COMPOSITIONAL VERIFICATION OF BUSINESS PROCESSES MODELLED WITH BPMN

Luis E. Mendoza Morales[1], Manuel I. Capel Tuñón[2] and María A. Pérez[1]
[1]*Processes and Systems Department, Simón Bolívar University, PO Box 89000, Caracas 1080-A, Venezuela*
[2]*Software Engineering Department, University of Granada, Aynadamar Campus, 18071 Granada, Spain*

Keywords: Business process modelling, Model–checking, Task model, Compositional verification, Formal specification.

Abstract: A specific check that is required to be performed as part of the *Business Process Modelling* (BPM) is on whether the activities and tasks described by *Business Processes* (BPs) are sound and well–coordinated. In this work we present how the *Model–Checking* verification technique for software can be integrated within a *Formal Compositional Verification Approach* (FVCA) to allow the automatic verification of BPs modelled with *Business Process Modelling Notation* (BPMN). The FVCA is based on a formal specification language with composition constructs. A timed semantics of BPMN defined in terms of the Communicating Sequential Processes + Time (CSP+T) extends untimed BPMN modelling entities with timing constrains in order to detail the behavior of BPs during the execution of real scenarios that they represent. With our proposal we are able to specify and to develop the *Business Process Task Model* (BPTM) of a target business system. In order to show a practical use of our proposal, a BPTM of an instance of a BPM enterprise–project related to the *Customer Relationship Management* (CRM) business is presented.

1 INTRODUCTION

The *Business Process Modelling Notation* (BPMN) (OMG, 2009) has become the "de facto" standard graphical notation for *Business Process Modelling* (BPM). BPMN describes processes in terms of order dependencies between subprocesses and atomic tasks. In a short time, BPMN has been supported by a variety of BPM tools (OMG, 2009), and several companies start using it as their standard modeling technique. However, existing verification tools can not directly be applied to BPMN models. BPMN is a graphical notation that differ from the formal languages required by most existing verification tools. Moreover, most automatic verification techniques and tools operate on models described by using formal modeling languages (as Petri nets or Process Algebras), not often used in industry. Then, to automatically carry out the verification of a BPMN model the use of formal languages is required as well as to transform/interpret original BP models into "as–equivalent–as–possible" executable formal models (knows as *Business Process Task Model* —BPTM). The idea of obtaining directly an executable model (i.e., a BPTM) from a BP conceptual[1] one (e.g., a BPMN model) led us to propose a software verification framework, called *Formal Compositional Verification Approach* (FCVA), applicable to the BPM domain. With FCVA, the *correctness* of any BPTM can be *model–checked* to determine the satisfaction of temporal BP properties, i.e., if the tasks behaviour conforms to the communication protocols, temporal consistency between collaborative tasks, etc., and temporal BP rules, such as task timeliness and performance. We propose the construction of a BPTM (i.e., a executable model of the BP) as a set of process terms following the construction rules of the Communicating Sequential Processes + Time (CSP+T) process calculus. Thus, the behavioural aspects and temporal constraints of a BPMN model are specified and verified in the corresponding BPTM by using the CSP+T formal specification language, as we will show in the sequel by the discussion of an instance of *Customer Relationship Management* (CRM) business. However, due our approach is mainly supported on CSP–based calculus (i.e., the model checkers are based on *refinement con-*

[1]A BP descriptive model based on qualitative assumptions about its elements, their interrelations, and BP boundaries.

cept (Roscoe, 1997)), its major limitation is that it is restricted to check *security properties* (Roscoe, 1997).

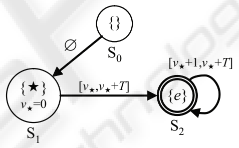
In the literature we can find different works that address the verification and validation of BP modelled with BPMN. There are formal methods for verifying BPMN models based on the π calculus (Ma et al., 2008) or Petri Nets (Aalst, 2002), tools which can debug grammatical errors in BPMN models and transforms diagrams into BPEL (OASIS, 2007), and techniques providing consistency of BPs written in *Business Process Execution Language for Web Services* (BPEL4WS) (OASIS, 2007) with *Model–Checking* (MC) (Díaz et al., 2005), among others. In (Morimoto, 2005) is presented a extended survey of recently proposed verification techniques for verifying BPMN models and compare them between each other and with respect to motivations, methods, and logics. Nevertheless, none of the cited works merge modelling of BPs with the specification, design and verification of BPTMs, and thus takes full advantage of the strengths of the process calculus. Differently from other research, our work is aimed at giving a systemic, integrated vision of specification, design and verification tasks of BPs, by incorporating the use of MC tools in the BPTM development cycle. In order to attain this, we establish how to combine different formalisms within the same semantic domain (i.e., *Kripke Structures* —KS), so we can use this kind of tool to allow us obtaining *the verification of the complete BPTM associated to a specific BP model*.

The remainder of this paper is structured as it follows. In the next section we give a short theoretical background (*Clocked Computation Tree Logic* —CCTL— and CSP+T) that supports our approach. Then, we give a brief description of BPMN, as an introduction to the time semantics which is subsequently proposed for some BPMN notational elements. Next, we describe the compositional verification proposal in detail. Finally, we apply our proposal to a BPM related to the CRM business. The last section gives the conclusions and future work.

2 FORMAL BACKGROUND

2.1 CSP+T

CSP+T (Žic, 1994) is a real–time specification language which extends Communicating Sequential Processes (CSP) (Roscoe, 1997) to allow the description of complex event timings, from within a single sequential process, of use in the behavioural specification of concurrent systems. CSP+T is a superset of CSP, as a major change to the latter, the traces of events are now *pairs* denoted as *t.e*, where *t* is the global *absolute* time at which event *e* is observed. The operators, related with timing and enabling–intervals included in CSP+T are: (a) the special process instantiation event denoted $\star$ (star); (b) the time capture operator ($\bowtie$) associated to the time stamp function $a_e = s(e)$ that allows storing in a variable *a* (marker variable) the occurrence time of an event *e* (marker event) when it occurs; and (c) the event–enabling interval $\mathrm{I}(T,t_1).a$, representing timed refinements of the *untimed* system behaviour and facilitates the specification and proof of temporal system properties (Žic, 1994). CSP–based MC tools take a process (representing the system implementation), and automatically check whether the process fulfils the system specification. Büchi automata (Alur and Dill, 1994) have emerged as formal models derived from *Kripke structures* (KS) (Clarke et al., 2000) to allow the analysis and verification of system behaviour. A variant of these are *timed Büchi automata* (TBA), see Figure 1, which are able to describe the time at which events happen on any system run and the temporal properties holding in the next possible set of system states.



$$P = 0.\star \rightarrow \mathrm{I}(T, v_\star).e \rightarrow P'$$

$$\text{where}: \begin{cases} v_\star = s(\star) = 0 \\ \{e\} \equiv occur(e) \end{cases}$$

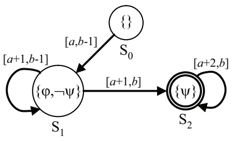
Figure 1: Kripke structure of a CSP+T process term.

2.2 CCTL

Clocked Computation Tree Logic (CCTL) (Rüf and Kropf, 1997) is a temporal logic extending CTL (Clarke et al., 2000) with quantitative bounded temporal operators. See (Rüf and Kropf, 1997) for more details. CCTL includes the CTL with the operators *until* (U) and the operator *next* (X) and other derived operators in LTL, such as R, B, C and S, useful to facilitate RTS properties specification. In CTL all “LTL-like” temporal operators are preceded by a run quantifier (A universal, E existential) which determines whether the temporal operator must be interpreted over one run (existential quantification) or over every run (universal quantification) starting in the actual configuration. CCTL is an *interval logics* that allow us to carry out a logical reasoning at the level of time intervals, instead of instants. Within our approach, the basic model for understanding concurrent systems is the *interval structure*[2]. Temporal logic MC

[2]A state transition system with labelled transitions, assuming that every interval structure has exactly one clock for the measure of time (Rüf and Kropf, 1997).

takes a structure (representing the system property) which is unwound into a model and a formula, and automatically checks if the structure (model) meets the specification (formula). The fundamental structures are *timed* KS (unit–delay, temporal) (Clarke et al., 2000); i.e., the model checker determines whether the KS is a model of the formula. Figure 2 shows a graphical example (a Büchi automaton (Alur and Dill, 1994)) of the KS of a CCTL formula.



$$\phi = \varphi \mathsf{U}_{[a,b]} \psi$$

Figure 2: Kripke structure of a CCTL formula.

