flowOutID_m";
NumberOfSeqActivity = "numberOfSeq";
IncomingMessages = "msgInID";
CreativeTypeMsgId = "CreElID";
ProactiveTypeMsgId = "ProElID";
ReactiveTypeMsgId = "ReElID";
TaskIDTransparancy = "traspValue";

var TaskIDTransparancy = "traspValue";

ReceiveWorkMLoopTaskID(taskID, subProcID, poolID) =
(if (NumberOfSeqActivity > 0)
{ minus(NumberOfSeqActivity) = NumberOfSeqActivity ? 1; }
→ (chTypemsgInID!msgIn → { CreativeTypeMsgId = CreElID,
ProactiveTypeMsgId = ProElID, ReactiveTypeMsgId = ReElID}
→ work.taskID.poolID
→ ReceiveWorkMLoopTaskID(taskID, subProcID, poolID)) else { Skip; }
☐ (terminate.poolID → Skip; )
☐ (terminate.poolID.subProcID → Skip; )

ReceiveMultiInstanceSeqTaskID(taskID, flowInID_1, ..., flowInID_n,
flowOutID_1, ..., flowOutID_m, subProcID, poolID) =
(enter.flowInID_1.poolID || ... || enter.flowInID_n.poolID)
→ enable.taskID.participantID
→ ReceiveWorkMLoopTaskID(taskID, subProcID, poolID)
→ (esc.flowOutID_1.poolID || ... || esc.flowOutID_m.poolID) → Skip; )
☐ (terminate.poolID → Skip; )
☐ (terminate.poolID.subProcID → Skip; )

Figure 9.58: Mapping Multiple Instance Task in Sequence Receiving.
PoolID = “poolID”;
SubProcID = “subProcID”;
TaskID = “taskID”;
IncomingEdges = “flowInID₁,...,flowInIDₙ”;
OutcomingEdges = “flowOutID₁,...,flowOutIDₘ”;
NumberOfParActivity = “numberOfPar”;
OutcomingMessages = “msgOutID”;
DataObject = “dtObjectName”;
chType = “com” or “coll” or “sInt”;
CollaborationTypeMsgId = “ColElID”;
TransparencyMedMsgId = “TraElID”;
SharingDatMsgId = “SharElID”;

var TaskIDTransparancy = “traspValue”;

Channel chTypemsgOutID 0;

SendTaskID(taskID, dtObjectName, poolID) =
{CollaborationTypeMsgId = ColElID,
TransparencyMedMsgId = TraElID, SharingDatMsgId = SharElID} →
chTypemsgOutID!dtObjectName → work.taskID.participantID → Skip;

SendParallelTaskID
(taskID, dtObjectName, subProcID, poolID) =
(∥ x : {0..numberOfPar} @
SendTaskID(taskID, dtObjectName, poolID))
☐ (terminate.poolID → Skip; )
☐ (terminate.poolID.subProcID → Skip; )

SendMultiInstanceParActivityID(taskID, flowInID₁,...,flowInIDₙ,
dObjectName, flowOutID₁,...,flowOutIDₘ, subProcID, poolID) =
(enter.flowInID₁.poolID || ... || enter.flowInIDₙ.poolID)
→ enable.taskID.poolID
→ SendParallelTaskID
(taskID, dtObjectName, subProcID, poolID)
→ (esc.flowOutID₁.poolID || ... || esc.flowOutIDₘ.poolID) → Skip; )
☐ (terminate.poolID → Skip; )
☐ (terminate.poolID.subProcID → Skip; )

Figure 9.59: Mapping Multiple Instance Task in Parallel Sending.
PoolID = "poolID";
SubProcID = "subProcID";
TaskID = "taskID";
IncomingEdges = "flowInID_1,...,flowInID_n";
OutcomingEdges = "flowOutID_1,...,flowOutID_m";
NumberOfParActivity = "numberOfPar";
IncomingMessages = "msgInID";
CreativeTypeMsgId = "CreElID";
ProactiveTypeMsgId = "ProElID";
ReactiveTypeMsgId = "ReElID";
TaskIDTransparancy = "traspValue";

var TaskIDTransparancy = "traspValue";

ReceiveTaskID(taskID, poolID) =
chTypemsgInID?msgIn \rightarrow \{ CreativeTypeMsgId = CreElID,
ProactiveTypeMsgId = ProElID, ReactiveTypeMsgId = ReElID \} \rightarrow
work.taskID.participantID \rightarrow Skip;

