When Can We Trust a Third Party?
A Soundness Perspective

Kees M. van Hee, Natalia Sidorova, and Jan Martijn van der Werf

Department of Mathematics and Computer Science
Technische Universiteit Eindhoven
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
{ k.m.v.hee, n.sidorova, j.m.e.m.v.d.werf }@tue.nl

Abstract. Organizations often do not want to reveal the way a product is created or a service is delivered. As a consequence, if two organizations want to cooperate, they contact a trusted third party. Each specifies how it wants to communicate with the other party. The trusted third party then needs to assure that the two organizations cooperate correctly. In this paper, we study requirements on trusted third parties to ensure correct cooperation between the different organizations.

1 Introduction

Organizations need to anticipate on the increasing dynamicity and complexity of business markets. Therefore, organizations focus more and more on their core activities. As a result, organizations need to cooperate in large networks. The organizations in the network have as common goal the delivery of their services. Such a network is called a *virtual enterprise* [11].

Communication between the organizations is asynchronous by nature: an organization sends some data, like an inquiry, to some other organization, and eventually the latter organization sends a response. Therefore, we use Petri nets to model organizations using *components*. Components can be composed into networks of components. Such a network is again a component. A component has an initial state and a desired final state. We say that a component, or a network of components communicates correctly if it is *sound*, i.e., if in all its reachable states the component is always able to eventually reach the desired final state.

Trust is an important property in a network of cooperating organizations: organizations share business knowledge and intellectual property with other organizations within the network in order to organize the component network properly and achieve desired goals. At the same time, organizations often want to keep some intellectual property within their organization and avoid sharing it for clear reasons. A common approach used in non-virtual life is the use of trusted third parties. It becomes nowadays also quite common in the virtual world. In this paper, we consider the use of a *third party*, also called a *notary*, that is trusted by all the organizations in the network. By using a notary, each

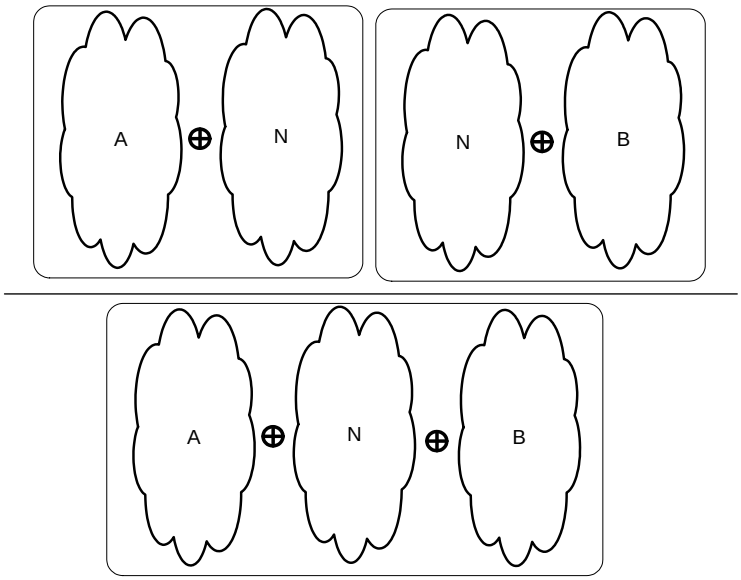


Fig. 1. If the notary N communicates correctly with A and B individually, we want to conclude correctness of the network of A, B and N

.

of the organizations explains to the notary the way it wants to conduct business, and the notary will assure that the organizations can do business together. This requires the notary to ensure that it communicates correctly with each of the organizations, i.e., that the notary with each of the individual organizations can reach the common goals, and secondly, that the complete network with all the organizations together can reach its common goals. If this is the case, we call the notary *trusted*.

In this paper we limit ourselves to the cooperation between two organizations using a notary. Rather than to use verification to check whether the communication between the notary and the two organizations is correct, we search for conditions such that if the communication between the notary and each of the individual organizations is correct, we can automatically conclude that the communication between the three parties is correct, as depicted in Fig. 1.