3 BPMN AND VERIFICATION

BPMN has emerged as an important open standard graphic notation for modelling and drawing BPs. The main goal of BPMN is to provide a notation that is readily understandable by all business users. BPMN specifies a single diagram, called *Business Process Diagram* (BPD). To depict a BP flow, you simply model the *Events* that occur to start the BP, the *Activities* and *Tasks* carried out, and the outcome of the BP flow. Business decisions and flow branching are modelled using *Gateways*. A Gateway is similar to a decision symbol in a flowchart. Furthermore, an *Activity* in the flow can be a *sub–processes*, which can be graphically shown by another BPD connected via a hyperlink to a process symbol. If an *Activity* is not decomposed into sub–processes, it is considered a *Task*. The Tasks are the lowest–level parts of a BP, i.e., the atomic parts of BPs. A *Pool* typically represents an *organization* or *business entity* and a *Lane* typically represents a *department* or a *business worker* within that organization or other things like functions, applications, and systems. When the BPM is done the Pools can be further partitioned into Lanes. Both Pools and Lanes represent *Business Process Participants* (BPPs) (OMG, 2009), i.e., these business entities, included in the BP, which follow process flows that perform Activities and Tasks. A BPD represents a *scenario* of a business model. A *scenario* describes how the workflow of a particular BP is realized, in terms of collaborating business entities or objects (Kruchten, 2003), within the business model.

According to (Wong and Gibbons, 2008), the BPMN specification does not yet have a formal behavioural semantics, commonly accepted, to attain this is very important for carrying out the behavioural specification and verification activities of critical BPs. This is particularly important when specifying BP collaboration, where task coordination depends on the execution order and on the duration of the other one. BPs analysts and designers need tools and methodological approaches that support critical BPs verification, as part of BPM. BPs verification, mainly in the early development cycle, can provoke to take corrective actions in time and at low cost for business. Moreover, validation of BPM results is extremely expensive and risky for the development process when postponed until system deployment. In this sense, our proposal will help analysts and designers working on BPM to conduct temporal verification of critical BP models before starting the software's life cycle implementation phase.

3.1 Improving the BPMN Semantics

Our proposal takes as its starting point the semantics for the BPMN analysis entities given in (Wong and Gibbons, 2008), combined with CSP+T operators; specifically, the *time capture operator* ($\bowtie$) and the *event–enabling interval* $\mathrm{I}(T,t).a$ (or $[t,t+T].a$), to specify the response times of some notational elements of BPMN and to control their time span, according to the maximum times at which every task must execute to meet the temporal constraints specified in the BP. In this way, a more precise and complete semantics is obtained for the local diagrams that represent individual participants, as well as for the global diagram that represents business collaboration, required by the BP and depicted in the BPD.

To briefly describe our proposal, the BPMN notational elements specification is shown in Figure 3. We define a direct map from the activities size (i.e., rounded rectangles) to the maximum (*ran.max*) and minimum duration (*ran.min*), which are established as part of the activities attributes. Furthermore, we denote as t_x the times at which the invocation events ε_x occur on the BPMN modelling entities, and with $Sx.ran.min$ and $Sx.ran.max$ the minimum and maximum duration ranges of *Sx* activities, respectively, according to what is established in BPMN.

In Figure 3 (a) the *start event* of BPMN is depicted, which represents the BP instantiation for its execution. In CSP+T, its specification is performed by means the $\star$ instantiation event and marking the occurrence instant of that event in the $v_\star$ marker variable; i.e.,

$$P(start) = (\star \bowtie v_\star \rightarrow SKIP; P(start)) \square (\varepsilon_{end} \rightarrow SKIP)$$

Let be the activity $S1$, which precedes the activity $S2$, according to the flow shown in Figure 3 (b). According to the BPMN semantics which we propose, the start of the activity $S2$ execution (i.e., the

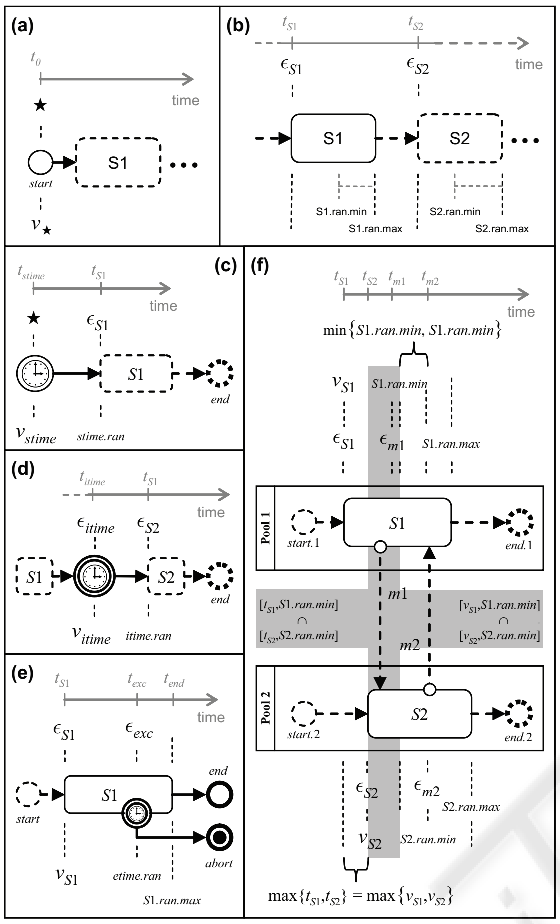


Figure 3: Timing graphical analysis of some BPMN notational elements.

occurrence of event ε_{S2}) depends on the ending instant of activity $S1$, which must occur within the time span of activity $S2$, given by the range $S1.ran.min$ to $S1.ran.max$. In its turn, the measurement of ranges $S1.ran.min$ and $S1.ran.max$ depends on the occurrence of event ε_{S1}. Then, we must make sure that the event ε_{S2} will timely occur; i.e., within the interval $[S1.ran.min, S1.ran.max]$ from the occurrence instant t_{S1}, stored in v_{S1}, at which the $S1$ (ε_{S1}) was invoked. In CSP+T the process term that specifies the expected behaviour is:

$$\begin{aligned}P(S1) = (&\varepsilon_{S1} \bowtie v_{S1} \rightarrow SKIP;\\ &\mathrm{I}(S1.ran.max - S1.ran.min, v_{S1} + S1.ran.min).\varepsilon_{S2}\\ &\rightarrow SKIP; P(S1))\\ &\square(\varepsilon_{end} \rightarrow SKIP)\end{aligned}$$

The BPMN *Timer Start* and *Timed Intermediate* events specify the delay in the BPMN modelling entity invocation which precedes the *Sequence Flow*. Then, according to the schema shown in Figure 3 (c) and 3 (d), the process terms in CSP+T that specifies those behaviours are:

$$\begin{aligned}P(stime) = (&\star \bowtie v_{stime} \rightarrow SKIP; \mathrm{I}(stime.ran, v_{stime}) \rightarrow SKIP;\\ &\varepsilon_{S1} \rightarrow SKIP; P(stime))\\ &\square(\varepsilon_{end} \rightarrow SKIP)\end{aligned}$$

$$\begin{aligned}P(itime) = (&\varepsilon_{itime} \bowtie v_{itime} \rightarrow SKIP; \mathrm{I}(T_{itime}, v_{itime}) \rightarrow SKIP;\\ &\varepsilon_{S2} \rightarrow SKIP; P(itime))\\ &\square(\varepsilon_{end} \rightarrow SKIP)\end{aligned}$$

