Compositional Verification of Business Processes by Model–Checking

Luis E. Mendoza[1], Manuel I. Capel[2] and María Pérez[1]

[1] Processes and Systems Department, Simón Bolívar University
P.O. box 89000, Baruta, Caracas 1080-A, Venezuela

[2] Software Engineering Department, University of Granada
ETSI Informatics and Telecommunication, 18071 Granada, Spain

Abstract. The work presented in this article is aimed at a contribution to the *Enterprise Information Systems* (EIS) verification. We describe here a *Formal Compositional Verification Approach* (FCVA) —based on Model–Checking (MC) techniques— applied to the verification of *Business Process* (BP) models represented by *Business Process Modelling Notation* (BPMN) diagrams. FCVA is compositional and thus allows the verification of a complex BP model carried out from verification of its parts. FCVA and a proposal of temporal semantics for BPMN allows the expression of time–dependent constructs of *BP Task Models* (BPTM) supported by an EIS. The interpretation of the BPMN graphical modelling entities into a formal specification language (CSP+T) allows us to use state–of–the–art MC tools to verify the behavioural part of BP models. A real–life example in the field of the *Customer Relationship Management* (CRM) business is presented to demonstrate the FCVA application in a practical way.

1 Introduction

Enterprise Information Systems (EIS) manage enterprise business, apply strategic and economic decisions, and hold communication with business partners. In this sense, the EIS implements cross–functional *Business Processes* (BPs), i.e., the set of *ways in which management chooses to coordinate the work* to achieve their (*business*) *objectives* and *user goals*, which transcends the boundaries between sales, marketing, manufacturing, and research and development. Therefore, an organization must have been obtained previously, as result of the *Business Process Modelling* (BPM), the complete definition of the set of BPs that support the EIS. Due to BPs specific characteristics (people integration, business rules, business goals, events, information, and resources) [1], the validation of *BP Task Model* (BPTM) is an extremely expensive and risky activity if it is delayed until the EIS deployment phase.

The main goal of *Business Process Modelling Notation* (BPMN) [1] being to provide a readily understandable notation for all its users, the lack of a precise semantics of its modelling entities impedes rigourous analysis and reasoning about the models obtained [2]. To cope with the above described situation, we propose an instantiation of our compositional verification framework, called *Formal Compositional Verification*

E. Mendoza L., I. Capel M. and PÃ¡rez M.
Compositional Verification of Business Processes by ModelâĂŞChecking.
DOI: 10.5220/0003022300600069
In *Proceedings of the 8th International Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems (ICEIS 2010)*, page
ISBN: 978-989-8425-12-6

Approach (FCVA) [3], which uses MC techniques and makes it possible to verify a BPTM supported by an EIS using the formal semantics of *Communicating Sequential Processes* (CSP) –based process calculus. We complement our FVCA [3] with a timed semantics of BPMN defined in terms of the *Communicating Sequential Processes + Time* (CSP+T) [4] formal specification language, which extends BPMN modelling entities with timing constraints in order to allow the expression of BPTM time–dependent constructs. By a sound interpretation of FCVA elements into *Kripke Structures* (KS) [5], it then becomes feasible to verify the behaviour of global BP (i.e., the BPTM) from its local BPs' participants.

Different works address the verification and validation of BP modelled with BPMN. In [6] is presented a extended survey of recently proposed verification techniques for verifying BPMN models and a comparison between them and with respect to motivations, methods, and logics. Differently from other research, our work is aimed at giving a systemic, integrated vision of specification, design and verification of BPTM derived from BPs, by incorporating the use of MC tools in the specification and verification of BPTM into the EIS development cycle.

The remainder of this paper is organised as follows. In section 2 short introductions to time semantics for BPMN modelling entities and to the Clocked Computation Tree Logic (CCTL) specification language are provided. In section 3 FCVA for BPMN verification is presented, followed by a formal description and validation of the compositional verification proposal. Section 4 describes the application to a BPM example related to the CRM business. Finally, in Section 5, conclusions are given and future work is described.