This paper is structured as follows. Sec. 2 introduces the basic notions needed throughout the paper. Sec. 3 explains the concept of components and their composition. In Sec. 4 we study the conditions under which the notary is guaranteed to ensure soundness of the composition of the three parties. Sec. 5 concludes the paper.

2 Preliminaries

Let S be a set. The powerset of S is denoted by $\mathcal{P}(S) = \{S' \mid S' \subseteq S\}$. We use $|S|$ for the number of elements in S. Two sets S and T are *disjoint* if $S \cap T = \emptyset$.

A *bag* m over S is a function $m : S \to \mathbb{N}$, where $\mathbb{N} = \{0, 1, \ldots\}$ denotes the set of natural numbers. We denote e.g. the bag m with an element a occurring once, b occurring three times and c occurring twice by $m = [a, b^3, c^2]$. The set of all bags over S is denoted by $\mathbb{N}^S$. Sets can be seen as a special kind of bag where all elements occur only once; we interpret sets in this way whenever we use them in operations on bags. We use $+$ and $-$ for the sum and difference of two bags, and $=, <, >, \leq, \geq$ for the comparison of two bags, which are defined in a standard way. The projection of a bag $m \in \mathbb{N}^S$ on some set U is a bag defined by $m_{|U}(s) = m(s)$ if $s \in U$ and $m_{|U}(s) = 0$ otherwise.

A *sequence* over S of length $n \in \mathbb{N}$ is a function $\sigma : \{1, \ldots, n\} \to S$. If $n > 0$ and $\sigma(i) = a_i$ for $i \in \{1, \ldots, n\}$, we write $\sigma = \langle a_1, \ldots, a_n \rangle$. The length of a sequence is denoted by $|\sigma|$. The sequence of length 0 is called the *empty sequence*, and is denoted by ϵ. The set of all finite sequences over S is denoted by S^*. We write $a \in \sigma$ if $\sigma(i) = a$ for some $1 \leq i \leq |\sigma|$. *Concatenation* of two sequences $\nu, \gamma \in S^*$, denoted by $\sigma = \nu; \gamma$, is a sequence defined by $\sigma : \{1, \ldots, |\nu| + |\gamma|\} \to S$, such that $\sigma(i) = \nu(i)$ for $1 \leq i \leq |\nu|$, and $\sigma(i) = \gamma(i - |\nu|)$ for $|\nu| + 1 \leq i \leq |\nu| + |\gamma|$. We inductively define the projection of $\sigma \in S^*$ on some set U by $a; \sigma'_{|U} = \langle a \rangle; \sigma'_{|U}$ if $a \in U$ and $a; \sigma'_{|U} = \sigma'_{|U}$ otherwise.

Definition 1 (Petri net [13]). *A* Petri net *N is a tuple (P, T, F) where (1) P and T are two disjoint sets of* places *and* transitions *respectively, we call an element of the set $(P \cup T)$ a* node *of N; and (2) $F \subseteq (P \times T) \cup (T \times P)$ is the* flow relation*. An element of F is called an* arc*.*

Let $N = (P, T, F)$ be a Petri net. Given a node $n \in (P \cup T)$, we define its preset by ${}^\bullet_N n = \{x \mid (x, n) \in F\}$, and its postset by $n^\bullet_N = \{x \mid (n, x) \in F\}$. We omit the subscript if the context is clear.

Let $N = (P, T, F)$ be a Petri net. A path from a node $n \in P \cup T$ to a node $m \in P \cup T$ is a sequence $\pi \in (P \cup T)^*$ such that $(\pi(i), \pi(i+1)) \in F$ for all $1 \leq i < n$. The set of all paths from n to m is denoted by $\Pi(n, m)$. A path is called *cyclic* if there exists a path π of length $n > 0$ such that $\pi(1) = \pi(n)$. If N has a cyclic path, the net is called *cyclic*. If no such cycle exists, it is called *acyclic*.

To describe the semantics of a Petri net, we use *markings*. A *marking* of N is a bag $m \in \mathbb{N}^P$, where $m(p)$ denotes the number of *tokens* in place $p \in P$. If $m(p) > 0$, place p is called *marked* in marking m. A Petri net N with a marking m is written as (N, m) and is called a *marked Petri net*.