According to Figure 3 (e), the process term in CSP+T specifying a task behaviour with an *Exception Flow*, will present the following syntax:

$$\begin{aligned}P(S1) = (&\varepsilon_{S1} \bowtie v_{S1} \rightarrow SKIP;\\ &\mathrm{I}(S1.ran.max - S1.ran.min, v_{S1} + S1.ran.min).\varepsilon_{end}\\ &\rightarrow (SKIP \triangle \mathrm{I}(S1.ran.max, v_{S1}).\varepsilon_{exc} \rightarrow SKIP;\\ &abort.1 \rightarrow STOP); P(S1))\\ &\square(\varepsilon_{end} \rightarrow SKIP)\end{aligned}$$

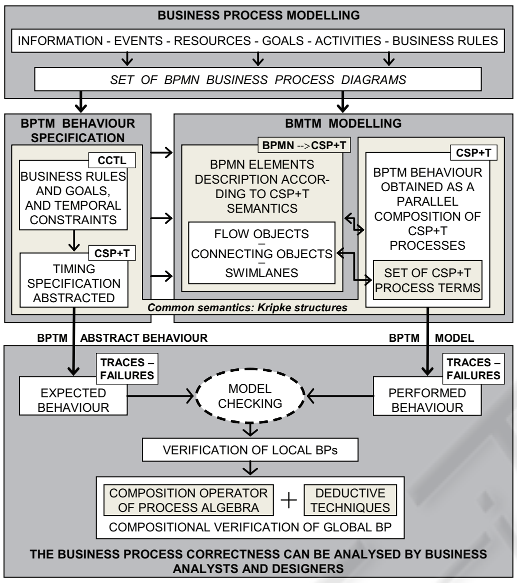
Finally, for the case of *Message Flows*, depicted in Figure 3 (f), the process terms that include the collaboration between two participants *Pool*1 and *Pool*2, are structured according to the following text:

$$\begin{aligned}P(S1) = (&\varepsilon_{S1} \bowtie v_{S1} \rightarrow SKIP;\\ &\mathrm{I}(\min\{S1.ran.min, S2.ran.min\}, \max\{v_{S1}, v_{S2}\}).\varepsilon_{m1}!x \rightarrow SKIP;\\ &\mathrm{I}(\min\{S1.ran.min, S2.ran.min\}, \max\{v_{S1}, v_{S2}\}).\varepsilon_{m2}?y \rightarrow SKIP;\\ &\mathrm{I}(S1.ran.max - S1.ran.min, v_{S1} + S1.ran.min).\varepsilon_{end.1} \rightarrow\\ &\rightarrow SKIP; P(S1))\\ &\square(\varepsilon_{end.1} \rightarrow SKIP)\end{aligned}$$

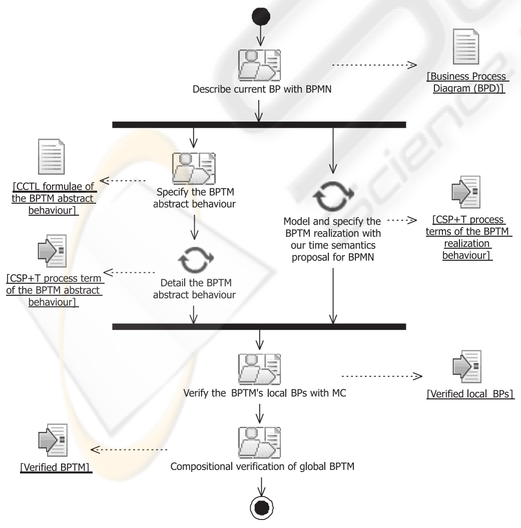
$$\begin{aligned}P(S2) = (&\varepsilon_{S2} \bowtie v_{S2} \rightarrow SKIP;\\ &\mathrm{I}(\min\{S1.ran.min, S2.ran.min\}, \max\{v_{S1}, v_{S2}\}).\varepsilon_{m1}?x \rightarrow SKIP;\\ &\mathrm{I}(\min\{S1.ran.min, S2.ran.min\}, \max\{v_{S1}, v_{S2}\}).\varepsilon_{m2}!y \rightarrow SKIP;\\ &\mathrm{I}(S2.ran.max - S2.ran.min, v_{S2} + S2.ran.min).\varepsilon_{end.2} \rightarrow\\ &\rightarrow SKIP; P(S2))\\ &\square(\varepsilon_{end.2} \rightarrow SKIP)\end{aligned}$$

4 BPTM VERIFICATION APPROACH

The BP model can have several views and each view is expressed through one or more diagrams (Eriksson and Penker, 1998), which can be of several types, depending on the situation or specific structure of the business that needs to be portrayed. The diagrams capture BP rules, goals, relations between objects and their interactions. These views are not separated models, but different perspectives of one or more aspects of the business being modelled. When together these views create a complete business model (Eriksson and Penker, 1998). In this work we focus only on the BP view of a business model. According to BPMN (OMG, 2009) and our objectives, we started from the BPD because is the mechanism used by BPMN for creating BP models, while at the same time BPD is able to handle the complexity inherent to BPs (OMG, 2009). As we introduce previously, a BPTM structure is a set of groups of tasks, representing a large number of possible real–world scenarios expressed in compact form. Thus, we are focused here on the BPTM, which allow us to obtain a description of most of the tasks that a BP accomplish (Paternò, 2001). On Figure 4 we see the graphical summary of our proposal that shows, (a) the integration of MC concepts with our timed semantics proposal for BPMN and (b) the workflow with the different paths to be followed in its application and artifacts (denoted inside brackets) that are obtained from the activities execution, to carry out the verification of a BPTM.


BUSINESS PROCESS MODELLING
INFORMATION - EVENTS - RESOURCES - GOALS - ACTIVITIES - BUSINESS RULES
SET OF BPMN BUSINESS PROCESS DIAGRAMS
BPTM BEHAVIOUR SPECIFICATION
CCTL
BUSINESS RULES AND GOALS, AND TEMPORAL CONSTRAINTS
CSP+T
TIMING SPECIFICATION ABSTRACTED
BMTM MODELLING
BPMN -->CSP+T
BPMN ELEMENTS DESCRIPTION ACCORDING TO CSP+T SEMANTICS
FLOW OBJECTS
CONNECTING OBJECTS
SWIMLANES
CSP+T
BPTM BEHAVIOUR OBTAINED AS A PARALLEL COMPOSITION OF CSP+T PROCESSES
SET OF CSP+T PROCESS TERMS
Common semantics: Kripke structures
BPTM ABSTRACT BEHAVIOUR
BPTM MODEL
TRACES – FAILURES
EXPECTED BEHAVIOUR
MODEL CHECKING
TRACES – FAILURES
PERFORMED BEHAVIOUR
VERIFICATION OF LOCAL BPs
COMPOSITION OPERATOR OF PROCESS ALGEBRA
DEDUCTIVE TECHNIQUES
COMPOSITIONAL VERIFICATION OF GLOBAL BP
THE BUSINESS PROCESS CORRECTNESS CAN BE ANALYSED BY BUSINESS ANALYSTS AND DESIGNERS

(a) Integrated view.


Describe current BP with BPMN
[Business Process Diagram (BPD)]
[CCTL formulae of the BPTM abstract behaviour]
Specify the BPTM abstract behaviour
Model and specify the BPTM realization with our time semantics proposal for BPMN
[CSP+T process terms of the BPTM realization behaviour]
[CSP+T process term of the BPTM abstract behaviour]
Detail the BPTM abstract behaviour
Verify the BPTM's local BPs with MC
[Verified local BPs]
[Verified BPTM]
Compositional verification of global BPTM

(b) Activity diagram.

Figure 4: Our verification proposal.

A complete behavioural description of the BPTM cannot be obtained by only using the syntactic information provided by BPD without considering dynamic behaviour and temporal constrains represented by the BPMN notational elements (i.e., activities — tasks and sub–processes—, and timer start and timer intermediate events) and the timed constrains relate to the participants collaboration (i.e., the message interchange represented by message flows). As result we obtain a set of detailed CSP+T process terms (i.e., the BPTM), which describes completely the temporal behaviour of the BP described by the BPD. We can check the correctness of the BPTM by using a MC tool w.r.t. previously specified properties for the BPTM derived from the business rules and goals.