2 BPTM's Behaviours in a Common Semantic Domain

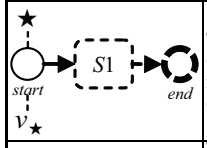
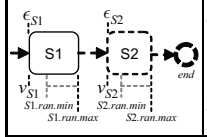
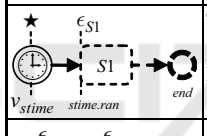
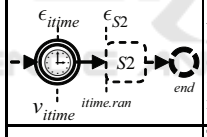
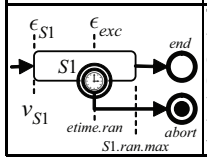
Most temporal logics and other system description formalisms, used for reactive systems (as BPTM) specification, can be interpreted as KS. According to [5] the systems best suited to verification by MC are those that are easily modelled by (finite) automata, such as KS ones [5]. Accordingly, [7] states that *translating formulae in temporal logics to automata is a standard approach for implementing MC.* Therefore, in this paper we use *Timed Büchi Automaton* (TBA) because these are the simplest automata over infinite words [5] able to represent time regular processes [8].

2.1 BPTM Model

To obtain a complete description of the BPTM's behaviour *interpreted* into CSP+T process terms, we apply the transformation rules that we briefly introduce below, which assume the semantics of the BPMN analysis entities given in [2] as the starting point for their definition. As a result of a mapping from BPMN [1] to CSP+T processes, each BPMN modelling entity (flow objects, connecting objects, and swimlanes) yield a syntactical sequential process term and specifies how to represent the entire participant's behaviour, according to discrete timed events and sequences of events. Due to space limitations, Table 1 only shows a graphical example of some transformation rules used for obtaining CSP+T process terms from BPMN modelling entities. The complete rules

set is presented in [9]. We denote as ϵ_x the invocation events of the BPMN modelling entities, $Sx.ran.min$ and $Sx.ran.max$ as the minimum and maximum time span of Sx *activities*, respectively, and $stime.ran$ and $itime.ran$ as the time delay defined by *timer start* and *timer intermediate* events, respectively, according to BPMN [1]. Briefly explained, the transformation is performed by mapping: (1) every BPMN modelling entity to a prefixed CSP+T process term; (2) every discrete duration time to a CSP+T event–enabling interval; and (3) the external choice to alternative selections performed by the environment of each process is applied to ensure that all processes terminate at the *end* of the business process execution.

Table 1. Some mapping rules from BPMN modelling entities to CSP+T terms.

BPMN element	Description	CSP+T process
	The *start event* corresponds to the CSP+T $\star$ instantiation event and the $v_\star$ marker variable is used to save the occurrence time of event $\star$.	$P(start) = (\star \bowtie v_\star \rightarrow SKIP \fatsemi P(start)) \Box (\epsilon_{end} \rightarrow SKIP)$
	The $S2$ *activity* begins when the ϵ_{S1} event occurs and the invocation of $S2$ activity (i.e., the occurrence of ϵ_{S2} event) must occur within the $[S1.ran.min, S1.ran.max]$ time interval. The activity $S1$ come before activity $S2$.	$P(S1) = (\epsilon_{S1} \bowtie v_{S1} \rightarrow SKIP \fatsemi \mathrm{I}(S1.ran.max - S1.ran.min, v_{S1} + S1.ran.min).\epsilon_{S2} \rightarrow SKIP \fatsemi P(S1)) \Box (\epsilon_{end} \rightarrow SKIP)$
	The *timer start* event establishes that the $S1$ activity must begin (i.e., the occurrence of ϵ_{S1} event), $stime.ran$ time units after the occurrence of $\star$ instantiation event.	$P(stime) = (\star \bowtie v_{stime} \rightarrow SKIP \fatsemi \mathrm{I}(stime.ran, v_{stime}) \rightarrow SKIP \fatsemi \epsilon_{S1} \rightarrow SKIP \fatsemi P(stime)) \Box (\epsilon_{end} \rightarrow SKIP)$
	According to the *timer intermediate* event, the $S2$ activity must begin (i.e., the occurrence of ϵ_{S2} event), $itime.ran$ time units after the occurrence of ϵ_{itime} event.	$P(itime) = (\epsilon_{itime} \bowtie v_{itime} \rightarrow SKIP \fatsemi \mathrm{I}(T_{itime}, v_{itime}) \rightarrow SKIP \fatsemi \epsilon_{S2} \rightarrow SKIP \fatsemi P(itime)) \Box (\epsilon_{end} \rightarrow SKIP)$
	The $S1$ activity execution can be interrupted (i.e., the occurrence of ϵ_{exc} event) at any time since its inception (i.e., the occurrence of ϵ_{S1} event) and until its total duration ends (i.e., within the $[v_{S1}, S1.ran.max]$ time interval).	$P(S1) = (\epsilon_{S1} \bowtie v_{S1} \rightarrow SKIP \fatsemi \mathrm{I}(S1.ran.max - S1.ran.min, v_{S1} + S1.ran.min).\epsilon_{end} \rightarrow (SKIP \bigwedge \mathrm{I}(S1.ran.max, v_{S1}).\epsilon_{exc} \rightarrow SKIP \fatsemi abort.1 \rightarrow STOP) \fatsemi P(S1)) \Box (\epsilon_{end} \rightarrow SKIP)$