Given a marked Petri net (N, m) with $N = (P, T, F)$, a transition $t \in T$ is *enabled*, denoted by $(N : m \xrightarrow{t})$, if ${}^\bullet t \leq m$. If a transition is enabled in (N, m), it can *fire*. A transition firing, denoted by $(N : m \xrightarrow{t} m')$, results in a new marking $m' = m - {}^\bullet t + t^\bullet$. We lift the firing to sequences of transitions in the standard way. A sequence $\sigma \in T^*$ of length n is a *firing sequence* from m_0 to

m_n, if there exist markings $m_i, m_{i+1} \in \mathbb{N}^P$ such that $(N : m_i \xrightarrow{\sigma(i)} m_{i+1})$ for all $1 \leq i < |\sigma|$. The set of reachable markings from a given marking m is denoted as $\mathcal{R}(N, m) = \{m' \mid \exists \sigma \in T^* : (N : m \xrightarrow{\sigma} m')\}$. We lift the set of reachable markings from a single marking to a set of markings in a standard way, i.e., given a set $M \subseteq \mathbb{N}^P$, $\mathcal{R}(N, M) = \bigcup_{m \in M} \mathcal{R}(N, m)$.

Given a marked Petri net (N, m_0) with $N = (P, T, F)$, a place $p \in P$ is called *k-bounded* for some $k \in \mathbb{N}$ if $m(p) \leq k$ for all markings $m \in \mathcal{R}(N, m_0)$. If all places are k-bounded, we call (N, m_0) k-bounded. A transition $t \in T$ is called *live* if for all markings $m \in \mathcal{R}(N, m_0)$ there exist a firing sequence $\sigma \in T^*$ and a marking $m' \in \mathcal{R}(N, m)$ such that $(N : m \xrightarrow{\sigma} m' \xrightarrow{t})$. If all transitions of (N, m_0) are live, (N, m_0) is called live. A transition $t \in T$ is called *quasi-live* if there exists a marking $m \in \mathcal{R}(N, m_0)$ such that $(N : m \xrightarrow{t})$. If all transitions of (N, m_0) are quasi-live, the marked Petri net is called quasi-live. A marking $m \in \mathcal{R}(N, m_0)$ is called a *home marking* if $m \in \mathcal{R}(N, m')$ for all $m' \in \mathcal{R}(N, m_0)$. A reachable marking $m \in \mathcal{R}(N, m_0)$ is called a *deadlock* of (N, m_0) if there is no transition $t \in T$ with $(N : m \xrightarrow{t})$. Given a desired marking $f \in \mathcal{R}(N, m_0)$, a non-empty subset of markings $L \subseteq \mathcal{R}(N, m_0)$ is called a *live-lock* w.r.t f if $f \notin \mathcal{R}(N, L)$ and $L = \mathcal{R}(N, L)$, i.e., from L the desired marking is not reachable, and no other marking then a marking in L can be reached from L.

On Petri nets, we define two classes based on their structure: S-Nets, also called state machines, and workflow nets. A Petri net $N = (P, T, F)$ is a *S-net* if $|{}^\bullet t| \leq 1$ and $|t^\bullet| \leq 1$ for all transitions $t \in T$.

Definition 2 (Workflow net, closure). *Let $N = (P, T, F)$ be a Petri net. It is a* workflow net *(WFN) if there exist two places $i \in P$ and $f \in P$, called the* initial place *and* final place *respectively, such that ${}^\bullet i = f^\bullet = \emptyset$, and all nodes of N are on a path from i to f. Its* closure *is the net $N^* = (P, T \cup \{t^*\}, F \cup \{(t^*, i), (f, t^*)\}$, where $t^* \notin T$.*

A workflow net is called *sound* if (1) it is weakly terminating, i.e., it always has the option to reach the final marking in which only the final place is marked, (2) it is properly completing, i.e., if in a marking the final place is marked, it is the only place marked, and (3) all transitions have a function, i.e., for every transition a reachable marking exists that enables the transition. Note that we use the classical soundness definition [1, 2].

