RAIRO-Theor. Inf. Appl. 46 (2012) 203–229
DOI: 10.1051/ita/2012002

Available online at:
www.rairo-ita.org

FORMAL METHODS TO IMPROVE PUBLIC ADMINISTRATION BUSINESS PROCESSES

Andrea Polini[1], Andrea Polzonetti[1] and Barbara Re[1]

Abstract. Starting from late 90's the public administration has started to employ a quite relevant amount of its budget in developing ICT solutions to better deliver services to citizens. In spite of this effort many statistics show that the mere availability of ICT based services does not guarantee per se their usage. Citizens have continued to largely access services through "traditional" means. In our study we suggest that the highlighted situation is partly due to the fact that relevant domain dependent requirements, mainly related to the delivery process of e-government digital services, are often ignored in the development of e-government solutions. We provide here a domain related quality framework and encoded it in a set of formal statements, so that we can apply automatic verification techniques to assess and improve ICT solutions adopted by public administrations. The paper discusses both the defined quality framework and the tool chain we developed to enable automatic assessment of ICT solutions. The tool chain is based on a denotational mapping of business process modeling notation elements into process algebraic descriptions and to the encoding of quality requirements in linear temporal logic formulas. The resulting approach has been applied to real case studies with encouraging results.

Mathematics Subject Classification. 68.

1. Introduction

Since late 90's the Public Administration (PA) has been an important domain in which to apply Information and Communication Technologies (ICT). The "electronic government" (e-government) words combination entered in the common

Keywords and phrases. Business process formal verification, business process quality assessment, domain dependent property checking, e-government quality framework.

[1] University of Camerino, School of Science and Technology, Computer Science Division, Camerino, Italy. polini.andrea@unicam.it

Article published by EDP Sciences

vocabulary of today's culture. Thanks to the investments made by the PA for the development of the digital society, services are today widely available *via* ICT based solutions and they are referred to as e-Government Digital Services (GDS). Nevertheless, even when they are developed and provided using up-to-date technologies, they are not so widely used by citizens. Many official European Union statistics, such as those provided by Eurostat[2], testify such a situation.

Several reasons can be listed to justify scarce GDS usage. Among the possible causes, such as for instance the digital divide phenomenon, we think that poorly structured Business Processes (BP) play an important role, in particular those related to service delivery. Too often the design and the development of GDSs mainly focus to technological aspects, where requirements coming from social, anthropological and organizational issues are ignored. At the same time delivery processes are typically defined mimicking those already in place within the PA, which nevertheless were conceived for human-to-human and paper based interactions.

Following a design science approach, as illustrated in [17], and in order to investigate the problem, we started a close collaboration with some Italian PAs. These cooperations permitted to highlight a set of domain-dependent requirements, with a special focus on BP related to the delivery of GDSs. Those requirements have been organized in a quality framework, *i.e.*, in a set of quality properties and dimensions. The framework indirectly provides a guide to quality for analysts and engineers engaged in the development of such BPs.

The final result has been an approach for BP design and verification, named BP4PA, which has been implemented in a user-friendly tool where a checklist is used to represent the quality framework and the service delivery is modeled using the Business Process Modeling Notation (BPMN) [37]. In order to enable automatic verification we mapped the checklist and the different BPMN constructs to the input formalims accepted by the selected model checker. In particular, the quality framework has been formally encoded as temporal properties in Linear Temporal Logic (LTL) [11] whereas a BPMN specification is mapped to a Communication Sequential Process (CSP) [18]. BP4PA contributes to reduce the gap existing between formal methods experts, with knowledge on formal tools, and domain experts, which possess the knowledge on the real issues that need to be solved. The approach has been applied to real case studies such as the evaluation of BPs related to the delivery of GDS (*e.g.*, moving from a city to another and new born registration) with encouraging results.

The paper is structured as follows. In the next section we report related works, successively the methodological approach applied during our research is illustrated. Section 4 provides some background material and then Section 5 discusses about domain knowledge in the e-government domain, focusing in particular on the quality framework. Section 6 introduces the proposed approach and Section 7 describes the tool supporting automatic BP verification. Section 8 shows how the approach and the tool have been applied in practice. Finally, Section 9 draws some conclusions and opportunities for future work.

[2] http://epp.eurostat.ec.europa.eu.

2. Related works

To the best of our knowledge the approach we present is the first attempt, within the e-government domain, that tries to provide an easy to use environment both for BP design and BP evaluation with respect to a domain specific quality framework. In the following section we introduce some relevant papers that inspired our work, both in the area of assessment of software system within the e-government domain, and for what concerns the application of formal methods in applied fields of study.

2.1. e-Government business process and maturity

The importance of fully functional e-government characterized by vertical and horizontal integration is clearly discussed in [22]. From our point of view the real benefits which e-government promises can be reached when system integration is fully implemented. The introduction of BP specification can contribute to such an integration. Indeed BP potentialities and capabilities are relatively unexplored in e-government. An interesting survey dealing with issues in the application of BPs to e-government can be found in [21]. A quite comprehensive and interesting discussion on e-government business models is proposed by Janssen and Kuk [19]. In this work the authors address the importance of cross-organizational service delivery and they propose a framework for studying e-government business model that involves design and implementation of delivery strategies for digital services. Our contribution considers the role of business model as addressed by Jansen and Kuk and it introduces characteristics that are process related considering an automatic and systematic approach toward BP improvement and e-government integration.

The area of maturity model for e-government is also related to our work. As a prominent example we refer to the e-GovMM approach already applied in the Chilean PA [32]. Even if it is a relevant contribution making clear the role of BP in PA ecosystems this paper just focuses on maturity assessment and it doesn't provide neither a contribution to the design phase nor an automatic verification strategy.

2.2. User friendly formal methods

In the literature there is a wide interest in applying techniques typically derived and used in the area of software development and evaluation, to the study of e-government organizations and related processes. A first interesting discussion on the topic resulted from a tutorial-workshop event on technological foundations of electronic governance held in Macao [8]. The event explored 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.

Other proposals provide user-friendly techniques hiding the complexity of formal languages and tools. There are for instance interesting approaches aiming at making model checking accessible to a large audience even for people that are

not trained in formal techniques. The work [26] provides a survey on BP verification techniques and an interesting classification of them. Further works in this area, not included in the survey and classified according to the formal model they propose, are: Automata [14], Petri Nets [9, 15, 27, 38, 44, 45] and Process Algebras [12, 13, 20, 31, 33, 39, 46].

The approach that seems to be closer to our work, at least for what concerns the definition of a mapping from BPMN to a formal notation, is discussed in [42]. In such a paper the authors demonstrate how process algebra such as CSP can be applied to model complex workflow systems. The authors have provided a formal semantic for BPMN elements in CSP processes using Z notation [41]. They use it to formally check compatibility of BPMN processes [40]. Nevertheless the proposed solution does not introduce a user-friendly and integrated environment for verification purpose. On the technical side the approach uses the FDR model checker and the ILOG BPMN modeler as stand-alone tools. Even if our work is inspired by the mapping defined in [42], we rewrote it introducing some modifications like, in particular, types for exchanged messages and events. We also imported the mapping within the eclipse[3] modeling framework and integrated development environment (IDE), so to make the successive verification activities easier with a complete integration of the various phases in a unique tool.

3. Methodological approach

Research in information systems is widely dominated by two research paradigms [17]:

- The **behavioural science** paradigm seeks to develop and verify theories that explain or predict human or organizational behaviour.
- The **design science** paradigm seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts.

In our work, we follow the design science paradigm where the fundamental principle is to create ICT artifacts intended to solve identified organizational problems. Artifacts are broadly defined as constructs (vocabulary and symbols), models (abstraction and representations), methods (algorithms and practices), and instantiations (implementations and prototype systems).