2.2 BPTM Properties

To specify the properties that the BPTM must exhibit, we use the CCTL [10], which is an *interval temporal logic* that allow us to carry out a logical reasoning at the level of time intervals, instead of instants. See [10] for more details. The algorithm described in [8] is used to construct a discrete TBA semantically equivalent to a CCTL formula ϕ. Afterwards, using the procedure described in [11], the TBAs of the BPTM properties described previously are transformed into CSP+T process terms. Thus, the expected behaviour of a BPTM is interpreted into a CSP+T process term P. Thus, the assertion $P \preceq \phi$ denotes that P meets the specification ϕ, where $\preceq$ represents that P simulates

ϕ (*the simulation assertion*), meaning that any behaviour of ϕ can be matched by a corresponding behaviour of P (but not necessarily vice versa). Consequently, by applying the rules in Table 1 and the simulation operator, we can reason and express the BPTM properties in the same specification language as the BPTM model.

3 Compositional Verification Approach

Our approach is based on the fact that the system C has been structured into several verified components working in parallel, $C = ||_{i:1..n} C_i$, where each component C_i satisfies the property ϕ_i, which represents the specification of the expected behaviour for the component. Our main goal here is to make possible the verification of the entire system's behaviour from its verified components. In this sense,

Definition 1 (Property compositionality). *A property ϕ is compositional iff for any two TBA $\mathcal{A}_1$, $\mathcal{A}'_1$, and $\mathcal{A}_2$ with $\mathcal{L}(\mathcal{A}_2) \cap \mathcal{L}(\phi) = \emptyset$ holds*

$$(\mathcal{A}_1 \vDash \phi) \Rightarrow ((\mathcal{A}_1 \| \mathcal{A}_2 \vDash \phi) \vee \mathcal{A}_1 \| \mathcal{A}_2 \vDash \delta)) \text{ and} \tag{1}$$

$$((\mathcal{A}_1 \sqsubseteq \mathcal{A}'_1) \wedge (\mathcal{A}'_1 \vDash \phi)) \Rightarrow (\mathcal{A}_1 \vDash \phi) \tag{2}$$

Local properties are preserved by parallel composition when the labelling is disjoint:

Lemma 1. *For two TBAs $\mathcal{A}_1$ and $\mathcal{A}_2$ and properties ϕ_1 and ϕ_2 with $\Sigma_1 \cap \Omega_2 = \emptyset$, $\Sigma_2 \cap \Omega_1 = \emptyset$, $\mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2) = \emptyset$ holds:*

$$((\mathcal{A}_1 \vDash \phi_1) \wedge (\mathcal{A}_2 \vDash \phi_2)) \Rightarrow (A_1 \| A_2 \vDash \phi_1 \wedge \phi_2). \tag{3}$$