The complete description of the BPTM temporal behaviour is obtained by applying our timed semantics proposal to some BPMN notational elements. Thus, some non–functional requirements (i.e., deadlock–freeness, reliability) and temporal constrains (i.e., timeliness, deadlines) that the BPTM must fulfill are *specified* in CCTL (see an example of such a formula in Figure 2), which is based on the interval structure and time–annotated automata (Rüf and Kropf, 1997). Afterwards, these properties are expressed by a set of CSP+T process terms that represents the abstract expected behaviour of the BPTM. As result, we obtain a set of detailed CSP+T process terms that specify and deal with behavioural aspects and temporal constrains of the BPMN notational elements involved into the BPTM realization. In this sense, the verification carried out here exclusively refers to the BPTM behaviour modelled by the CSP+T process conformed by the set of CSP+T process terms that describe the behaviour of the BPMN elements, i.e., the composition of the CSP+T process terms that represents the activities performed by the participants collaboration.

Once obtained the BPTM model (i.e., the set of CSP+T process term that represents the realization of the BP), we can proceed to BPTM verification according to the rules of CSP–based process calculus. By using CSP–based MC tools we *model check* the local BPs corresponding to the Pools within the BPD against the set of process process terms that represents the properties (i.e., the expected behaviour) that the BPTM must be accomplish. Finally, by the *BPTM compositional verification* Theorem, we obtain the complete verification of the BPTM behaviour that corresponds to the global BP *BPD*, according to the relation (1).

BPTM Compositional Verification. *Let the global BP BPD be structured into several business participants $Pool_i$ working in parallel, $BPD = \|_{i:1..n} Pool_i$.*

For a set of process terms $T(Pool_i)$ describing the behaviour of business participants $Pool_i$, properties ϕ_i, invariants ψ_i, and deadlock δ, with $\bigcap_{i:1..n} \Sigma_i = \varnothing$, $\bigcap_{i:1..n} \Omega_i = \varnothing$, and $\bigcap_{i:1..n} \mathcal{L}(T(Pool_i)) = \varnothing$, the following condition holds[3]*:*

$$T(BPD) \vDash (\phi \wedge \psi \wedge \neg\delta) \Leftrightarrow \mathop{\Big|\Big|}_{i:1..n} T(Pool_i) \vDash \bigwedge_{i:1..n} (\phi_i \wedge \psi_i) \wedge \neg\delta, \qquad (1)$$

where $T(BPD) = \|_{i:1..n} T(Pool_i)$.

Since our approach is aimed at representing BPTM concurrent aspects, the contribution is more focused on compositional verification of consistency and synchronization of concurrent local BPs which conform the BPTM than in other BPs oriented validations; i.e., according to our approach, the verification of structured BPTM can be carried out with correctness by only starting from the verification of the simplest BPMN local process.

As final remark, the main objective of this work is aimed at verification of BPTM, which are derived from a series of BPs modelled with BPMN. However, our proposal can be adapted to other BPM languages and standards which allow the transformation of the properties to verify and the modelling elements of BPTM into formal language constructs supported by MC tools; i.e., KS. See (Capel et al., 2008) to review an example of an adaptation of our BPTM verification approach to a BPTM derived from BPs modelled with BPM UML stereotypes.

5 AN APPLICATION OF FCVA

To show the applicability of our proposal, it was applied to a BPM enterprise–project related to the CRM business (Mendoza et al., 2007). To perform the verification of the BPTM associated with CRM BP using our approach, the business requirement analysis and context should be obtained beforehand, by means of a BPM. In summary, the BPM obtained the *Informing Customer*, *Customizing Service*, *Studying Behaviour Pattern*, *Product/Service Produce*, *Product/Service Sell* and *Assisting Customers* BPs that represent a minimum functionality of the CRM strategy and are key factors to understanding the CRM business. We will only use the BPMN BPDs obtained from the CRM BPM considered of interest to show our verification approach.

We will only show an example of application of the timed semantics proposed for BPMN and we only focus on the verification of one CRM BP. We selected to work with the *Product/Service Sell* BP, due to its importance to the CRM strategy. The required information to allow formal reasoning about CRM participant collaboration is displayed by the *Product/Service Sell* BPD shown in Figure 5, which allows a *Company* performing the activities associated with selling a Product/Service requested by a *Customer*). We can see that the *Customer* is represented by a *Pool* and the *Company* by other one, which exchange *Message Flows* to achieve the collaboration required by the BP. In turn, the Company is partitioned in *Lanes* (i.e., *Sales*, a *Logistic_agent*, and *Attention_channel*), representing the Company's internal participants involved in the realization of the BP. The Product/Service Sell BP starts when a Customer requests a communication with the Company. In this sense, the BP meets Customer requirements to buy certain *Product/Service*. However, the Product/Service Sell BP can be initiated by the Company to respond to CRM strategies to sell any Product/Service to the Customers. As shown in Figure 5, the BP provides a high collaboration from the participants to achieve their execution, which deserves a synchronization of the activities involved in message flows.

5.1 BPTM Definition and Description

Known as Product/Service Sell BP modelled with BPMN, now the next step is to obtain its specification in CSP+T, according to the proposal briefly described in section 3.1, which amounts to the definition and semantic description of the BPMN modelling entities that represent duration times[4]. We define the sets *CU*, *CO*, and *CO*2 for indexing the processes mapped to the modelling entities of Customer (i.e., *Cus*), Company (i.e., *Com*) participants, and the subprocess *co_s*2 (i.e., *SubCom*), respectively (see Figure 5), pointed out below:

$$\begin{aligned} CU = {} & \{start.1, cu\_s1, cu\_s2, cu\_s3, cu\_s4, cu\_s5, cu\_s6, \\ & xgate.1, end.1, abort.1\} \\ CO = {} & \{start.2, co\_s1, co\_s2, co\_s21, co\_s3, co\_s4, co\_s5, \\ & co\_s6, co\_s7, co\_s8, agate.1, agate.2, end.2, abort.2\} \\ CO2 = {} & \{start.3, co\_s21, end.3\} \end{aligned}$$

$$\begin{aligned} Cus = {} & \text{let } X = \Box i : (\alpha Y \backslash \{fin.1, abt.1\}) \bullet \\ & \qquad (i \rightarrow X \Box fin.1 \rightarrow SKIP \Box abt.1 \rightarrow STOP) \\ & \quad Y = (\|i : CU \bullet \alpha P(i) \circ P(i)) \\ & \quad \text{within}(Y \mid [\alpha Y] \mid X) \backslash \{\!| init.Cus |\!\} \\ Com = {} & \text{let } Z = \Box j : (\alpha R \backslash \{fin.2, abt.2\}) \bullet \\ & \qquad (j \rightarrow Z \Box fin.2 \rightarrow SKIP \Box abt.2 \rightarrow STOP) \\ & \quad R = (\|j : CO \bullet \alpha P(j) \circ P(j)) \\ & \quad \text{within}(R \mid [\alpha R] \mid Z) \backslash \{\!| init.Com |\!\} \end{aligned}$$

[3] Σ_i, Ω_i, and $\mathcal{L}(T(Pool_i))$, represents the set of input and output signals, and labelling, respectively, of the process $T(Pool_i)$.

[4] Here, duration times are expressed in seconds, according to the function sec defined in (Wong and Gibbons, 2008).