Seven guidelines are suggested for building and usage of an artifact.

- **Guideline 1: design as an artifact** – research must produce a viable artifact.
- **Guideline 2: problem relevance** – technology based solutions has to contribute to important and relevant business problems.
- **Guideline 3: design evaluation** – the artifact has to be useful for the specified problem and its evaluation is crucial.
- **Guideline 4: research contributions** – clear and verifiable contributions in the areas of the design artifact, design foundations and/or design methodologies have to be provided.

[3] http://www.eclipse.org.

- **Guideline 5: research rigor** – the artifact itself must be rigorously defined, formally represented, coherent and internally consistent.
- **Guideline 6: design as a search process** – the search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
- **Guideline 7: communication of research** – the results must be presented effectively both to technology-oriented and to management-oriented audiences.

Our work has been carried on according to the design-science paradigm and following its guidelines. A deep and complete discussion of such aspects is out of the scope of the paper, nevertheless for the sake of comprehension it can be useful to provide a short discussion of the approach in light of the introduced paradigm. The specification and the precise encoding of e-government quality framework and the BP4PA tool are some of the provided artifacts. All of them are technology-based solutions to important and relevant business problems. The utility, quality and effectiveness of produced artifacts have been demonstrated *via* observations, experimentations and descriptive design evaluation methods. Doing that we revised the quality framework several times and we considered an iterative model according to the general software development process for the design and implementation of BP4PA. This work does not provide a novel contribution with respect to theroretical aspects. Nevertheless the proposed approach permits to users, not acquainted with formal methods, to transparently apply them in the e-government application domain. Finally, the communication of our research was mainly scientific. We presented our work to a technology-oriented audience. However the main results of our research were also discussed with PA managers, emphasizing the knowledge required to effectively apply the artifacts within the specific context.

4. Background

4.1. e-Government general overview

The advent 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[4], which in 2000 covered and investigated this topic, e-government has been recognized as one of the major priorities in the innovation process for the PA. In the literature several definitions for e-government have been proposed. Some of them focus on the role of the provided services, where others consider the point of view of the citizens, finally some definitions are mainly interested in the internal processes that are put in place by the PA to provide services using ICT solutions. In this paper we refer to e-government as "the use of ICT in PA combined with organizational changes and new skills in order to improve public services and democratic processes and strengthen support to public policies" [5]. It is nowadays clear that to successfully implement e-government

[4] http://www.lisbonregionsnetwork.eu/.

experiences it is important to look at the domain as a whole without ignoring technical, organizational and social aspects.

The concept of "Service" certainly represents a fundamental abstraction in the e-government domain and it can be considered the primary output of governmental organizations [5]. GDS encapsulates processes and informative resources making services available through ICT. In order to define highly effective strategies for service provision it is important not to split service definition and development from the delivery process with which it will be successively used. This process is typically influenced by a complex network of partnerships and the corresponding activities to be carried on by the different organizations, which then need to be considered in GDS design and implementation. This scenario drives our discussion and motivates the holistic approach to BP design and verification we propose.

The promotion of e-government introduces a lot of advantages related to effectiveness, efficiency and accountability of government. At the same time, it reduces the cost and improves the access to government information and services.

Current maturity of ICT enables a relatively simple development and delivery of GDS. European Commission studies show that 83% of basic GDSs are today available on-line [4]. Comparing these data with the ones reported by similar surveys in 2006 [3] and in 2007 [36] it is evident that PA interest in GDSs is quite strong and quickly raising. Nevertheless a different perception is evident if e-government effectiveness, *i.e.* real usage by citizens, is considered. In this case the result is that many of the available GDSs are seldom accessed and used by citizens. In [1] the authors recognize that "*practical experiences and researches confirm that users' acceptance is not guaranteed per se. Public approval is quite often below what developers expected*". In other words citizens do not use e-government services just because they are available. Many different reasons contribute to the highlighted situation and certainly the well known digital divide phenomenon [28] can be considered as one of the possible causes of low GDSs usage. Nevertheless our impression, corroborated by an informal investigation among the people working in IT departments of PAs, so with high ICT skills, is that other important factors strongly contribute to the underlined scenario.

4.2. Business process and BPMN

Providing successful GDSs is a quite complex task. Requirements come from many different sources and the implementation often foresees the interactions among several PAs. Technically GDSs are modeled and implemented using notations and tools based on the BP concept. "*A BP is a collection of related and structured activities undertaken by one or more organizations in order to pursue some particular goal. Within an organization a BP results in the provisioning of services or in the production of goods for internal or external stakeholders. Moreover BPs are often interrelated since the execution of a BP often results in the activation of related BPs within the same or other organizations*" [23]. In addition to the BP concept Business Process Management (BPM) supports BP experts

providing methods, techniques and software to model, implement, execute and optimize BPs which involve humans, software applications, documents and other sources of information [16].

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 a time-consuming and error-prone activity. Therefore, techniques which help organizations to implement high-quality BPs and to increase process modeling efficiency, have become a highly attractive topic both for industries and for the academy. Certainly many different commercial tools have been developed to support BPM, nevertheless for what concerns the modeling phase they mainly 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 the BPMN [37], the event-driven process chain [25] and the UML activity diagram. There are also more academic related languages, being the Yet Another Work-flow Language [35], based on Petri nets, probably the most prominent example.

In our work we refer to BPMN 1.1, in the following just BPMN. This is certainly the most used language in practical context also given its intuitive graphical notation. Nevertheless BPMN does not have a clearly defined semantic. For this reason, and in order to permit formal verification of BP, we defined a mapping of BPMN constructs to CSP processes.

A BPMN process is made up of BPMN elements composed to express the desired sequence of activities. A BPMN process can include objects, sequence flows and message flows. An object can be an event, an activity, or a gateway. Events are necessary to start, to terminate or to interrupt the defined process. An activity can be an atomic task or a composed activity including other atomic activities. A gateway permits to influence the path followed by a process and for this they are also referred to as routing constructs. A sequence flow links two objects in a process model and denotes a control flow (*i.e.*, ordering) relation. Finally message flows are used to capture the interactions among processes.

Being an Object Management Group (OMG) standard, BPMN is constantly undergoing revisions and extensions. The current release is BPMN 2.0 published in January 2011 [29]. It solves most of the problems of BPMN, in particular it has a more structured and precise meta-model. Nevertheless a formal semantics is still missing, so that the basic principles and the motivations of our approach are still valid, even though the application of the approach to BPMN 2.0 processes will require to revise the defined mapping.

4.3. Process algebras and CSP

Process algebras are used to formally model concurrent and communicating systems and to provide a tool for the high-level description of interactions, communications and synchronization among a collection of independent processes.

They also define composition operations that allow process descriptions to be manipulated and integrated. Moreover the resulting formal representation of the system is suitable for being submitted to formal reasoning techniques to assess whether it satisfies given properties or not. CSP is an event based notation primarily aimed at describing the sequencing of activities within a process and the synchronization (or communication) between different processes. Events represent a form of co-operative synchronization between processes and the environment. Both the processes and the environment may influence the behavior of the other by enabling or refusing certain events or sequences of events. 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 kinds of events are used to implement channels permitting to represent a more structured communication schema among processes. Below we report the grammar in BNF for the CSP language, where P and Q represent generic processes and e represents a generic event.

$$P ::= STOP \mid SKIP \mid e \rightarrow P \mid P \Box Q \mid P \|_A Q \mid P; Q \qquad e ::= x | x \cdot e \qquad (4.1)$$

In detail the process $STOP$ represents a deadlocked process, where the process $SKIP$ is used to represent a successful termination. Process $e \rightarrow P$ denotes a process able to perform the event e, after which it will behave as process P. Process $P \Box Q$ refers to the external choice between processes P and Q. The process is ready to behave either as P or as Q and external factors will make the choice among this two possibilities. Process $P \| Q$ denotes the interleaved parallel composition of processes P and Q. Process $P \|_A Q$ 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. Finally events can be composed using the dot operator ".". Compound events can be used as channels communicating data objects synchronously between the process and the environment. CSP processes are closed terms built up out of actions and other processes using the operators above. The original CSP process algebra includes other operators that we do not present here since they are not strictly useful for the purpose of our presentation. Finally, the operational semantics of CSP is typically given by a set of inference rules defining a mapping from CSP terms to transition systems. We do not report such rules here and the interested reader can refer to [18].