On the other hand, it is also a requirement that composition preserves refinement in the case of parallel composition:

Lemma 2. *For two composable TBAs $\mathcal{A}_1$ and $\mathcal{A}_2$, and any automata $\mathcal{A}'_2$ holds*

$$\mathcal{A}_2 \sqsubseteq \mathcal{A}'_2 \Rightarrow (\mathcal{A}_1 \| \mathcal{A}_2 \sqsubseteq \mathcal{A}_1 \| \mathcal{A}'_2). \tag{4}$$

Each component must also satisfy the "invariant" (ψ_i) expression which represents the behaviour of other system components with respect to C_i. The special symbol $\neg\delta$ is used to denote that *deadlock* (i.e., a state without any outgoing transition) cannot be reached. The property ϕ and invariant ψ that are satisfied by the system C, have been obtained from the local properties ϕ_i (i.e., $\bigwedge_{i:1..n} \phi_i \Rightarrow \phi$) and invariances ψ_i (i.e., $\bigwedge_{i:1..n} \psi_i \Rightarrow \psi$), respectively. As result, we can obtain the complete verification of the system by using the Theorem 1:

Theorem 1 (System Compositional Verification). *Let the system C be structured into several components working in parallel, $C = ||_{i:1..n} C_i$. For a set of $TBA(C_i)$ describing the behaviour of components C_i, properties ϕ_i, invariants ψ_i, and deadlock δ, with $\bigcap_{i:1..n} \Sigma_i = \emptyset$, $\bigcap_{i:1..n} \Omega_i = \emptyset$, and $\bigcap_{i:1..n} \mathcal{L}(TBA(C_i)) = \emptyset$, the following condition holds:*

$$TBA(C) \vDash (\phi \wedge \psi \wedge \neg\delta) \Leftrightarrow \bigparallel_{i:1..n} TBA(C_i) \vDash \bigwedge_{i:1..n} (\phi_i \wedge \psi_i) \wedge \neg\delta, \tag{5}$$

where $TBA(C) = \|_{i:1..n} TBA(C_i)$.

The practical application of assertion (5) includes (manually) performing an inductive *satisfaction checking* process on the range of the components number ($i : 1..n$) of the system. The FDR2 [12] model checker can automate this proof.

Based on previous concepts and ideas, we propose a possible instantiation of our conceptual scheme called FCVA [3], as shown in Fig. 1, to specify and verify BPTM derived from BPs supported by EIS. The rationale of FCVA instantiation is that the behavioural correctness of local BPs can be individually verified, in isolation, based on the well–defined communication behaviour specified by their message flows, and verification of the global BP behaviour performed using the results of the verification of local BPs. Our instantiation uses the CSP+T process calculus, which has a simple but powerful form of composition given by concurrent composition and hiding operators.

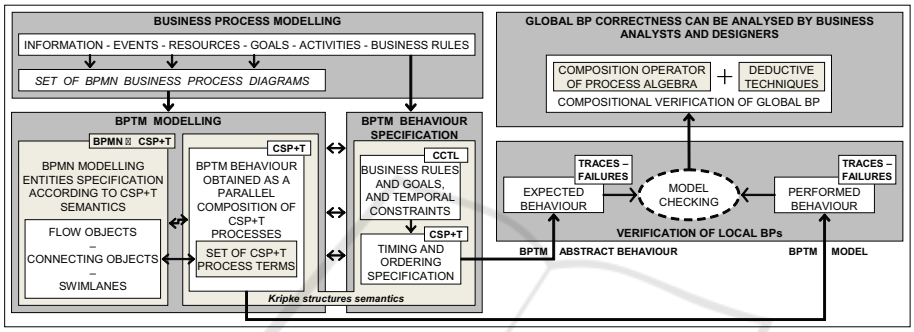


Fig. 1. Integrated view of compositional verification for BPTM.