Definition 3 (Soundness). *A workflow net $N = (P, T, F)$ with initial place i and final place f is called* sound *if (1) $[f]$ is a home marking of $(N, [i])$, (2) for any reachable marking $m \in \mathcal{R}(N, [i])$, if $m \geq [f]$ then $m = [f]$, and (3) $(N, [i])$ is quasi live.*

A WFN $N = (P, T, F)$ with initial place i is sound if and only if the marked Petri net $(N^*, [i])$ is live and bounded [1].

If we give a tuple a name, we subscript the elements with the name of the tuple, e.g. for $N = (A, B, C)$ we refer to its elements by A_N, B_N, and C_N. If the context is clear, we omit the subscript.

3 Components and their Composition

In this paper, we use asynchronously communicating components [5,7]. We therefore model our components using Petri nets with interface places, called *open Petri nets* (OPNs) [10,14]. An OPN has two types of places: *internal places* for the inner control of the component, and *interface places* to communicate with its environment. An interface place is either an output place, i.e., it sends a message to the environment, or an input place, i.e., it requires a message from the environment. Further, a component has an initial and a final marking, defining the desired begin and end markings of the component.

Definition 4 (Open Petri net, skeleton, open workflow net [3]). *An* open Petri net *(OPN) is an 6-tuple* (P, I, O, T, F, i, f) *where*

- $((P \cup I \cup O, T, F), i)$ *is a marked Petri net;*
- P *is a set of* internal places*;*
- I *is a set of* input places*, and* $^\bullet I = \emptyset$*;*
- O *is a set of* output places*, and* $O^\bullet = \emptyset$*;*
- P*,* I *and* O *are pairwise disjoint;*
- $\forall t \in T : |(^\bullet t \cup t^\bullet) \cap (I \cup O)| \leq 1$*; and*
- $i \in \mathbb{N}^P$ *is the* initial marking*; and*
- $f \in \mathbb{N}^P$ *is the* final marking*.*

We call the set $I \cup O$ *the* interface places *of the OPN. Two OPNs* N *and* M *are called* disjoint *if* $(P_N \cup I_N \cup O_N \cup T_N) \cap (P_M \cup I_M \cup O_M \cup T_M) = \emptyset$*. An OPN* N *is called* closed *if* $I_N = O_N = \emptyset$*. We write* $\mathcal{R}(N, m)$ *for* $\mathcal{R}((P_N \cup I_N \cup O_N, T_N, F_N), m)$ *for* $m \in \mathbb{N}^{P_N \cup I_N \cup O_N}$*.*

The skeleton *of* N *is defined as the Petri net* $\mathcal{S}(N) = (P_N, T_N, F)$ *with* $F = F_N \cap ((P_N \times T_N) \cup (T_N \times P_N))$*. For nodes* $n \in (P_N \cup T_N)$*, we write* ${}^\circ_N n$ *and* t°_N *as a shorthand for* ${}^\bullet_{\mathcal{S}(N)} t$ *and* $t^\bullet_{\mathcal{S}(N)}$*, respectively.*

If $\mathcal{S}(N)$ *is a workflow net with initial place* s *and final place* o*,* $i = [s]$ *and* $f = [o]$*,* N *is called an* open workflow net*.*

OPNs are composed with each other to build networks of communicating components. As a network of components can be used as a component again, the result of the composition is a component too. We say two OPNs are *composable* if the only elements shared between the two OPNs are their interface places, such that input places of the one are output places of the other and vice versa. Composition is then defined as the union of the two OPNs.

Definition 5 (Composition of OPNs [3]). *Two OPNs* A *and* B *are* composable*, denoted by* $A \oplus B$*, if and only if* $(P_A \cup I_A \cup O_A \cup T_A) \cap (P_B \cup I_B \cup O_B \cup T_B) = (O_A \cap I_B) \cup (I_A \cap O_B)$*.*

If A *and* B *are composable, their* composition *results in an OPN* $A \oplus B = (P_A \cup P_B \cup H, (I_A \cup I_B) \setminus H, (O_A \cup O_B) \setminus H, T_A \cup T_B, F_A \cup F_B, i_A + i_B)$ *with* $H = (O_A \cap I_B) \cup (I_A \cap O_B)$*.*

Note that two disjoint OPNs are composable by definition. Two important properties of composition are commutativity and projection, as shown in [14].