4.4. Temporal logic and 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 given formal specifications or properties, using methods based on sound mathematical tools. Many different formal approaches can be applied to systems verification. Our interest is mainly

in model checking techniques [2], which consist in a systematic and when possible exhaustive, exploration of an operational model to verify wheter it 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-free analysis and generic temporal logic properties, such as those expressed using LTL [11] are typical properties that can be verified on a complex system. Since its first inception many model checking tools have been proposed and developed with the definition of several formalisms to represent the system and languages to specify the properties to verify. In our work we integrate the PAT model checker [34] due to its flexibility and since it uses the CSP formalism as input language.

In the following we define templates of LTL assertions where: "$\Box$" is the temporal operator, which reads as "always", "$\Diamond$" is the temporal operator which reads as "eventually", "U" is the temporal operator which reads as "until" and "X" is the temporal operator, which reads as "next".

5. Domain knowledge and e-government requirements

Domain requirements intend to highlight aspects of a specific application domain that on one hand are typically not known to BP and technical experts and on the other hand are quite obvious to domain experts. Missing to report domain requirements generally results in a project failure or in low quality systems. The discovery of domain requirements is particularly critical in the e-government domain where the software to implement is directed to citizens with ample differences in the capability of operating with ICT and in the distrust they have with respect to such technologies. In developing services for the PA, it becomes mandatory to consider and encode requirements aiming at removing such possible hurdles in service acceptance. Nevertheless such knowledge is not in the mind of BP or technical experts, instead people with expertise in social and anthropological fields can provide useful points of views. Such experts have to be involved in the requirements discovery phase and help in expressing the requirements they think the software system should satisfy.

In our work we considered such an aspect and, in collaborations with experts in the domain, we identified specific quality requirements for GDS delivery processes. We highlight those aspects that can be considered independent from a specific service under development. The result has been the definition of a quality framework for the delivery process of GDS that has five different dimensions, where each dimension foresees different quality levels. The framework considers coordination, control, sharing, transparency and inclusion as detailed in the following.

5.1. Coordination

With the term **coordination** we mean the capability of a PA, involved in the execution of a BP, of using different ICT in order to interact with other PAs to provide a service to the citizens.

Lack of coordination is the lowest possible level of coordination that is observable within a PA according to two different 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 the formal definition of a BP can not be observed and the approach can not be applied. It is in general the citizen that, knowing the PAs involved in the provisioning of the service, drives the process, physically moving back and forth from one administration to the other. Lack of coordination can also emerge when, given wrong BP specifications, interactions activities could results in blocking conditions.

Communication is implemented by the PA without fully integrated electronic systems. In this case the civil servants involved in the BP have different degrees of freedom in the execution of the tasks. So, for instance, the requests coming from another PA *via* an e-mailing system are processed by a civil servant according to his/her knowledge and expertise. The reception of a message does not automatically activate a task within the BP nor it is tracked by the software system.

Collaboration enables PA to take part to the GDS delivery with a fully automated BP. Requests from other participating PAs enter the organization using ICT and are handled in an automated manner without requiring the intervention of a civil servant. It is worth noting that in some cases regulations and laws could impose human intervention such as for instance when document signature is requested. Even if such activities enable a degree of freedom to the civil servants, who contribute with their knowledge, they have to be fully supported by the system.

Semantic integration implements the highest level of coordination *via* collaboration mechanisms enriched with semantic support. Explicit formal specification of the reality related to the delivery process is shared between participants to guarantee the understandability of the communications. In this case the system supports and enables the civil servants making explicit the knowledge requested to fulfill the task.

5.2. 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 GDS according to its scope of responsibility.

No sharing is observed when the administration keeps track 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 BP brings we should certainly mention data redundancy and misalignment among PA data storages.

Data sharing implements an automatic way of retrieving citizens related data interacting with the specific PA that is in charge of maintaining the needed information.

5.3. Control

With the term **control** we refer to activation policies applied to drive one step after another the BP from its start to its final fulfillment. There is the case in which a GDS announces itself to interested citizens. It is different from an approach in which the service delivery only waits for a request from an interested citizen. The two different paradigms have a major impact on service usage. Too often GDSs are not used since citizens do not know them.

Reactive control is implemented by those PA that wait for citizens requests before they start the service delivery. Citizens awareness on the "to do list" is the only driver for the service activation and delivery.

Proactive control enables the GDS to announce 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 that are inherently reactive. Nevertheless if possible proactive control can greatly foster service usage.

Creative refers to the presence of activities influencing the promotion of related and maybe relevant services. In such a case the PA implements services that inform the citizens of all the other services in which they maybe interested (or that the citizen has to activate). For instance a citizen who starts the procedure for getting married maybe 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 corresponding access points.

5.4. Transparency

With the term **transparency** we mean the ability of the PA to make citizens aware of the delivery process execution in terms of activities and people in charge to govern it, improving the citizen perceived trust.

No transparency is observed when the activities of a given service are not visible outside the administration. Therefore a PA implementing this level 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 BP, could feel frustrated given the lack of feedbacks on the execution.

Activity aware is observed when the administration implements BPs 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 BP 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 not to overwhelm the citizen with not so relevant information. The citizen will certainly feel much more comfortable with such a kind of GDS delivery and he/she will not feel so much affected by long lasting BP if he/she periodically receives still alive communications.

Role aware implements level activity aware transparency and it is enriched by the explicit identification of a civil servant responsible for the activity related to the BP. This is the civil servant who 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 citizen feels that something is not going as expected, he/she can directly get in contact with the civil servant responsible for the procedure.

5.5. Inclusion

With the term **inclusion** we mean the ability of the administration to provide services to citizens considering their different abilities. Economic, social and geographical conditions as well as physical inabilities have to be considered by BP developers. In our investigation we underline three major sources 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 the service *via* many and heterogeneous devices, such as PC, wap phone, PDA, ... (see [10,30] for a review of e-government access devices).

Profile inclusion refers to the service capabilities to support citizens physical diversities, poor ICT skills and low education levels. It is very relevant for disadvantaged people that are often excluded and marginalized by the introduction of ICT.

Language inclusion refers to the ability of the service to be used by people with different nationalities switching 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.

5.6. Business process definition in the e-government domain

The genesis of a BP to be applied within the PA sector is rather complex and its definition typically requires the involvement of many different stakeholders. Particularly interesting is the definition of those BPs aiming at describing the cooperation of offices belonging to different PAs. So, for instance, the BP permitting to move the citizen main residence requires that at least the two municipalities share a common understanding of the actions that are necessary to put in place in order to reach the final goal.

In general, limiting ourselves to the case of GDSs requiring the involvement of different offices within the same country, a BP is defined and realized according to a multi-stage process where at least the following roles/steps can be identified:

- The central administration or directly the law establishes the effects that a service should produce.
- A governing board, composed of domain experts, is appointed to define a high level process that permits to the involved PAs to provide the service producing the effects established by law.
- A PA involved in service provisioning implements the process possibly employing ICT.