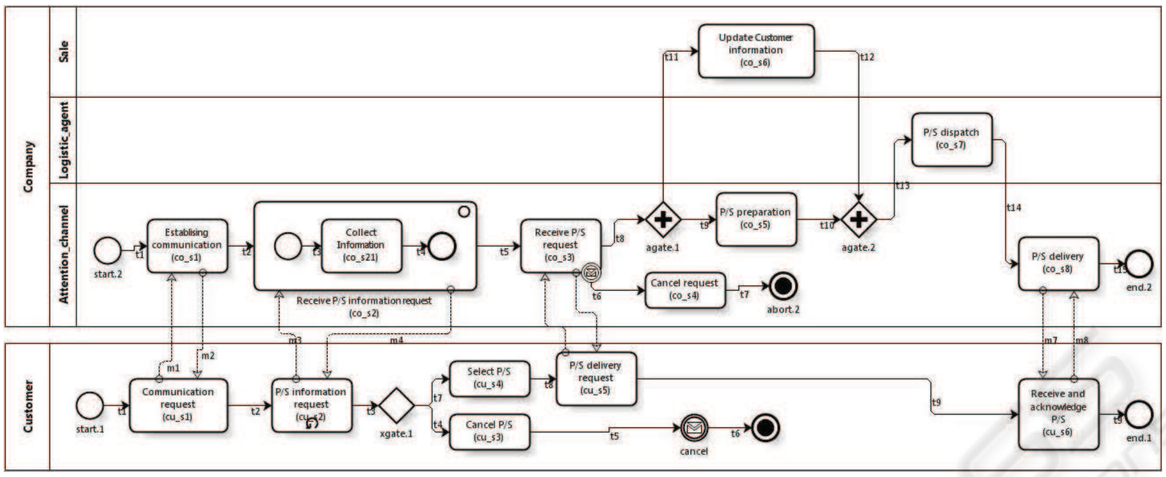

Company
Sale
Logistic_agent
Attention_channel
Customer
Update Customer information (co_s6)
P/S dispatch (co_s7)
Establising communication (co_s1)
Collect Information (co_s21)
Receive P/S information request (co_s2)
Receive P/S request (co_s3)
P/S preparation (co_s5)
Cancel request (co_s4)
P/S delivery (co_s8)
agate.1
agate.2
abort.2
start.2
end.2
Communication request (cu_s1)
P/S information request (cu_s2)
xgate.1
Select P/S (cu_s4)
Cancel P/S (cu_s3)
P/S delivery request (cu_s5)
Receive and acknowledge P/S (cu_s6)
cancel
start.1
end.1
m1
m2
m3
m4
m7
m8
t1
t2
t3
t4
t5
t6
t7
t8
t9
t10
t11
t12
t13
t14
t15

Figure 5: BPD of the *Product/service Sell* BP.

$$
\begin{aligned}
\mathit{SubCom} =& \text{let } T = \Box k : (\alpha W \setminus \{\mathit{fin}.3\}) \bullet (k \to T \Box \mathit{fin}.3 \to \mathit{SKIP})\\
& W = (\|k : \mathit{CO2} \bullet \alpha P(k) \circ P(k))\\
& \text{within}(W \mid [\alpha W] \mid T) \setminus \{\!| \mathit{init} |\!\}
\end{aligned}
$$

where for each $i \in CU$, $j \in CO$, and $k \in CO2$, the processes $P(i)$, $P(j)$, and $P(k)$, respectively, are defined next. We use $n \in \mathbb{N}$ to denote the number of *Product/Service information request (cu_s2)* Activity instances. We will only present some of the processes that make up the *Cus*, *Com*, and *SubCom* due to space limitations, to illustrate the application of the proposed semantics.

$$
P(\mathit{start}.1) = (t0.\star \to \mathit{init}.\mathit{Cus}.\mathit{cu\_s1} \to \mathit{SKIP}) \Box \mathit{fin}.1 \to \mathit{SKIP}
$$

$$
\begin{aligned}
P(\mathit{cu\_s2}) =&\\
\text{let } A(n) =& n > 0 \;\&\; (\mathit{init}.\mathit{Cus}.\mathit{cu\_s2} \to \mathit{SKIP} \fatsemi \mathit{starts}.\mathit{Cus}.\mathit{cu\_s2} \to \mathit{SKIP} \fatsemi\\
& \mathit{msg}.\mathit{cu\_s2}!x : \{\mathit{in}, \mathit{last}\} \to \mathit{SKIP} \fatsemi \mathit{msg}.\mathit{cu\_s2}.\mathit{out} \to\\
& \to \mathit{SKIP} \fatsemi\\
& \mathit{init}.\mathit{Cus}.\mathit{xgate}.1 \to \mathit{SKIP} \fatsemi A(n-1)) \Box \mathit{init}.\mathit{Cus}.\mathit{xgate}.1\\
& \to \to \mathit{SKIP}\\
X(n) =& (\mathit{init}.\mathit{Cus}.\mathit{cu\_s2} \to \mathit{SKIP} \Box \mathit{init}.\mathit{Cus}.\mathit{xgate}.1 \to \mathit{SKIP}) \fatsemi\\
& (n > 1 \;\&\; (\mathit{init}.\mathit{Cus}.\mathit{cu\_s2} \to (\mathit{msg}.\mathit{cu\_s2}.\mathit{in} \to X(n-1)\\
& \quad \Box \mathit{msg}.\mathit{cu\_s2}.\mathit{last} \to \mathit{init}.\mathit{Cus}.\mathit{xgate}.1 \to \mathit{SKIP}))\\
& \Box\; n = 1 \;\&\; (\mathit{init}.\mathit{Cus}.\mathit{cu\_s2} \to \mathit{msg}.\mathit{cu\_s2}.\mathit{last} \to\\
& \quad \mathit{init}.\mathit{Cus}.\mathit{xgate}.1 \to \mathit{SKIP})\\
& \Box\; n = N \;\&\; \mathit{msg}.\mathit{cu\_s2}.\mathit{end} \to \mathit{init}.\mathit{Cus}.\mathit{xgate}.1 \to \mathit{SKIP})\\
\text{within}&((A(n) \mid [\mathit{SynSet}] \mid X(n)) \fatsemi P(\mathit{cu\_s2})) \Box \mathit{fin}.1 \to \mathit{SKIP}\\
& \mathit{SynSet} = \{\mathit{msg}.\mathit{cu\_s2}.\mathit{in}, \mathit{msg}.\mathit{cu\_s2}.\mathit{last}, \mathit{init}.\mathit{Cus}.\mathit{cu\_s2},\\
& \qquad \mathit{init}.\mathit{Cus}.\mathit{xgate}.1\}
\end{aligned}
$$

$$
\begin{aligned}
P(\mathit{cancel}) =& (\mathit{init}.\mathit{Cus}.\mathit{cancel} \to \mathit{SKIP} \fatsemi \mathit{msg}.\mathit{cancel}!x : \{\mathit{can}\} \to \mathit{SKIP} \fatsemi\\
& \mathit{msg}.\mathit{cancel}.\mathit{out} \to \mathit{SKIP} \fatsemi \mathit{init}.\mathit{Cus}.\mathit{abort}.1 \to \mathit{SKIP} \fatsemi\\
& P(\mathit{cancel})) \Box \mathit{fin}.1 \to \mathit{SKIP}
\end{aligned}
$$

$$
P(\mathit{abort}.1) = (\mathit{init}.\mathit{Cus}.\mathit{abort}.1 \to \mathit{SKIP} \fatsemi \mathit{abt}.1 \to \mathit{STOP}) \Box \mathit{fin}.1 \to \mathit{SKIP}
$$

$$
\begin{aligned}
P(\mathit{co\_s2}) =& (\mathit{init}.\mathit{Com}.\mathit{co\_s2} \bowtie \mathit{vs2} \to \mathit{SKIP} \fatsemi \mathit{msg}.\mathit{co\_s2}!x : \{\mathit{in}, \mathit{last}\} \to\\
& \to \mathit{SKIP} \fatsemi\\
& \mathit{msg}.\mathit{co\_s2}.\mathit{out} \to \mathit{SKIP} \fatsemi \mathit{starts}.\mathit{Com}.\mathit{co\_s2} \to \mathit{SKIP} \fatsemi\\
& (\mathit{SubCom} \mid [\{\mathit{end}.3\}] \mid \mathit{end}.3 \to\\
& \mathrm{I}(86400 - 64800, \mathit{vs2} + 64800).\mathit{init}.\mathit{Com}.\mathit{co\_s3} \to \mathit{SKIP})\\
& \mid [\{\mathit{init}.\mathit{Com}.\mathit{co\_s3}\}] \mid\\
& \mathrm{I}(86400 - 64800, \mathit{vs2} + 64800).\mathit{init}.\mathit{Com}.\mathit{co\_s3} \to \mathit{SKIP}) \fatsemi\\
& \fatsemi P(\mathit{co\_s2}))\\
& \Box \mathit{fin}.2 \to \mathit{SKIP}
\end{aligned}
$$

$$
\begin{aligned}
P(\mathit{end}.2) =& \mathit{init}.\mathit{Com}.\mathit{end}.2 \to \mathit{SKIP} \fatsemi \mathit{fin}.2 \to \mathit{SKIP}\\
P(\mathit{co\_s21}) =& (\mathit{init}.\mathit{Com}.\mathit{co\_s21} \bowtie \mathit{vs21} \to \mathit{SKIP} \fatsemi \mathit{starts}.\mathit{Com}.\mathit{co\_s21} \to\\
& \to \mathit{SKIP} \fatsemi\\
& \mathrm{I}(1800, \mathit{vs21}).\mathit{init}.\mathit{Com}.\mathit{end}.3 \to \mathit{SKIP} \fatsemi P(\mathit{co\_s21})) \Box \mathit{fin}.3 \to\\
& \to \mathit{SKIP}
\end{aligned}
$$