The BPM is considered outside the scope of FCVA. Both the formal description of the BPTM behaviour and the specification of its properties must be directed by the BPMN *Business Process Diagram* (BPD) and the business rules and goals, respectively. FCVA instantiation consists of the following integrated processes (see Fig. 1):

BPTM Modelling. Firstly, the complete description of the BPTM's behaviour, modelled by the CSP+T process term $T(C)$ is *interpreted* into a set of CSP+T process terms $T(C_i)$ by using the proposed time semantics for BPMN modelling entities introduced in section 2.1.

BPTM Behaviour Specification. Then, requirements and temporal constraints that the BPTM must fulfill are *specified* in CCTL, which is based on the interval structure and time–annotated automata [10]. Afterwards, these properties are expressed by CSP+T process terms $T(\phi_i)$, $T(\psi_i)$, $T(\neg\delta)$.

Verification. Finally, by performing the following steps, we proceed to verify the BPTM behaviour:

1. Firstly, the local process $T(C_i)$ representing the local BPs are *model checked* against the set of process terms $T(\phi_i)$, and $T(\psi_i)$, $T(\neg\delta)$. According to the trace and failure semantics of CSP–based algebra, we proceed to verify:

$$T(\phi_i) \sqsubseteq_T T(C_i) \wedge T(\psi_i) \sqsubseteq_T T(C_i) \wedge T(\neg\delta) \sqsubseteq_T T(C_i)$$
$$T(\phi_i) \sqsubseteq_F T(C_i) \wedge T(\psi_i) \sqsubseteq_F T(C_i) \wedge T(\neg\delta) \sqsubseteq_F T(C_i)$$

2. Secondly, we obtain the verification of local BPs correctness, according to the following assertions:

– Related to consideration of safety issues:

$$\forall t \in \text{traces}(T(\phi_i)) \exists t' \in \text{traces}(T(C_i)) : t' \Rightarrow \phi_i \Leftrightarrow T(C_i) \models \phi_i$$
$$\forall t \in \text{traces}(T(\psi_i)) \exists t' \in \text{traces}(T(C_i)) : t' \Rightarrow \psi_i \Leftrightarrow T(C_i) \models \psi_i$$
$$\forall t \in \text{traces}(T(\neg\delta)) \exists t' \in \text{traces}(T(C_i)) : t' \Rightarrow \neg\delta \Leftrightarrow T(C_i) \models \neg\delta$$

– Related to consideration of liveness issues:

$$\forall (t, X) \in \mathcal{SF}[\![T(\phi_i)]\!] \exists (t', X) \in \mathcal{SF}[\![T(C_i)]\!] : (t', X) \Rightarrow \phi_i \Leftrightarrow T(C_i) \models \phi_i$$
$$\forall (t, X) \in \mathcal{SF}[\![T(\psi_i)]\!] \exists (t', X) \in \mathcal{SF}[\![T(C_i)]\!] : (t', X) \Rightarrow \psi_i \Leftrightarrow T(C_i) \models \psi_i$$
$$\forall (t, X) \in \mathcal{SF}[\![T(\neg\delta)]\!] \exists (t', X) \in \mathcal{SF}[\![T(C_i)]\!] : (t', X) \Rightarrow \neg\delta \Leftrightarrow T(C_i) \models \neg\delta$$

3. Finally, by the application of Theorem 1 we obtain the complete verification of the BPTM behaviour $T(C)$, according to the assertion (5) instantiated for CSP+T process terms ($T(C) = \|_{i:1..n}\ T(C_i)$).

4 Example of Application

To show the applicability of our proposal, it was applied to a BPM enterprise–project related to the CRM business. 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 carrying out formal reasoning about the CRM participant collaboration is displayed in the *Product/Service Sell* BPD shown in Fig. 2, which allows a *Company* to perform the activities associated with selling a Product/Service requested by a *Customer*. As shown in Fig. 2, the BP depicts a high collaboration between the participants to achieve their execution, which means a synchronization of the activities involved in message flows.