The approach we propose can be fruitfully employed by governing board to identify possible flaws in the BP definition that will result in limited usage by citizens. At the same time PAs involved in the provisioning of a service can use our approach to judge the quality of the provided GDS. A PA can substitute the specification of the applied internal process within the global definition of the BP so to check wheter a good quality level can be reached.

6. Formal verification in e-Government

e-Government BP analysts should continuously review their defined artefacts to be sure that they satisfy important quality requirements. In particular, as for our proposal, processes should satisfy the requirements described in Section 5. Checking that a process satisfies a quality requirement is a complex task for a human that can also lead to uncertain results. In order to make automatic this activity processes and requirements are reconducted to formal notations where model checking techniques can be fruitfully applied.

The approach, which is sketched in Figure 1, relies on three main components providing the following functionalities: (i) BP *specification* and quality requirements selection *via* user-friendly notations; (ii) *mapping* of a process specification and of a set of quality requirements to CSP processes and to LTL assertions, respectively; (iii) formal *verification* of defined processes with respect to the specified set of requirements. In case the verification phase ends highlighting some problems, *i.e.* at least one of the selected requirements results to be violated, the Business Process should be revised and verification activities should be restarted.

The remaining part of this section is structured as follows. Section 6.1 introduces the mapping from BPMN to CSP process notation. Successively Section 6.2 describes the mapping from the quality framework to LTL.

6.1. Mapping from BPMN to CSP

BPMN version 1.1 does not define a precise semantic for the provided elements and indeed one of the objectives of version 2.0 is to clarify the semantic of BPMN elements. Nevertheless, even if a huge effort has been spent toward this objective,

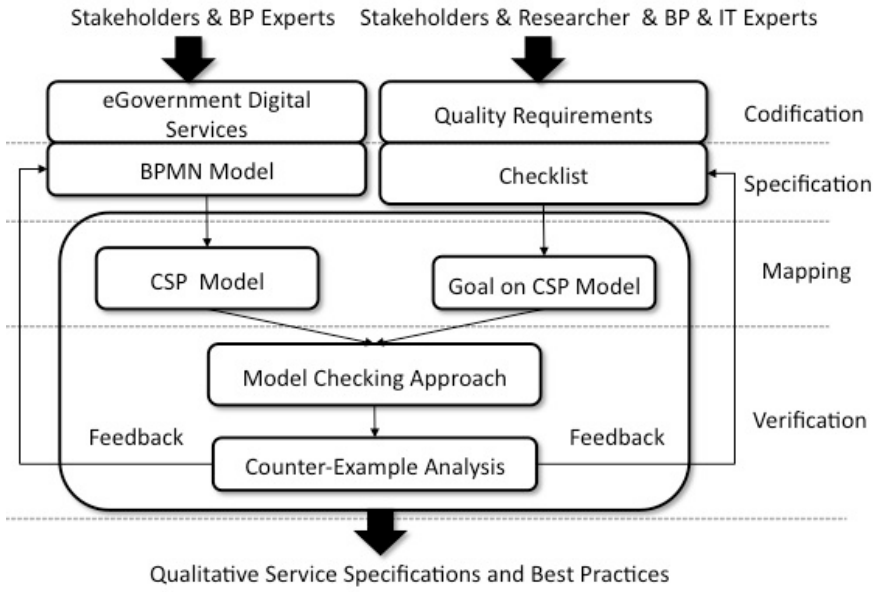


FIGURE 1. BP4PA – the proposed approach.

the current version still contains descriptions in natural language. In our work we have defined a precise semantic for BPMN elements 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 element that could possibly be different from that intuitively given by some BP developers. 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 which could lead to a low quality process implementation. Nevertheless, in order to reduce such a risk, we have derived design rules that impose some restrictions on the usage of the available BPMN elements. This is particularly important when BPMN elements miss to specify details for verification purpose or when they could lead to a particularly ambiguous interpretation. The selection of CSP as target language of the mapping is mainly related to practical opportunities. First of all the availability of a partial mapping from BPMN to CSP [42] permitted to reduce the time needed to derive a first prototype of the tool. The choice was also influenced by our acquaintance with the PAT model checker [34] which takes as input CSP process specifications.

Our mapping covers all the core BPMN elements and almost all the elements introduced by the OMG notation. Few elements, mainly dealing with transactions, such as compensation events, cancel events and time, have been kept outside the mapping. The main reason for this choice is that they are seldom used

in practice [47], at least in the e-government domain. Another reason relates to the fact that time properties are not included in the current version of the quality framework we defined.

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

- Tasks have to be typed to support specific domain-dependent characterizations.
- Each task can include at most one type of communication (send or receive). In this way we ask the BP developer to explicitly define the order as a sequence of messages exchange.
- Messages have to be typed to support specific domain-dependent characterizations.
- Pools have to be typed to address the role they play within the process.
- Pools have to be explicitly represented as loop-task or loop-subprocess. No implicit cycles are admitted in the process design. This constraint permits to have more structured BPs avoiding the presence of loop generated by kind of unconditional jump. Besides making the verification step more difficult, the presence of such a kind of loops makes the specification more complex and less understandable.
- Collapsed sub-processes are not supported. Moreover for each sub-process end and start events have to be explicitly provided. This is necessary since they support the trigger of elements inside the sub-process.

When a BP has been modeled according to the constraints listed above the approach permits to derive a CSP model from the BPMN model. The mapping has been defined according to the following general principles.

- Each BPMN graphical object included within a pool is formally represented by a CSP process or a parallel composition of generated CSP processes - we will name such a process *element CSP*.
- Each pool is mapped to a parallel composition of *element CSP* processes. In this case no message exchange will be observable – we will name such a process *private CSP*.
- The whole process results from the parallel composition of the *private CSP* processes including their interactions implemented *via* messages exchange – we will name such a processes *abstract CSP*.

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.

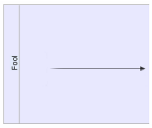
Due to lack of space we report just some rules. A wider discussion on the mapping can be found in [6].

- The rule to transform the BPMN pool elements produces a CSP global constant (Fig. 2a). 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 (Fig. 2b). Such a CSP process is able to perform an event `esc` after

$PoolID = \text{“}poolID\text{”};$
$PoolName = \text{“}poolName\text{”};$

$\#define\ poolName\ poolID;$

(a) Pool.



$PoolID = \text{“}poolID\text{”};$
$SubProcID - \text{“}subProcID\text{”};$
$FlowID = \text{“}flowID\text{”};$

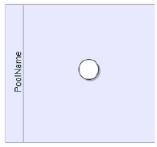
$FlowID(flowID,\ subProcID,\ poolID) =$
$(esc.flowID.poolID \rightarrow enter.flowID.poolID \rightarrow Skip;)$
$\sqcap\ (terminate.poolID \rightarrow Skip;)$
$\square\ (terminate.poolID.subProcID \rightarrow Skip;)$

(b) Flow.

FIGURE 2. Examples of mapping rules from BPMN Pool and flow to CSP.

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 a flow is outgoing. After that a 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 same control flow is incoming. When the CSP process, output of the mapping for the 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 element produces a CSP process (Fig. 3a). Such a 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 identifier of the BPMN start and of the pool that contains such a element. Each `esc` event is characterized by the identifier of the outgoing flow and by the pool



$PoolID = \text{“}poolID\text{”};$
$StartID = \text{“}startID\text{”};$
$OutgoingEdges = \text{“}flowID_1, flowID_2, ..., flowID_n\text{”};$

$startID(startID,\ flowID_1, ...,\ flowID_n,\ poolID)\ =$
$enable.startID.PoolID\ \rightarrow\ (esc.flow_1.poolID\ ||\ ...\ ||\ esc.flow_n.poolID)\ \rightarrow\ Skip;$

(a) Start Even.