Corollary 6 (Commutativity, projection property [14]). *Let A and B be two composable OPNs. Then $N = A \oplus B = B \oplus A$, and $(\mathcal{S}(A) : m_{|P_A} \xrightarrow{\sigma_{|T_A}} m'_{|P_A})$ for all firing sequences $\sigma \in T_n^*$ and markings $m, m' \in \mathbb{N}^{P_N}$ such that $(\mathcal{S}(N) : m \xrightarrow{\sigma} m')$.*

The composition operator allows to create arbitrary networks of communicating components. As long as the interface places match, it is allowed to compose the components. However, it does not guarantee that the components communicate correctly. Composition is thus a syntactic check whether components are able to communicate.

Components communicate correctly if all components in the network are able to reach their desired final marking, and no messages are pending in one of the interface places. Further, we do not want to have transitions that are unreachable in the composition. To express this property, we use the notion of *soundness* for components: a component is sound if, ignoring the communication with other components in the network, all components can reach their final marking, no tokens are left in the network, and for each transition in the network, a marking should be reachable in which the transition is enabled.

Definition 7 (Soundness). *An OPN N is* sound *if:*

1. $\forall m \in \mathcal{R}(\mathcal{S}(N), i_N) : f_N \in \mathcal{R}(\mathcal{S}(N), m)$ *(weak termination);*
2. $\forall m \in \mathcal{R}(\mathcal{S}(N), i_N) : m \geq f_N \implies m = f_N$ *(proper completion); and*
3. $\forall t \in T_N : \exists m \in \mathcal{R}(\mathcal{S}(N), i_N) : {}^\circ t \leq m$ *(quasi liveness).*

Note that this soundness definition is stronger than the soundness notion used in [3], where soundness has been defined as the combination of weak termination and proper completion.

A direct consequence of the projection property and soundness is that if in a composition between A, B and C, such that A and C are disjoint, and A and B are composable, as well as B and C, and B is in its final marking, then the other two components can reach their final marking as well.

Lemma 8. *Let A, B and C be three pairwise composable OPNs such that A and C are disjoint, and $A \oplus B$ and $B \oplus C$ are sound. Define $L = A \oplus B \oplus C$. Then $f_L \in \mathcal{R}(\mathcal{S}(L), m)$ for all markings $m \in \mathcal{R}(\mathcal{S}(L), i_L)$ such that $f_B \leq m$.*

Proof. Define $K = A \oplus B$. By the projection property, $(\mathcal{S}(K) : i_K \xrightarrow{\sigma_{|T_K}} m_{|P_K})$. Since $f_B \leq m$, and K is sound, there exists a firing sequence $\mu \in T_A^*$ such that $(\mathcal{S}(K) : m_{|P_K} \xrightarrow{\mu} f_K)$, and hence $(\mathcal{S}(L) : m \xrightarrow{\mu} m')$ for some $m' \in \mathbb{N}^{P_L}$ with $f_K \leq m'$. Applying the same argument for $B \oplus C$, there exists a firing sequence $\nu \in T_B^*$ such that $(\mathcal{S}(L) : m' \xrightarrow{\nu} f_L)$, which proves the statement. □

4 Soundness Using Trusted Third Parties

Organizations have to cooperate more and more in order to do their business. However, they often do not want to share the way they operate, for example to hide internal business knowledge or intellectual property. An often proposed solution is a third party that is trusted by all organizations within the network. This third party, the notary, needs to ensure that it knows how the organizations within the network want to operate. On the one hand the notary needs to ensure that it works correctly with each individual organization, and on the other hand, that the network of all organizations, including the notary, is correct.