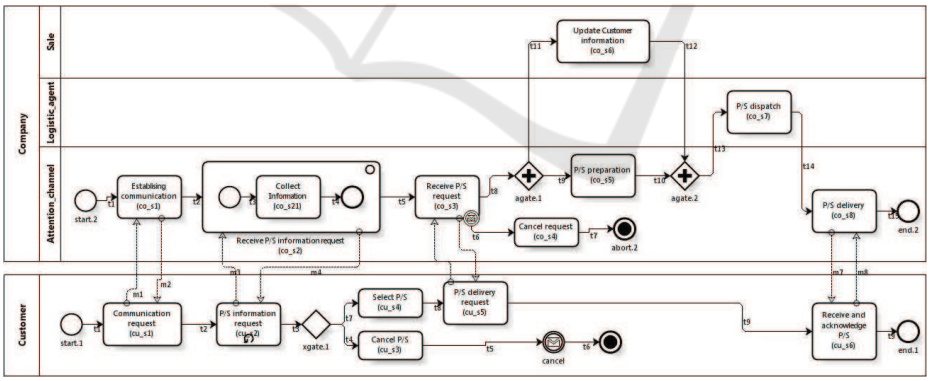


Fig. 2. BPD of the *Product/service Sell* BP.

4.1 BPTM Definition and Description

To obtain the specification of the Product/Service Sell BPD in CSP+T, according to the proposal briefly described in section 2.1, we define the sets CU and CO, for indexing the processes mapped to the modelling entities of Customer (i.e., Cus) and Company (i.e., Com) participants, respectively (see Fig. 2):

$$CU = \{start.1, cu\_s1, cu\_s2, cu\_s3, cu\_s4, cu\_s5, cu\_s6, xgate.1, end.1, abort.1\}$$

$$CO = \{start.2, co\_s1, co\_s2, co\_s21, co\_s3, co\_s4, co\_s5, co\_s6, co\_s7, co\_s8, agate.1, agate.2, end.2, abort.2\}$$

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

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

where for each $i \in CU$ and $j \in CO$, the processes $P(i)$ and $P(j)$, respectively, are defined next. Due to space limitations, we will only present some of the processes that make up the Cus and Com, to illustrate the application of the proposed semantics[1].

$$P(start.1) = (t0.\star \rightarrow init.Cus.cu\_s1 \rightarrow SKIP) \Box fin.1 \rightarrow SKIP$$

$$\begin{aligned} P(co\_s3) = (&init.Com.co\_s3 \bowtie vs3 \rightarrow SKIP \fatsemi starts.Com.co\_s3 \rightarrow \\ &(SKIP \triangle (\mathrm{I}(600, vs3).msg.co\_s3?x : \{cancel\} \rightarrow SKIP \fatsemi init.Com.co\_s4 \rightarrow SKIP) \Box \\ &(msg.co\_s3!x : \{in, last\} \rightarrow SKIP \fatsemi msg.co\_s3.out \rightarrow SKIP \fatsemi \\ &\mathrm{I}(600, vs3).init.Com.agate.1 \rightarrow SKIP) \fatsemi P(co\_s3))) \Box fin.2 \rightarrow SKIP \end{aligned}$$

$$P(end.2) = init.Com.end.2 \rightarrow SKIP \fatsemi fin.2 \rightarrow SKIP$$

Finally, the collaboration between the participants Customer and Company is the parallel composition of processes Cus and Com, as it is denoted by the PSS CSP+T process term, which conforms the BPTM of the Product/Service Sell BP to be verified.

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

4.2 Properties Definition

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), we must define the properties that each participant must fulfil, which show the execution sequence of BPMN modelling entities expected when they execute the partial processes of whom each 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 are defined below.

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

[1] Here, duration times are expressed in seconds, according to the function sec defined in [2]

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

Using the procedure described in [11], we obtained the processes $T(\phi_{Cus})$ and $T(\phi_{Com})$, which are the operational interpretation CCTL formulas previously specified. These process terms describe the expected behaviour for the processes Cus and Com that conform the BPTM, according to the CSP+T process calculus.