$PoolID = \text{“}poolID\text{”};$
$SubProcID = \text{“}subProcID\text{”};$
$TaskID = \text{“}taskID\text{”};$
$IncomingEdgesIDs = \text{“}flowInID_1, ..., flowInID_n\text{”};$
$OutcomingEdgesIDs = \text{“}flowOutID_1, ..., flowOutID_m\text{”};$
$TaskIDTransparancy = \text{“}traspValue\text{”};$

$var\ TaskIDTransparancy\ =\ \text{“}traspValue\text{”};$

$TaskID(taskID,\ flowInID_1,\ ...\ ,\ flowInID_n,$
$flowOutID_1,\ ...\ ,\ flowOutID_m,\ subProcID,\ poolID)\ =$
$((enter.flowInID_1.poolID\ ||\ ...\ ||\ enter.flowInID_n.poolID)\ \rightarrow$
$enable.taskID.poolID\ \rightarrow\ work.taskID.poolID\ \rightarrow$
$(esc.flowOutID_1.poolID\ ||\ ...\ ||\ esc.flowOutID_m.poolID)\ \rightarrow\ Skip;\ )$
$\square\ (terminate.poolID\ \rightarrow\ Skip;\ )$
$\square\ (terminate.poolID.subProcID\ \rightarrow\ Skip;\ )$

(b) Task.

FIGURE 3. Examples of mapping rules from BPMN start event and task to CSP.

identifier that contains such a flow. The general idea is that the CSP process is immediately enabled without any interaction with the environment. Then the synchronization of the `esc` events 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 processes related to flows element and then it consumes itself.

- The rule to transform the BPMN simple task elements produces a CSP process (Fig. 3b). Such a 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 corresponding 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 corresponding 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 (Figs. 4a, 4b). For what concerns tasks sending and receiving messages we introduced a CSP dedicated channel enabling the message exchange.

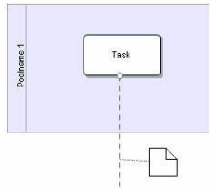
6.2. Mapping from guality requirements to LTL

As well as for the mapping of a process the quality framework has been reconducted to a formal representation. For each requirement we specified a template of an LTL property according to the format supported by the PAT model checker [34]. Then for each BP the templates are translated into LTL properties representing the selected requirements for the specific BP. LTL has been considered adequate to represent the properties we had in mind. In particular properties to be checked could ask that when a path contains an action by one administration, some other action, possibly from another administration, should follow. So in its simplest form the property would take the general structure $Fa \implies Fb$ which cannot be represented in CTL. The requirements could be equally expressed as CSP processes using then refinement checking to verify quality properties on specified processes, as suggested in [24, 43]. Nevertheless to us LTL seems easier to use even with reference to possible future exptensions of the quality framework.

6.2.1. Notation

Let GDS() be the *Abstract CSP* resulting from the mapping using the following input parameters.

- $P = \{p_1, p_2, \ldots, p_i, \ldots, p_m\}$ is the set of `pools` involved in the design of the GDS under study, where m is a natural number strictly greater then zero representing the number of pools involved in the GDS. In particular p_1 always

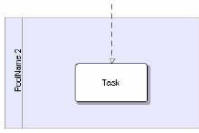


$PoolID = \text{“}poolID\text{”};$
$SubProcID = \text{“}subProcID\text{”};$
$TaskID = \text{“}taskID\text{”};$
$IncomingEdges = \text{“}flowInID_1, ..., flowInID_n\text{”};$
$OutcomingEdges = \text{“}flowOutID_1, ..., flowOutID_m\text{”};$
$OutcomingMessages = \text{“}msgOutID\text{”};$
$DataObject = \text{“}dtObjectName\text{”};$
$chType = \text{“}com\text{”} \text{ or } \text{“}coll\text{”} \text{ or } \text{“}sInt\text{”};$
$CollaborationTypeMsgID = \text{“}ColElID\text{”};$
$TransparencyMedMsgID = \text{“}TraElID\text{”};$
$SharingDatMsgID = \text{“}SharElID\text{”};$

$Channel\ chTypemsgOutID\ 0;$

$SendTaskID(taskID,\ flowInID_1,\ ...\ ,flowInID_n,\ dtObjectName$
$flowOutID_1,\ ...\ ,flowOutID_m,\ subProcID,\ poolID) =$
$((enter.flowInID_1.poolID\ \|\ ...\ \|\ enter.flowInID_n.poolID) \rightarrow$
$enable.taskID.participantID \rightarrow work.taskID.participantID \rightarrow$
$chTypemsgOutID!dtObjectName \rightarrow (\{CollaborationTypeMsgID = ColElID,$
$TransparencyMedMsgID = TraElID,\ SharingDatMsgID = SharElID\} \rightarrow$
$(esc.flowOutID_1.poolID\ \|\ ...\ \|\ esc.flowOutID_m.poolID) \rightarrow Skip;\ )$
$\square\ (terminate.poolID \rightarrow Skip;\ )$
$\square\ (terminate.poolID.subProcID \rightarrow Skip;\ )$

(a) Task sending.



$PoolID = \text{“}poolID\text{”};$
$SubProcID = \text{“}subProcID\text{”};$
$TaskID = \text{“}taskID\text{”};$
$IncomingEdges = \text{“}flowInID_1, ..., flowInID_n\text{”};$
$OutcomingEdges = \text{“}flowOutID_1, ..., flowOutID_m\text{”};$
$IncomingMessages = \text{“}msgInID\text{”};$
$CreativeTypeMsgID = \text{“}CreElID\text{”};$
$ProactiveTypeMsgID = \text{“}ProElID\text{”};$
$ReactiveTypeMsgID = \text{“}ReElID\text{”};$
$TaskIDTransparancy = \text{“}traspValue\text{”};$

$var\ TaskIDTransparancy\ =\ \text{“}traspValue\text{”};$

$ReceiveTaskID(taskID,\ flowInID_1,\ ...\ ,flowInID_n,$
$flowOutID_1,\ ...\ ,flowOutID_m,\ subProcID,\ poolID) =$
$((enter.flowInID_1.poolID\ \|\ ...\ \|\ enter.flowInID_n.poolID)$
$\rightarrow enable.taskID.participantID$
$\rightarrow channelTypemsgInID?msgIn \rightarrow \{CreativeTypeMsgID = CreElID,$
$ProactiveTypeMsgID = ProElID,\ ReactiveTypeMsgID = ReElID\}$
$\rightarrow work.taskID.participantID$
$\rightarrow (esc.flowOutID_1.poolID\ \|\ ...\ \|\ esc.flowOutID_m.poolID) \rightarrow Skip;\ )$
$\square\ (terminate.poolID \rightarrow Skip;\ )$
$\square\ (terminate.poolID.subProcID \rightarrow Skip;\ )$

(b) Task receiving.

FIGURE 4. Examples of mapping rules from BPMN sending and receiving task to CSP.

represents the pool of the citizens, while p_i, $1 < i < m$, represent the PA involved in the GDS and p_m represents the pool knowledge repository.