Finally, the collaboration between the participants Customer and Company is the parallel composition of processes *Cus* and *Com*, as it is denoted by the process term CSP+T *PSS*:

$$
\mathit{PSS} = (\mathit{Cus} \mid [\alpha \mathit{Cus} \| \alpha \mathit{Com}] \mid \mathit{Com}) \setminus \{\!| \mathit{msg} |\!\}
$$

The set of processes previously described (*Cus*, *Com*, and *PSS*), conform the BPTM of the Product/Service Sell BP expressed in CSP+T. In this sense, this BPTM is the one to be verified with respect to the specified properties in CCTL that are presented in the next section.

5.2 Properties Definition

In order to show the application of our proposal, we will work with the following property, which is connected with the *obligation of receiving and obtaining the Product/Service delivery confirmation, once the Customer has initiated the communication with the Company*. As we will proceed with the verification of the BPTM behaviour (previously denoted as *PSS*) from the sub-processes that make it up (i.e., *Cus* and *Com*), by applying our compositional verification approach, then we must define the properties that each participant must fulfil, which show the execution sequence of BPMN notational elements expected when they execute the partial processes of whom it is responsible. The participants must execute all their activities as they are pointed out in the workflow in order to achieve the functioning of the global process. The partial properties, which we must verify in processes *Cus* and *Com*, respectively, to obtain the verification of process *PSS*, are defined below.

$$\begin{aligned}
\phi_{Cus} = {} & \mathsf{AG}_{[a,b]}(Start.1 \rightarrow \mathsf{A}[cu\_s1\ \mathsf{U}_{[a+1,b-5]}\ (cu\_s2 \wedge \\
& \mathsf{A}[cu\_s2\ \mathsf{U}_{[a+2,b-4]}\ (xgate.1\ \wedge \mathsf{A}[xgate.1\ \mathsf{U}_{[a+3,b-3]}\ (cu\_s4 \wedge \\
& \mathsf{A}[cu\_s4\ \mathsf{U}_{[a+4,b-2]}\ (cu\_s5\ \wedge \mathsf{A}[cu\_s5\ \mathsf{U}_{[a+5,b-1]}\ (cu\_s6 \wedge \\
& \mathsf{A}[cu\_s6\ \mathsf{U}_{[a+6,b]}\ End.1])])])])])])
\end{aligned}$$

$$\begin{aligned}
\phi_{Com} = {} & \mathsf{AG}_{[a,b]}(Start.2 \rightarrow \mathsf{A}[co\_s1\ \mathsf{U}_{[a+1,b-8]}\ (co\_s2 \wedge \\
& \mathsf{A}[cu\_s2\ \mathsf{U}_{[a+2,b-7]}\ (co\_s3\ \wedge \mathsf{A}[co\_s3\ \mathsf{U}_{[a+3,b-6]}\ (agate.1 \wedge \\
& \mathsf{A}[agate.1\ \mathsf{U}_{[a+4,b-5]}\ (\{co\_s5 \vee co\_s6\} \wedge \\
& \mathsf{A}[\{co\_s5 \vee co\_s6\}\ \mathsf{U}_{[a+6,b-3]}\ (agate.2 \wedge \\
& \mathsf{A}[agate.2\ \mathsf{U}_{[a+7,b-2]}\ (co\_s7\ \wedge \mathsf{A}[co\_s7\ \mathsf{U}_{[a+8,b-1]}\ (co\_s8 \wedge \\
& \mathsf{A}[co\_s8\ \mathsf{U}_{[a+9,b]}\ End.2])])])])])])])])
\end{aligned}$$

According to the CSP–based process calculus, the expected behaviour must be expressed according to the event sequence that should be observed as result of BPTM run. In this sense, we then have to interpret the prior property according to the expected sequence of events that the Product/Service Sell BP must show off in order to perform its verification. The operational interpretation CCTL formulas previously specified, according to the process calculus CSP+T, are the processes $T(\phi_{Cus})$ and $T(\phi_{Com})$ that are presented below and describe the expected behaviour for the participants that realize the BPTM.

$$\begin{aligned}
T(\phi_{Cus}) = {} & t_0.\star \rightarrow T(Start.1) \\
& T(Start.1) = \mathrm{I}((b-6)-a,a).init.Cus.cu\_s1 \rightarrow \\
& \rightarrow T(cu\_s1) \\
& T(cu\_s1) = \mathrm{I}((b-5)-(a+1),a+1).init.Cus.cu\_s2 \rightarrow \\
& \rightarrow T(cu\_s2) \\
& T(cu\_s2) = \mathrm{I}((b-4)-(a+2),a+2).init.Cus.xgate.1 \rightarrow \\
& \rightarrow T(xgate.1)) \\
& T(xgate.1) = \mathrm{I}((b-3)-(a+3),a+3).init.Cus.cu\_s4 \rightarrow \\
& \rightarrow T(cu\_s4) \\
& T(cu\_s4) = \mathrm{I}((b-2)-(a+4),a+4).init.Cus.cu\_s5 \rightarrow \\
& \rightarrow T(cu\_s5) \\
& T(cu\_s5) = \mathrm{I}((b-1)-(a+5),a+5).init.Cus.cu\_s6 \rightarrow \\
& \rightarrow T(cu\_s6) \\
& T(cu\_s6) = \mathrm{I}(b-(a+6),a+6).init.Cus.end.1 \rightarrow T(End.1) \\
& T(End.1) = SKIP \fatsemi T(\phi_{Cus})
\end{aligned}$$

$$\begin{aligned}
T(\phi_{Com}) = {} & t_0.\star \rightarrow T(Start.2) \\
& T(Start.2) = \mathrm{I}((b-9)-a,a).init.Com.co\_s1 \rightarrow T(co\_s1) \\
& T(co\_s1) = \mathrm{I}((b-8)-(a+1),a+1).init.Com.co\_s2 \rightarrow \\
& \rightarrow T(co\_s2) \\
& T(co\_s2) = \mathrm{I}((b-7)-(a+2),a+2).init.Com.co\_s3 \rightarrow \\
& \rightarrow T(co\_s3)) \\
& T(co\_s3) = \mathrm{I}((b-6)-(a+3),a+3).init.Com.agate.1 \rightarrow \\
& \rightarrow T(agate.1) \\
& T(agate.1) = (\mathrm{I}((b-5)-(a+4),a+4).init.Com.co\_s5 \rightarrow \\
& \rightarrow T(co\_s5)) \,\Box \\
& \qquad (\mathrm{I}((b-5)-(a+4),a+4).init.Com.co\_s6 \rightarrow \\
& \rightarrow T(co\_s6)) \\
& T(co\_s5) = (\mathrm{I}((b-4)-(a+5),a+5).init.Com.co\_s6 \rightarrow \\
& \rightarrow T(co\_s6)) \,\Box \\
& \qquad (\mathrm{I}((b-3)-(a+6),a+6).init.Com.agate.2 \rightarrow \\
& \rightarrow T(agate.2)) \\
& T(cu\_s6) = (\mathrm{I}((b-4)-(a+5),a+5).init.Com.co\_s5 \rightarrow T(co\_s5)) \,\Box \\
& \qquad (\mathrm{I}((b-3)(a+6),a+6).init.Com.agate.2 \rightarrow T(agate.2)) \\
& T(agate.2) = \mathrm{I}((b-2)-(a+7),a+7).init.Com.co\_s7 \rightarrow T(co\_s7) \\
& T(co\_s7) = \mathrm{I}((b-1)-(a+8),a+8).init.Com.co\_s8 \rightarrow T(co\_s8) \\
& T(co\_s8) = \mathrm{I}((b)-(a+9),a+9).init.Com.end.2 \rightarrow T(End.2) \\
& T(End.2) = SKIP \fatsemi T(\phi_{Com})
\end{aligned}$$