4.3 Verifying the Collaboration

According to our approach, to perform the verification of the BPTM we must verify first that the processes Cus and Com fulfil the properties specified in section 4.2. Then, according to the semantic domain to which CSP calculus, it can be checked that the following refining assertions are fulfilled:

$$T(\phi_{Cus}) \sqsubseteq_T Cus,\ T(\phi_{Com}) \sqsubseteq_T Com,\ T(\phi_{Cus}) \sqsubseteq_F Cus,\ T(\phi_{Com}) \sqsubseteq_F Com \tag{6}$$

To verify the above assertions, we are going to work according to the semantic model of CSP without temporal operators, since, according to the *timewise refinement*, 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 use the process terms CSP $UT(\phi_{Com})$ and $UT(\phi_{Cus})$, which correspond to the expected untimed behaviour of untimed processes $UT(Com)$ and $UT(Cus)$, respectively.As can be observed in the FDR2 screenshot in Fig. 3, 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). Thus, we obtained 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. Thus, the assertions in (6) are true.

According to assertion (5) (see section 3), to prove the correctness of the BPTM of the *Product/Service Sell* BP w.r.t. its expected behaviour, it must be demonstrated that:

$$PSS \vDash \phi_{PSS} \ \Leftrightarrow\ (Cus|[\alpha Cus \| \alpha Com]|Com)\backslash\{|\ msg\ |\} \ \vDash\ \phi_{Cus} \wedge \phi_{Com}\ .$$

We have previously verified with FDR2 that:

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

We must determine whether the Cus and Com local BPs are "composable". Thus, we must verify that it fulfills the following two 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:

$$\Sigma_{Cus} \cap \Sigma_{Com} = \emptyset \tag{7}$$
$$\Sigma_{Cus} = \{msg.cu\_s1.out, msg.cu\_s2.out, msg.cancel.out, msg.cu\_s5.out, msg.cu\_s6.out\}$$
$$\Sigma_{Com} = \{msg.co\_s1.out, msg.co\_s2.out, msg.co\_s3.out, msg.co\_s3.can, msg.co\_s8.out\}$$

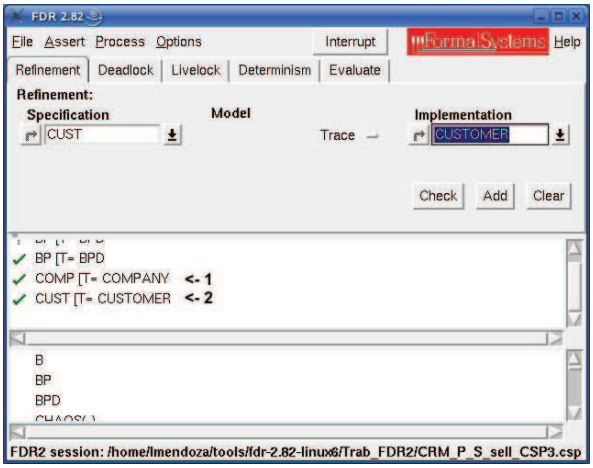


Fig. 3. FDR2 screenshot.

$$\Omega_{Cus} \cap \Omega Com = \emptyset \tag{8}$$
$$\Omega_{Cus} = \{msg.cu\_s1.in, msg.cu\_s1.last, msg.cu\_s2.in, msg.cu\_s2.last, msg.cancel.can, msg.cu\_s5.in, msg.cu\_s5.last, msg.cu\_s6.in, msg.cu\_s6.last\}$$
$$\Omega_{Com} = \{msg.co\_s1.in, msg.co\_s1.last, msg.co\_s2.in, msg.co\_s2.last, msg.co\_s3.in, msg.co\_s3.last, msg.co\_s8.in, msg.co\_s8.lastmsg.co\_s8.last\}$$