- $T = \{t_{1,1}, t_{1,2}, \ldots, t_{1,n_1}, t_{2,1}, t_{2,2}, \ldots, t_{2,n_2}, \ldots, t_{i,n_i}, \ldots, t_{m,1}, t_{m,2}, \ldots, t_{m,n_m}\}$ represents the set of `tasks` included in the mapped BP, where n_i, $1 \leq i \leq m$ represents the number of tasks for the corresponding pool p_i.
- $TC \subset T$ represents the set of `tasks` able to send or receive a message.
- $S = \{s_{1,1}, s_{1,2}, \ldots, s_{1,l_1}, s_{2,1}, s_{2,2}, \ldots, s_{2,l_2}, \ldots, s_{i,l_i}, \ldots, s_{m,1}, s_{m,2}, \ldots, s_{m,l_m}\}$ refers to the set of `start events` included in the mapped BP, where l_i, $1 \leq i \leq m$ represents the number of start events for the pool p_i.
- $E = \{e_{1,1}, e_{1,2}, \ldots, e_{1,k_1}, e_{2,1}, e_{2,2}, \ldots, e_{2,k_2}, \ldots, e_{i,k_i}, \ldots, e_{m,k_1}, e_{m,k_2}, \ldots, e_{m,k_m}\}$ represents the set of `end events` included in the mapped BP, where k_i, $1 \leq i \leq m$ represents the number of end event for the pool p_i.

6.2.2. Coordination

To check coordination we assume the mapping results in a deadlock free process and the task sending or receiving a message is able to manage such a message with different levels of integration.

The template (6.1) makes explicit that the service is at the level of lack of coordination. We check whether process GDS() is deadlock-free, where "deadlockfree" is a PAT reserved keyword. A service is deadlock-free if and only if the interactions among PAs will never enter a state where there is no further move.

$$\#assert\ GDS()\ deadlockfree \tag{6.1}$$

The template 6.2 makes explicit that the service implements the communication requirement. Such a requirement has to be verified for all tasks sending or receiving a message in the pool representing the PA. Informally speaking, the assertion asks whether the mapped service can always eventually enable a manual task or not. A counterexample to this assertion would be a trace which leads to a mapped service where "$enableman.t_{i,n_i}.p_i$" is not satisfied.

$$\begin{array}{l} \forall i\ and\ n_i,\ 1 < i < m: \\ \#assert\ GDS()\ \models\ \Box \Diamond\ enableman.t_{i,n_i}.p_i \end{array} \tag{6.2}$$

The templates (6.3) and (6.4) make explicit that the service implements the collaboration and semantic integration requirement respectively. Such templates implement a similar approach to the one for corresponding to the communication requirements.

$$\begin{array}{l} \forall i\ and\ n_i,\ 1 < i < m\ : \\ \#assert\ GDS()\ \models\ \Box \Diamond\ enableint.t_{i,n_i}.p_i \end{array} \tag{6.3}$$

$$\begin{array}{l} \forall i\ and\ n_i,\ 1 < i < m: \\ \#assert\ GDS()\ \models\ \Box \Diamond\ enablekm.t_{i,n_i}.p_i \end{array} \tag{6.4}$$

6.2.3. Control

The template (6.5) makes explicit that the service implements the reactive control. Such a requirement has to be verified for all starts in the pool representing the citizen and it should exist at least one task in the pools representing the PA. Informally speaking, the assertion asks whether in the mapped service the activation of tasks in the pool related to the PA are enabled by a start event in the pool citizens. The assertion says that starting from process GDS(), every reachable system state satisfies the proposition "$enable.s_{1,z}.p_1$" that implies "$enable.s_{1,z}.p_1$" until "$enable.t_{i,n_i}.p_i$" has been verified.

$$\begin{array}{l} \forall z,\ 1 \leq z \leq l_1 \rightarrow \exists i \ and\ n_i, 1 < i < m : \\ \#assert\ GDS()\ \models\ \Box\ enable.s_{1,z}.p_1\ \rightarrow\ enable.s_{1,z}.p_1\ U\ enable.t_{i,n_i}.p_i \end{array} \tag{6.5}$$

The template (6.6) makes explicit that the service implements the proactive control. Such a requirement has to be verified for all starts in the pool administration and there exists at least a task in the pool representing the citizen activated by such a start event. According to this general scenario we define the template (6.6) of LTL assertion implementing a similar approach to those one of the reactive requirement.

$$\begin{array}{l} \forall z,\ 1 \leq z \leq l_1 \rightarrow \exists i \ and\ n_i, 1 < i < m : \\ \#assert\ GDS()\ \models\ \Box\ enable.t_{i,n_i}.p_i\ \rightarrow\ enable.t_{i,n_i}.p_i\ U\ enable.s_{1,z}.p_1 \end{array} \tag{6.6}$$

In case neither the assertion (6.5) nor the assertion (6.6) are verified the service implement the creative requirement.

6.2.4. Sharing

The sharing requirement predicates over the way in which the PA handles and shares citizens data with other administrations in order to participate in the delivery of a specific GDS. To check the sharing levels, we assume that a specific participant, typed with knowledge repository role, is considered in the design of a GDS. Informally speaking, the assertion asks whether in the mapped service the activation of the start in the pool knowledge repository is followed by the activation of a task until the end is activated in the same pool. This assertion says that starting from process GDS(), every reachable system state satisfies the proposition "$enable.s_{i,l_i}.p_i$" that implies "$enter.t_{i,n_i}.p_i$" until "$enable.e_{i,k_i}.p_i$" has been verified.

$$\begin{array}{l} \exists i \ and\ n_i \ and\ l_i \ and\ k_i \ and\ i = m : \\ \#assert\ GDS()\ \models\ \Box\ enable.s_{i,l_i}.p_i\ \rightarrow\ enter.t_{i,n_i}.p_i\ U\ enable.e_{i,k_i}.p_i \end{array} \tag{6.7}$$

In the case the property is not verified the sharing is not implemented.

6.2.5. Transparency

The template (6.8) makes explicit that the service implements the transparency related task requirement. Such a requirement has to be verified for all the pools

representing the PA. Informally speaking, the assertion asks whether the mapped service can always execute a task that is followed by an observation point in the execution. The assertion says that starting from process GDS(), every system state reached satisfies the proposition "$work.t_{i,n_i}.p_i$" that in the next state "$trastask.t_{i,n_i}.p_i$" has to hold.

$$\begin{array}{l} \forall i \text{ and } n_i,\ 1 < i < m : \\ \#assert\ GDS()\ \models\ X\ trastask.t_{i,n_i}.p_i\ \rightarrow\ work.t_{i,n_i}.p_i \end{array} \tag{6.8}$$

The template (6.9) makes explicit that the service implements the role transparency. Such a template implements a similar approach to the one defined for the task transparency requirement.

$$\begin{array}{l} \forall i \text{ and } n_i,\ 1 < i < m : \\ \#assert\ GDS()\ \models\ X\ trasrole.t_{i,n_i}.p_i\ \rightarrow\ work.t_{i,n_i}.p_i \end{array} \tag{6.9}$$

6.2.6. Inclusion

The template (6.10) makes explicit that the service implements the inclusivity related channel requirement. Such a requirement has to be verified for all the pools representing the PA. The assertion asks whether the mapped service can always eventually enable an inclusive event or not. A counterexample to this assertion would be a trace which leads to a mapped service where "$inclusivech.p_i$" is not satisfied.

$$\forall i,\ 1 < y < m :\ \#assert\ GDS()\ \models\ \Box\ inclusivech.p_i \tag{6.10}$$

The templates (6.11) and (6.12) make explicit that the service implements the profile inclusivity and the language inclusivity respectively.

$$\forall i,\ 1 < y < m : \#assert\ GDS()\ \models\ \Box\ inclusiveprof.p_i \tag{6.11}$$

$$\forall i,\ 1 < y < m :\ \#assert\ GDS()\ \models\ \Box\ inclusivelang.p_i \tag{6.12}$$