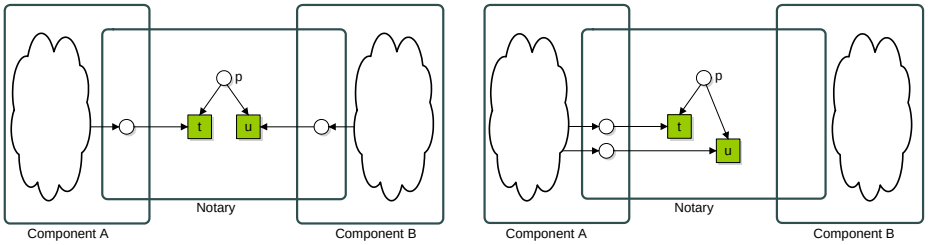
As the main purpose of a notary is to ensure correct behavior of the communication between the two organizations that want to cooperate, we model the notary by an OWN. The main actions of the notary are the sending and receiving of messages of the different components. Therefore, each transition that is communicating is labeled with the sending or receiving of a message, or as silent if the transition represents an internal step of the notary. We restrict the notary to state machines, i.e., each notary is sound by its structure [9].

Definition 9 (Notary). *Let A and B be two disjoint components. A* notary, *between A and B is an OWN N such that (1) A and N, as well as B and N are composable, (2) $\mathcal{S}(N)$ is an S-net, (3) each transition is connected to at most one interface place, i.e., $|({}^\bullet t \cup t^\bullet) \cap (I \cup O)| \leq 1$ for all $t \in T$, and, (4) each interface place to exactly one transition, i.e., $|{}^\bullet x \cup x^\bullet| = 1$ for all $x \in I \cup O$.*

As each transition is connected to at most one component, we introduce the communication function $C_N : T \to \{A, B, \tau\}$ *such that $C_N(t) = X$ if ${}^\bullet t \cap O_X \neq \emptyset$ for $X \in \{A, B\}$ and $C_N(t) = \tau$ otherwise.*

4.1 Acyclic Notaries

We first consider the case of an acyclic notary. If a notary is acyclic, then its set of possible firing sequences is finite. For acyclic notaries, soundness can be guaranteed, as shown in this section.



(a) Incorrect control (b) Correct control

Fig. 2. Conflicts in a notary

One source of possible erroneous behavior lies in the control of conflicts: if in a notary two transitions share a place in their presets, then the transitions should either be both controlled by the same component, or by the notary. Consider the examples of Fig. 2. Taking the composition $A \oplus N$ of Fig. 2(a), then transition u is always enabled if transition t is enabled, whereas in Fig. 2(b), component A controls the conflict in the composition of A and N. If the composition is sound, then the example of Fig. 2(a) is not possible, as shown in the next lemma.

Lemma 10 (Conflict control). *Let A and B be two components such that A and C are disjoint, and let N be an acyclic notary between A and B. If $A \oplus N$ and $N \oplus B$ are sound, then for all places $p \in P$ and transitions $t, u \in p^\bullet$ we have $C_N(t) = C_N(u)$.*

Proof. Let $p \in P$ and $t, u \in p^\bullet$. Suppose $C_N(t) \neq C_N(u)$. This implies that at least one of the transitions t and u is controlled by a component (otherwise we would have $C_N(t) = C_N(u) = \tau$). Without loss of generality, assume this transition to be t and component to be A, i.e., $C_N(t) = A$. Then there exists a place $q \in I_N \cap O_A$, with $q \in {}^\bullet t$.

Define $M = A \oplus N$. Since N is an S-net, ${}^\circ_M u \subset {}^\circ_M t$. Since M is sound, there exists a reachable marking $m \in \mathcal{R}(\mathcal{S}(M), i_M)$ with $(\mathcal{S}(M) : m \xrightarrow{t})$, and thus $m(q) > 0$. Note that transition u is also enabled in m. Hence, we can fire transition u and obtain a marking $m' \in \mathbb{N}^{P_M}$: $(\mathcal{S}(M) : m \xrightarrow{u} m')$, where $m' = m - {}^\circ_M u + u^\circ_M$ and $m'(q) = m(q)$ (since $q \notin {}^\circ_M u$).

As N is acyclic and t is the only transition consuming from q, the token from q will never be consumed by any sequence firing from m'. Thus, M is not sound, which is a contradiction. Hence, the lemma holds. □

The lemma shows that conflicts (choices) in the notary component are always controlled correctly, either by a single component or the notary itself. Consequently, if the composition of the components of A and B with N individually is sound, the composition of the three is sound as well, as proven in the next theorem.