5.3 Verifying the Collaboration

Once obtained the set of CSP+T process terms that represent the BPTM as well the properties which it has to fulfil, we start to perform the verification of the BPTM. According to our approach, we must verify that the processes representing the behaviour of the participants in the BPTM (i.e., *Cus* and *Com*) fulfil the properties specified in section 5.2. Then, according to the semantic domain to which CSP calculus, it can be checked that the following refining relations are fulfilled:

$$T(\phi_{Cus}) \sqsubseteq_T Cus \quad , \quad T(\phi_{Com}) \sqsubseteq_T Com \qquad (2)$$

$$T(\phi_{Cus}) \sqsubseteq_F Cus \quad , \quad T(\phi_{Com}) \sqsubseteq_F Com \qquad (3)$$

To verify the above relationships, we are going to work according to the semantic model of CSP without temporal operators, since, as pointed out in (Schneider, 2000), untimed safety and liveness properties of a timed system should verifiable in the untimed model and later should be used in the timed analysis. Furthermore, this allows us to integrate the use of FDR2 tool to carry out the verification of processes that represent the participants. In the sequel we present the process terms CSP $UT(\phi_{Com})$ and $UT(\phi_{Cus})$, which correspond to the expected untimed behaviour of untimed processes $UT(Com)$ and $UT(Cus)$ (which are not shown due to space limitations), respectively, of Customer and Company participants:

$$\begin{aligned}
UT(\phi_{Cus}) = {} & \star \rightarrow UT(Start.1) \\
& UT(Start.1) = init.Cus.cu\_s1 \rightarrow UT(cu\_s1) \\
& UT(cu\_s1) = init.Cus.cu\_s2 \rightarrow UT(cu\_s2) \\
& UT(cu\_s2) = init.Cus.xgate.1 \rightarrow UT(xgate.1)) \\
& UT(xgate.1) = init.Cus.cu\_s4 \rightarrow UT(cu\_s4) \\
& UT(cu\_s4) = init.Cus.cu\_s5 \rightarrow UT(cu\_s5) \\
& UT(cu\_s5) = init.Cus.cu\_s6 \rightarrow UT(cu\_s6) \\
& UT(cu\_s6) = init.Cus.end.1 \rightarrow UT(End.1) \\
& UT(End.1) = SKIP \fatsemi UT(\phi_{Cus})
\end{aligned}$$

$$\begin{aligned}
UT(\phi_{Com}) = {} & \star \rightarrow UT(Start.2) \\
& UT(Start.2) = init.Com.co\_s1 \rightarrow UT(co\_s1) \\
& UT(co\_s1) = init.Com.co\_s2 \rightarrow UT(co\_s2) \\
& UT(co\_s2) = init.Com.co\_s3 \rightarrow UT(co\_s3)) \\
& UT(co\_s3) = init.Com.agate.1 \rightarrow UT(agate.1) \\
& UT(agate.1) = (init.Com.co\_s5 \rightarrow UT(co\_s5)) \,\Box \\
& \qquad (init.Com.co\_s6 \rightarrow UT(co\_s6)) \\
& UT(co\_s5) = (init.Com.co\_s6 \rightarrow UT(co\_s6)) \,\Box \\
& \qquad (init.Com.agate.2 \rightarrow UT(agate.2)) \\
& UT(cu\_s6) = (init.Com.co\_s5 \rightarrow UT(co\_s5)) \,\Box \\
& \qquad init.Com.agate.2 \rightarrow UT(agate.2))
\end{aligned}$$

$$\begin{aligned}UT(agate.2) =& init.Com.co\_s7 \rightarrow UT(co\_s7)\\ UT(co\_s7) =& \; init.Com.co\_s8 \rightarrow UT(co\_s8)\\ UT(co\_s8) =& \; init.Com.end.2 \rightarrow UT(End.2)\\ UT(End.2) =& \; SKIP \fatsemi UT(\phi_{Com})\end{aligned}$$

According to the timewise refinement concept (Schneider, 2000), the description of an untimed process sets constraints on the ordering and ultimate availability of events, and allows all timed behaviours that are consistent with its description. In this sense, we can write the following relations:

$$T(\phi_{Cus}) \sqsubseteq_T Cus \;\Rightarrow\; UT(\phi_{Cus}) \sqsubseteq_T UT(Cus), \quad (4)$$

$$T(\phi_{Com}) \sqsubseteq_T Com \;\Rightarrow\; UT(\phi_{Com}) \sqsubseteq_T UT(Com), \quad (5)$$

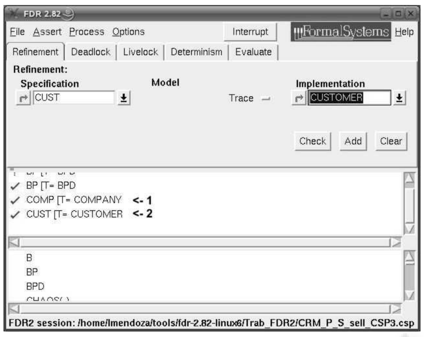
$$T(\phi_{Cus}) \sqsubseteq_F Cus \;\Rightarrow\; UT(\phi_{Cus}) \sqsubseteq_F UT(Cus), \quad (6)$$

$$T(\phi_{Com}) \sqsubseteq_F Com \;\Rightarrow\; UT(\phi_{Com}) \sqsubseteq_F UT(Com), \quad (7)$$

which establish that the verification of untimed terms in CSP is a necessary condition for the verification of timing CSP+T terms. This allows us to check the timed component behaviour on the basis of the sequence events admitted by the untimed CSP model, excluding from analysis the events sequence that may not correspond with the correct order of events, resulting from the aggregation of the timing constraints of timed CSP+T model. This will minimize the state explosion problem because MC tools works over an untimed model of the system that is smaller, and corresponds directly with the correct event sequence execution of the timed model.

Thus, the behaviour of the participants Customer and Company specified in CSP are verified w.r.t. the semantic domains of traces and failures, which ensures that safety and liveness properties are satisfied, respectively. Then, we can obtain that the behaviour of the *Cus* and *Com* process terms are correct, i.e., all timed behaviour of CSP+T process terms are consistent with its description. In other words, the timewise refinement of CSP+T process terms is consistent with the untimed description of CSP process terms, and these impose further constraints upon the timed behaviour of CSP+T process terms. Thus, the relations (2) and (3) are true.

Consequently, we consider that the behaviour verification of constituent participants of the BPTM should be performed using the FDR2 MC tool, since we are working with an algebra based on CSP, such as CSP+T. As can be observed in the FDR2 screenshot in Figure 6, the verification of local BP of each participant untimed model in CSP, COMPANY (i.e., $UT(Com)$) and CUSTOMER (i.e., $UT(Cus)$), of the BPTM for *Product/Service Sell* BP satisfies the untimed expected behaviour of each, COMP (i.e., $UT(\phi_{Com})$) and CUST (i.e., $UT(\phi_{Cus})$), respectively (see check marks at rows one and two, respectively).


FDR 2.82
File Assert Process Options
Interrupt
FormalSystems
Help
Refinement Deadlock Livelock Determinism Evaluate
Check Add Clear

Refinement:
Specification: CUST
Model: Trace
Implementation: CUSTOMER

✓ BP [T= BPD
✓ COMP [T= COMPANY <- 1
✓ CUST [T= CUSTOMER <- 2

B
BP
BPD

FDR2 session: /home/lmendoza/tools/fdr-2.82-linux6/Trab_FDR2/CRM_P_S_sell_CSP3.csp

Figure 6: FDR2 screenshot.