7. Tool chain

The formal verification approach illustrated in this article 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 extension such as the BPMN modeler and it uses the functionalities of the PAT model checker. The CSP model is derived taking advantage of the Eclipse Modeling Framework (EMF), which is a powerful mechanism available within the eclipse platform to define metamodels. EMF together with other frameworks enabling the graphical rendering of the BPMN constructs, is at the base of BPMN modeler. Therefore through EMF,

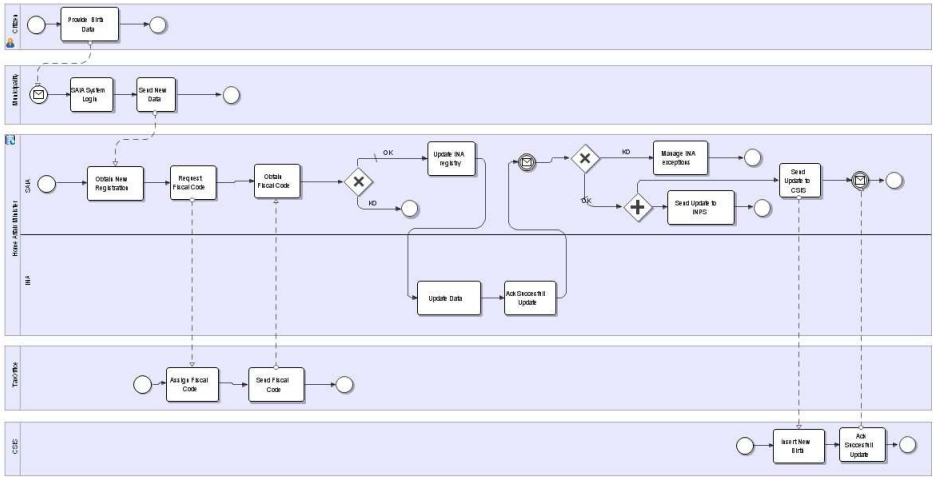


FIGURE 5. Delivery process for the new born registration service.

and its API, it is possible to interact with the defined BPMN model to retrieve the list of elements, which have been included within a BPMN specification. In this way it is possible to implement a simple parser that for each BPMN element generates the corresponding CSP code, using in our case the syntax of the PAT model checker and according to the mapping rules. Similarly the code generation includes the specification of variables enabling the checking of relevant quality requirements. After that the transformation of the BPMN specification to the corresponding CSP model has been carried on, the verification step can take place. To make also this step easier we integrated the PAT model checker within the Eclipse framework. As a result the whole tool-chain is integrated in a unique integrated development environment.

Finally it is worth mentioning that the tool permits to enlarge the set of properties that can be verified on a defined BP. In particular BP analysts can use the tool to define and check a property they considers relevant for the specific BP.

8. EXPERIMENTATION

As for any approach using model checking techniques it is important to check whether the state explosion phenomenon could hinder its applicability to real case studies. In our case we experimented with ten real processes, for the sake of clarity in Figure 5 we report the new born registration process on which the approach has been applied (for further detail see [7]). For all the considered processes relatively small state sets were generated. In particular, the experiments we have conducted using a desktop PC equipped with a Core 2 Duo 2.20 GHz and 4 GB RAM, have highlighted that defined BP can be checked with respect to the properties included

Table 1. Experimental result.

Service name	Num. of states	BPMN elem.	Coord. (s)	Control (s)	Sharing (s)	Transp. (s)	Inclusiv. (s)
Car registration	3 065 595	66	1629.18	3254.22	3 280 48	1631.28	4896.25
Passport request	735 785	68	594.28	596.08	2368.12	590.17	1777.94
Enrollment in education	44 963	45	33.88	100.85	67.75	33.50	100.59
Declaration to police	32 333	43	14.14	19.60	27.60	13.70	41.17
Social contribution	19 398	42	18.59	19.51	38.35	10.37	28.68
Social security benefits	14 295	43	10.12	9.89	9.93	4.90	14.69
Job search	4044	31	3.36	4.68	3.03	1.50	4.52
Health-related services	2451	36	1.98	1.84	1.78	0.88	2.62
Public libraries	333	18	0.28	0.12	0.07	0.06	0.17
Customs declaration	204	20	0.20	0.11	0.10	0.05	0.14

in the framework in less than 60 min, for the most complex BP scenarios. Moreover the most complex BP 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 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 BP designers. On the other side, in order to avoid the state explosion phenomenon, the mapping we provide keeps away complex data set from the resulting CSP specification. Therefore the tool does not permit to analyse a defined BP for properties related to behaviour inducted by data values. Another issue of the approach refers to the necessity of providing, just for analysis purpose, additional information for many BP elements.

In Table 1 we report a fragment of the conducted experimental results. It refers to EU based services modelled in line with Italian law and our local scenario.

Nevertheless the study has also highlighted that all the defined BPs reach low quality levels for all the quality dimensions defined in the framework. Given that citizens seldom use the studied services, the defined framework does not contradict itself. In particular, considering transparency, that maybe is the most intuitive dimension in the framework, the tool reported that in defined BPs no information were typically provided to citizens in order to inform them about the execution of the process.

9. Conclusions and future work

In this paper we have presented a quality framework and a tool for the verification and enhancement of BP related to the delivery of GDS.

The framework codifies knowledge coming from the PA domain and it has been formally encoded using temporal logic. Moreover we mapped the constructs of a widely used notation for BP definition to a language suitable for being manipulated by a model checker. The result has been BP4PA, an approach and a tool chain

which permits to check whether BP abide by the requirements enclosed in the framework.

The approach has been applied to real case studies with encouraging results. In the future we are planning to explore the possibility of enlarging the framework with additional requirements. In particular, we intend to consider requirements related to organizational parameters and time properties. Moreover we will continue our strict cooperation with local PAs in order to contribute with an incremental approach of domain analysis and the evaluation of additional real case studies.

References

[1] C. Bavec, On stimulus for citizens' use of e-government services, in *International Multiconference on Computer Science and Information Technology*. Wisia (2008) 391–395.

[2] E.M. Clarke, O. Grumberg and D.A. Peled, *Model Checking*. Hardcover (2000).

[3] G. Colclough, *The user challenge benchmarking the supply of online public services – 7th measurement*. Technical report, prepared by Capgemini for European Commission Directorate General for Information Society and Media (2007).

[4] G. Colclough and D. Tinholt, *Smarter, faster, better e-government – 8th benchmarking measurement*. Technical report, prepared by Capgemini, RAND Europe, IDC, SOGETI and DTi. For European Commission Directorate General for Information Society and Media (2009).

[5] Communication from the Commission to the Council, the European Parliament, the European Economic, Social Committee, and the Committee of the Regions, *The Role of eGovernment for Europe's Future.* Technical report, Commission of the European Communities, Brussels (2003).

[6] F. Corradini, D. Falcioni, A. Polini, A. Polzonetti and B. Re, *From bpmn to csp – toward business process verification for e-government service delivery.* Technical report, University of Camerino (2009).

[7] F. Corradini, A. Polini, A. Polzonetti and B. Re, Business processes verification for e-government service delivery. *IS Management* **27** (2010) 293–308.

[8] J. Davies, T. Janowski, A.K. Ojo and A. Shukla, Technological foundations of electronic governance, in *Proc. of ICEGOV*, edited by T. Janowski and T.A. Pardo. *ACM Int. Conf. Proc. Ser.* **232** (2007) 5–11.

[9] R.M. Dijkman, M. Dumas and C. Ouyang, *Formal semantics and analysis of BPMN process models using petri nets.* Online (2007).

[10] W. Ebbers and W. Pieterson, The multi-channel citizen: a study of the use of service channels by citizens in the Netherlands, in *Electronic Government, 6th International Conference, EGOV Proceedings* (2007).