ReceiveParallelTaskID(taskID, poolID) (taskID, subProcID, poolID) =
(\| \ x : \{0.. numberOfPar \} \& ReceiveTaskID (taskID, poolID))
\& (terminate.poolID \rightarrow Skip; )
\& (terminate.poolID.subProcID \rightarrow Skip; )

ReceiveMultiInstanceParActivityID(taskID, flowInID_1, ..., flowInID_n,
flowOutID_1, ..., flowOutID_m, subProcID, poolID) =
((enter.flowInID_1.poolID \& \& \& enter.flowInID_n.poolID)
\rightarrow enable.taskID.participantID
\rightarrow ReceiveParallelTaskID(taskID, subProcID, poolID)
\rightarrow (esc.flowOutID_1.poolID \& \& \& esc.flowOutID_m.poolID) \rightarrow Skip; )
\& (terminate.poolID \rightarrow Skip; )
\& (terminate.poolID.subProcID \rightarrow Skip; )

Figure 9.60: Mapping Multiple Instance Task in Parallel Receiving.
Appendix C - BPMN Modeler Tools

Free and Open Source BPMN modelers

Following a list of BPMN modeling tools that has been investigated for the implementation of BP4PA tool. Advantages and disadvantages of their use in a most integrated environment as possible has been considered.

- INTALIO Designer
- ILOG Jviews BPMN Modeler
- TIBCO Business Studio
- ADONIS Community Edition
- BizAgi Process Modeler
- Soyatec eBPMN
- Eclipse STP BPMN Modeler

Trial Demo Tools

To complete our discussion in the following we report a list of tools available under trial demo version. They are commercial tools considered from the beginning not suitable to our research contribution.

- ActiveVOS BPM
- Altova Umodel
- Avolution Abacus
- Appian BPM
• aXway Process Manager
• Barium Live (on-line)
• Oryx (on-line)
• Borland Together
• Casewise Corporate Modeler
• Ekuar BPMN On-line Modeler (on-line)
• Embarcaedro studio
• Lombardi TeamWork
Appendix D - Acronyms

(A)

ACSI  American Customer Satisfaction Index
ACP   Algebra of Communicating Processes

(B)

B2C   Business To Citizen
BSC   Balanced Scorecard
BP    Business Process
BPEL  Business Process Execution Language for Web Services
BPM   Business Process Management
BPMN  Business Process Management Notation
BPR   Business Process Re-engineering

(C)

CAD   Digital Administration Code
CCS   Calculus of Communicating Systems
CNIPA National Center for IT in Public Administration
CRM   Customer Relation Management
CSI   Customer Satisfaction Index
CSP   Communicating Sequential Processes
APPENDIX D - ACRONYMS

(E)
EC   European Commission
ECSI  European Customer Satisfaction Index
EMF   Eclipse Modeling Framework
ERP   Enterprise Resource Planning
EU    European Union

(G)
G2C   Government to Citizen
G2B   Government to Business
G2G   Government to Government
G2E   Government to Employee
G2N   Government to Non Profit

(K)
KPI   Key Performance Indicators

(I)
ICAR  Interoperability and Applicative Cooperation among Regions
ICSI  Italian Customer Satisfaction Index
ICT   Information and Communication Technology
IP    Internet Protocol
IS    Information Society
ISO   International Organization for Standardization
IT    Information Technology

(L)
LAN   Local Area Network
LOTUS Language of Temporal Ordering Specification

(N)
NCSI  Norwegian Customer Satisfaction Index
| (O) | OMG | Object Management Group |
| (P) | PA | Public Administration |
|     | PAT | Process Analysis Toolkit |
| (R) | RUPA | Rete Unitaria della Pubblica Amministrazione |
| (S) | SLA | Service Level Agreement |
|     | SCSB | Swedish Customer Satisfaction Barometer |
|     | SOA | Service-Oriented Architecture |
|     | SOAP | Simple Object Access Protocol |
|     | SPC | Public Internet-working System |
| (T) | TQM | Total Quality Management |
| (U) | UDDI | Universal Description Discovery and Integration |
|     | URL | Uniform Resource Locator |
| (W) | W3C | World Wide Web Consortium |
|     | WMC | Workflow Management Coalition |
|     | WS | Web Service |
|     | WSCDL | Web Service Choreography Description Language |
|     | WSDL | Web Service Definition Language |
(X)

XML  eXtended Markup Language
XSD  XML Schema Definition Language

(Y)

YAWL  Yet Another Workflow Language
PhD in Information Science and Complex Systems
Collection of theses


XXII-10-4 Roberta Alfieri. Computational approaches to model the cell cycle, a biological complex system. January 2010.


XXII-10-6 Barbara Re. Quality of (Digital) Services in e-Government. January 2010.