According to relation (1) (see section 4), to prove the correctness of the BPTM of the *Product/Service Sell* BP w.r.t. its expected behaviour, it must be demonstrated that:

$$PSS \models \phi_{PSS} \;\Leftrightarrow\; (Cus \mid [\alpha Cus \| \alpha Com] \mid Com) \setminus \{\!| \, msg \, |\!\} \;\models\; \phi_{Cus} \wedge \phi_{Com} \;.$$

We have verified with FDR2 that:

$$Cus \models \phi_{Cus} \;\text{ and }\; Com \models \phi_{Com} \;.$$

Based on the detailed design of *Cus* and *Com* local BP shown in Figure 5, we must determine whether these local BPs are “composable”. Thus, we must verify that it fulfills the following 2 conditions:

1. The input signals (Σ_{Cus} and Σ_{Com}) and the output signals (Ω_{Cus} y Ω_{Com}) of both local BP are disjointed, which can be seen below:

$$\Sigma_{Cus} \cap \Sigma_{Com} = \varnothing \quad (8)$$

$$\begin{aligned}\Sigma_{Cus} =& \{msg.cu\_s1.out, msg.cu\_s2.out, msg.cancel.out,\\ & msg.cu\_s5.out, msg.cu\_s6.out\}\\ \Sigma_{Com} =& \{msg.co\_s1.out, msg.co\_s2.out, msg.co\_s3.out,\\ & msg.co\_s3.can, msg.co\_s8.out\}\end{aligned}$$

$$\Omega_{Cus} \cap \Omega Com = \varnothing \quad (9)$$

$$\begin{aligned}\Omega_{Cus} =& \{msg.cu\_s1.in, msg.cu\_s1.last, msg.cu\_s2.in,\\ & msg.cu\_s2.last, msg.cancel.can, msg.cu\_s5.in,\\ & msg.cu\_s5.last, msg.cu\_s6.in, msg.cu\_s6.last\}\\ \Omega_{Com} =& \{msg.co\_s1.in, msg.co\_s1.last, msg.co\_s2.in,\\ & msg.co\_s2.last, msg.co\_s3.in, msg.co\_s3.last, \}\\ & msg.co\_s8.in, msg.co\_s8.lastmsg.co\_s8.last\}\end{aligned}$$

2. The labelling sets of both components, $\mathcal{L}(Cus)$ and $\mathcal{L}(Com)$, are disjointed, which can also be verified as follows:

$$\mathcal{L}(Cus) \cap \mathcal{L}(Com) = \varnothing \quad (10)$$

$$\begin{aligned}\mathcal{L}(Cus) =& \{start.1, cu\_s1, cu\_s2, cu\_s3, cu\_s4, cu\_s5, cu\_s6,\\ & xgate.1, end.1, abort.1\}\\ \mathcal{L}(Com) =& \{start.2, co\_s1, co\_s2, co\_s21, co\_s3, co\_s4, co\_s5,\\ & co\_s6, co\_s7, co\_s8, agate.1, agate.2, end.2, abort.2\}\end{aligned}$$

Having verified that the relations (8), (9), and (10), are true, we conclude that *Cus* and *Com* are “composable”. By the BPTM compositional verification theorem (see section 4), we have:

$$(Cus \mid [\alpha Cus \| \alpha Com] \mid Com) \setminus \{\!| \, msg \, |\!\} \models \phi_{Cus} \wedge \phi_{Com}$$

and because

$$PSS = (Cus \mid [\alpha Cus \| \alpha Com] \mid Com) \setminus \{\!|\, msg \,|\!\} \text{ and } \phi_{PSS} = \phi_{Cus} \wedge \phi_{Com},$$

we obtain $PSS \models \phi_{PSS}$.

Finally, we have obtained the verification of a BPTM corresponding to the *Product/Service Sell* BP from their verified local BP, *Customer* and *Company*. Therefore, we can claim that our approach has been successfully applied to an instance of CRM business. Finally, we can affirm that our approach may be a means to precise the semantic of BPMN and to perform the verification of complex global BP modelled with BPMN from collections of its verified local BPs.

6 CONCLUSIONS

In this paper we have presented FCVA for compositional global BP verification from independently verified local BPs performed by the bp participants. Also is proposed to complement the FVCA with a timed semantics of BPMN defined in terms of CSP+T formal specification language, which extends the BPMN elements with timing constrains in order to detail the behaviour that they represent. We have shown the value and practicality of our approach by means of the application to a real–life BP in the field of CRM, which has to meet timed collaboration requirements. The CSP+T specification of the BPTM at the design phase can be verified against the CCTL specification of the BP properties. As a consequence, the complete BPTM developed from its core participants can also be proved correct by means of the formal language CSP+T that allows local verification results of CSP+T syntactical terms —representing individual local BPs— to be exported into the entire global BP verification, which is obtained as a concurrent composition of process terms.

Future and ongoing work will focus on the application of FCVA and the timed semantics of BPMN to other BPs verification; our goal is to conduct in–depth research on verification of these specifications, and to obtain tool supporting BPM by using state–of–the–art verification tools.

REFERENCES

Aalst, W. (2002). *Making Work Flow: On the Application of Petri Nets to Business Process Management, LNCS 2360: Application and Theory of Petri Nets 2002*, pages 1–22. Springer–Verlag, Berlin.

Alur, R. and Dill, D. (1994). A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235.

Capel, M., Mendoza, L., and Benghazi, K. (2008). Automatic verification of business process integrity. *Int. J. Simulation and Process Modelling*, 4(3/4):167–182.

Clarke, E., Grumberg, O., and Peled, D. (2000). *Model Checking*. MIT. The MIT Press, Cambridge, USA.

Díaz, G., Pardo, J.-J., Cambronero, M.-E., Valero, V., and Cuartero, F. (2005). *Automatic Translation of WS–CDL Choreographies to Timed Automata, LNCS 3670: Formal Techniques for Computer Systems and Business Processes*, pages 230–242. Springer–Verlag, Berlin.

Eriksson, H.-E. and Penker, M. (1998). *Business Modeling With UML: Business Patterns at Work*. John Wiley & Sons, Inc., New York, USA.

Kruchten, P. (2003). *The Rational Unified Process: An Introduction, Second Edition*. Addison-Wesley Longman Publishing Co., Inc., Boston, USA, 3rd edition.

Ma, S., Zhang, L., and He, J. (2008). Towards formalization and verification of unified business process model based on pi calculus. *Proc. ACIS International Conference on Software Engineering Research, Management and Applications*, 1:93–101.

Mendoza, L., Marius, A., Pérez, M., and Grimán, A. (2007). Critical success factors for a customer relationship management strategy. *Inf. Softw. Technol.*, 49(8):913–945.

Morimoto, S. (2005). *A Survey of Formal Verification for Business Process Modeling, LNCS 5102: Proc. 8th International Conference on Computational Science (ICCS 2008)*, pages 514–522. Springer–Verlag, Berlin.

OASIS (2007). *Web Services Business Process Execution Language Version 2.0*. OASIS Open, Billerica, USA.

OMG (2009). *Business Process Modeling Notation – version 1.2*. Object Management Group, Massachusetts, USA.

Paternò, F. (2001). *Handbook of Software Engineering And Knowledge Engineering: Recent Advances*, chapter Task Models in Interactive Software Systems. World Scientific Publishing Co., Inc., River Edge, USA.

Roscoe, A. (1997). *The Theory and Practice of Concurrency*. Prentice–Hall International Ltd., Hertfordshire UK.

Rüf, J. and Kropf, T. (1997). Symbolic model checking for a discrete clocked temporal logic with intervals. In *Proceedings of the IFIP WG 10.5 International Conference on Correct Hardware Design and Verification Methods*, pages 146–163, London, UK. Chapman & Hall, Ltd.

Schneider, S. (2000). *Concurrent and Real–Time Systems – The CSP Approach*. John Wiley & Sons, Ltd., Chichester, England.

Žic, J. (1994). Time–constrained buffer specifications in CSP+T and Timed CSP. *ACM Transaction on Programming Languages and Systems*, 16(6):1661–1674.

Wong, P. and Gibbons, J. (2008). *A Process Semantics for BPMN, LNCS 5256: Proc. 10th International Conference on Formal Engineering Methods (ICFEM 2008)*, pages 355–374. Springer–Verlag, Berlin.