[11] E.A Emerson, Temporal and modal logic, in *Handbook of theoretical computer science formal models and semantics* B. Cambridge, MA, USA (1990).

[12] A.D. Farrell, M.J. Sergot and C. Bartolini, Formalising workflow: a CCS-inspired characterisation of the YAWL workflow patterns. *Group Decision and Negotation* **16** (2007) 213–254.

[13] A. Forster, G. Engels, T. Schattkowsky and R. Van Der Straeten, Verification of business process quality constraints based on visual process patterns, in *TASE '07: Proc. of the First Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering*. IEEE Computer Society, Washington, DC, USA (2007) 197–208.

[14] X. Fu, T. Bultan and J. Su, Analysis of interacting BPEL web services, in *WWW '04: Proc. of the 13th international conference on World Wide Web*. ACM, New York, NY, USA (2004) 621–630.

[15] R. Hamadi and B. Benatallah, A petri net-based model for web service composition, in *ADC '03: Proc. of the 14th Australasian database conference*. Australian Computer Society, Inc., Darlinghurst, Australia, Australia (2003) 191–200.
[16] P. Harmon, *Business Process Change – A guide tor Business Manager and BPM and Six Sigma Professionals*. Horgan Kaufmann (2004).
[17] A.R. Hevner, S.T. March, J. Park and S. Ram, Design science in information systems research. *Manage. Inf. Syst. Q.* **28** (2004) 75–105.
[18] C.A.R. Hoare, *Communicating Sequential Processes*. Prentice Hall (2004).
[19] M. Janssen and G. Kuk, E-government business models for public service networks. *Int. J. Electron. Govern. Res.* **3** (2007) 54–71.
[20] W. Janssen, R. Mateescu, S. Mauw, P. Fennema and P. van der Stappen, Model checking for managers, in *Proc. of the 5th and 6th International SPIN Workshops on Theoretical and Practical Aspects of SPIN Model Checking*. Springer-Verlag, London, UK (1999) 92–107.
[21] M. Janssen, G. Kuk and R.W. Wagenaar, A survey of web-based business models for e-government in the netherlands. *Gov. Inf. Q.* **25** (2008) 202–220.
[22] K. Layne and J. Leeb, Developing fully functional e-government: a four stage model. *Gov. Inf. Q.* **18** (2001) 122–136.
[23] A. Lindsay, D. Downs and K. Lunn, Business processes-attempts to find a definition. Special Issue on Modelling Organisational Processes. *Inf. Softw. Technol.* **45** (2003) 1015–1019.
[24] G. Lowe, Specification of communicating processes: temporal logic *versus* refusals-based refinement. *Form. Asp. Comput.* **20** (2008) 277–294.
[25] J. Mendling, *Metrics for Process Models*. Springer (2008).
[26] S. Morimoto, A survey of formal verification for business process modeling, in *ICCS '08: Proc. of the 8th international conference on Computational Science, Part II*. Springer-Verlag Berlin, Heidelberg (2008) 514–522.
[27] S. Narayanan and S.A. McIlraith, Simulation, verification and automated composition of web services, in *WWW '02: Proc. of the 11th international conference on World Wide Web*. ACM, New York, NY, USA (2002) 77–88.
[28] P. Norris, *Digital Divide Civic Engagement, Information Poverty and the Internet Worldwide*. Cambridge University Press (2001).
[29] OMG, *Business process model and notation (bpmn) 2.0*. Technical report, Object Management Group (2009).
[30] W. Pieterson and J. van Dijk, Channel choice determinants; an exploration of the factors that determine the choice of a service channel in citizen initiated contacts, in *dg.o '07: Proc. of the 8th annual international conference on Digital government research*. Digital Government Research Center (2007) 173–182.
[31] G. Salaün, L. Bordeaux and M. Schaerf, Describing and reasoning on web services using process algebra, in *ICWS '04: Proc. of the IEEE International Conference on Web Services*. IEEE Computer Society Washington, DC, USA (2004) 43.
[32] M. Solar, G. Valdés, H. Astudillo and M. Iribarren, G. Concha and M. Visconti, *Conception, development and implementation of an e-government conception, development and implementation of an e-government maturity model in public agencies*. Elsevier in Government Information Quarterly (2011).
[33] C. Stefansen, SMAWL: a SMAll Workflow Language based on CCS, in *Proc. of CAiSE Short Paper Proceedings* (2005).
[34] J. Sun, Y. Liu and J. Song Dong, Model checking CSP revisited: introducing a Process Analysis Toolkit, in *Proc. of ISoLA*, edited by T. Margaria and B. Steffen. *Commun. Comput. Inform. Sci.* **17** (2008) 307–322.
[35] W.M.P. van der Aalst and A. H.M. ter Hofstede, YAWL: yet another workflow language. *Inf. Syst.* **30** (2005) 245–275.
[36] P. Wauters and G. Colclough, *On-line availability of public services: How is europe progressing? Web based survey on electronic public services report of the 6th measurement*. Technical report (2006). Prepared by Capgemini for European Commission Directorate General for Information Society and Media (2006).

[37] S.A. White and D. Miers, *BPMN Modeling and Reference Guide Understanding and Using BPMN*. Future Strategies Inc. (2008).

[38] P. Wohed, W.M.P. van der Aalst, M. Dumas and A.H.M. ter Hofstede, Analysis of web services composition languages: the case of BPEL4WS, in *ER*, edited by I.-Y. Song, S.W. Liddle, T.W. Ling and P. Scheuermann. *Lect. Notes Comput. Sci.* **2813** (2003) 200–215.

[39] P.Y.H. Wong and J. Gibbons, A process-algebraic approach to workflow specification and refinement, in *Proc. of 6th International Symposium on Software Composition. Lect. Notes Comput. Sci.* **4829** (2007).

[40] P.Y.H. Wong and J. Gibbons, Verifying business process compatibility (short paper), in *QSIC '08: Proc. of the 2008 The Eighth International Conference on Quality Software*. IEEE Computer Society, Washington, DC, USA (2008) 126–131.

[41] P.Y.H. Wong and J. Gibbons, A Process Semantics for BPMN, in *Proc. of 10th International Conference on Formal Engineering Methods. Lect. Notes Comput. Sci.* **5256** (2008). Extended version available at http://web.comlab.ox.ac.uk/oucl/work/peter.wong/pub/bpmnsem.pdf.

[42] P.Y.H. Wong and J. Gibbons, Formalisations and applications of bpmn. Special issue on FOCLASA (2008). *Sci. Comput. Program.* (2009).

[43] P. Wong and J. Gibbons, Property specifications for workflow modelling, in Integrated Formal Methods, edited by M. Leuschel and H. Wehrheim. *Lect. Notes Comput. Sci.* **5423** (2009) 56–71.

[44] M.T. Wynn, H.M.W. Verbeek, W.M.P. van der Aalst, A.H.M. ter Hofstede and D. Edmond, *Bus. Process Manag. J.* **15** (2009) 74–92.

[45] J. Ye, S. Sun, W. Song and L. Wen, Formal semantics of BPMN process models using YAWL. *Intelligent Information Technology Applications, 2007 Workshop on* **2** (2008) 70–74.

[46] L. Zhao, Q. Li, X. Liu and N. Du, *A modeling method based on CCS for workflow* (2009) 376–384.

[47] M. zur Muehlen and J. Recker, How much language is enough? Theoretical and practical use of the business process modeling notation, in *Proc. of CAiSE*, edited by Z. Bellahsene and M. Léonard. *Lect. Notes Comput. Sci.* **5074** (2008) 465–479.

Communicated by E. Moggi.
Received December 21, 2010. Accepted January 17, 2012.