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

$$\mathcal{L}(Cus) \cap \mathcal{L}(Com) = \emptyset \tag{9}$$
$$\mathcal{L}(Cus) = \{start.1, cu\_s1, cu\_s2, cu\_s3, cu\_s4, cu\_s5, cu\_s6, xgate.1, end.1, abort.1\}$$
$$\mathcal{L}(Com) = \{start.2, co\_s1, co\_s2, co\_s21, co\_s3, co\_s4, co\_s5, co\_s6, co\_s7, co\_s8, agate.1, agate.2, end.2, abort.2\}$$

Having verified that the assertions (7), (8), and (9), are true, we conclude that Cus and Com are "composable". By Theorem 1 (see section 3), we have:

$$(Cus|[\alpha Cus \| \alpha Com]|Com)\backslash\{|\ msg\ |\} \models \phi_{Cus} \wedge \phi_{Com}$$

and because

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

we have

$$PSS \models \phi_{PSS}$$

Finally, we have obtained the verification of a BPTM corresponding to the *Product/Service Sell* BP from their verified local BP, Customer and Company.

5 Conclusions

In this paper we have presented and validated FCVA for compositional software verification from independently verified individual components and its instantiation to specify and verify the BPTM derived from BPs supported by an EIS. The local BPs are

modelled as CSP+T process terms, since it supports syntactical composition of process terms by the concurrent composition operator. Also a timed semantics of BPMN defined in terms of CSP+T formal specification language is presented to complement the FVCA, which allows us to detail the response times of activities and tasks, temporal constraints referring to task communication and collaboration, and the valid time span to capture exception flows, according to the expected behaviour of BPs. We have shown the value and practicality of our approach by means of its application to a real–life example in the field of CRM with timed collaboration requirements. Thus, the complete BPTM, derived 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. MC was used by passing the CSP+T terms through FDR2 to prove the correctness of global BPs.

Future and ongoing work will focus on the application of FCVA and the timed semantics of BPMN proposed to BPTM verification case studies; our future work will consist of doing in–depth research on the verification of these specifications, and to obtain automatic tool support for BPM by using state–of–the–art verification tools.

References

1. OMG: Business Process Modeling Notation – version 1.2. Object Management Group, Massachusetts, USA (2009)
2. Wong, P., Gibbons, J. In: A Process Semantics for BPMN, LNCS 5256: Proc. 10th Int. Conf. on Formal Engineering Methods ICFEM. Springer–Verlag, Berlin (2008) 355–374
3. Capel, M., Mendoza, L. In: Automatic Compositional Verification of Business Processes, LNBIP 24: Enterprise Information Systems. Springer Berlin, Heidelberg, Germany (2009) 479–490
4. Žic, J.: Time–constrained buffer specifications in CSP+T and Timed CSP. ACM Transaction on Programming Languages and Systems 16 (1994) 1661–1674
5. Bérard, B., Bidoit, M., Finkel, A., Laroussinie, F., Petit, A., Petrucci, L., Schnoebelen, P., McKenzie, P.: Systems and software verification: model-checking tech. and tools. (1999)
6. Morimoto, S. In: A Survey of Formal Verification for Business Process Modeling, LNCS 5102: Proc. 8th International Conference on Computational Science (ICCS 2008). Springer–Verlag, Berlin (2005) 514–522
7. Demri, S., Sattler, U.: Automata-theoretic decision procedures for information logics. Fundam. Inf. 53 (2002)
8. Mendoza, L., Capel, M.: Algorithm proposal to automata generation from CCTL formulas. Technical report, University of Granada (2008)
9. Mendoza, L., Capel, M., Pérez, M.: Compositional verification of business processes modelled with BPMN. In: Proc. 12th Int. Conf. on Enterprise Information Systems (ICEIS 2010), Setúbal, Portugal, INSTICC Press (2010) to appear
10. Rüf, J., Kropf, T.: 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. (1997)
11. Mendoza, L., Capel, M.: Procedure proposal to automata generation from CSP+T process terms. Technical report, University of Granada (2009)
12. Formal Systems (Europe) Ltd: Failures–Divergence Refinement – FDR2 User Manual. Formal Systems (Europe) Ltd, Oxford (2005)