Theorem 11. *Let A and B be two OPNs such that A and B are disjoint. Let N be an acyclic notary between A and B. If $A \oplus N$ and $N \oplus B$ are sound, then $A \oplus N \oplus B$ is sound.*

Proof. Define $M = A \oplus N \oplus B$. Suppose M is not sound. Then there exist a marking $m \in \mathcal{R}(\mathcal{S}(M), i_M)$ and a firing sequence $\sigma \in T^*_M$ such that $(\mathcal{S}(M) : i_M \xrightarrow{\sigma} m)$, $f_M \notin \mathcal{R}(\mathcal{S}(M), m)$. Moreover, since N is acyclic, there exists such an m, and, additionally $\gamma_{|T_N} = \emptyset$ for all firing sequences $\gamma \in T^*_M$ such that $(\mathcal{S}(M) : m \xrightarrow{\gamma})$. Now consider the following two possible cases:

1. Notary N is in its final place in this marking m, i.e. $m \geq f_N$, but A or B are not in their final markings;
2. Notary N is not in its final place in marking m, i.e. $m \ngeq f_N$.

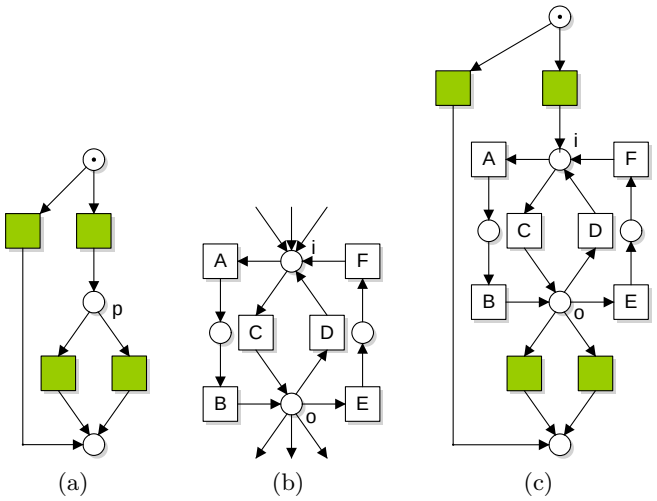


Fig. 3. An acyclic S-net (a), a single-entry-single-exit loop (b), and the refinement of place p in (a) with loop (b)

The first case contradicts Lm. 8. Consider the second case. No transitions of N will be enabled in $(\mathcal{S}(M), m)$, and N is not in its final marking. As $f_N \neq m_{|P_N}$ and $\mathcal{S}(N)$ is an S-net, there exist a place $p \in P_N$ of the notary such that $m(p) > 0$. $p^\bullet$ cannot contain any transition t with $C_N(t) = \tau$, since this transition would be enabled in m. Due to Lm. 10, all transitions t from $p^\bullet$ have the same value for $C_N(t)$. Without the loss of generality, we suppose it to be A. Since $A \oplus N$ is sound, there is a firing sequence $\sigma; t$ from marking $m_{|A \oplus N}$ in $A \oplus N$ such that $\sigma \in T_A^*$ and $t \in T_N$. This firing sequence is then also a firing sequence of M, but this contradicts the statement that no transition from T_N can fire starting from m in M.

Therefore, $A \oplus N \oplus B$ is sound. □

4.2 Simple Cyclic Notaries

Acyclic notaries ensure the correctness of the composition of two components if these components communicate correctly with the notary. Often, cyclic behavior between components is needed. For example, in order to agree on some quote, several cycles may be involved. In this section, we extend acyclic nets with single-entry-single-exit (SESE) loops.

A SESE loop consists of an entry place and an exit place, not being the same, and all nodes inside the loop are on a path from entry to exit or vice versa, on a path from exit to entry. Furthermore, we require each place in the loop to have exactly one transition in its preset and one in its postset, except for the entry and exit place. An example of a SESE loop is depicted in Fig. 3(b).

An S-net is a *Simple Cyclic S-net* (SCS-net) if all loops in the net are disjoint, i.e., each place and transition of the net belongs to at most one loop. In simple

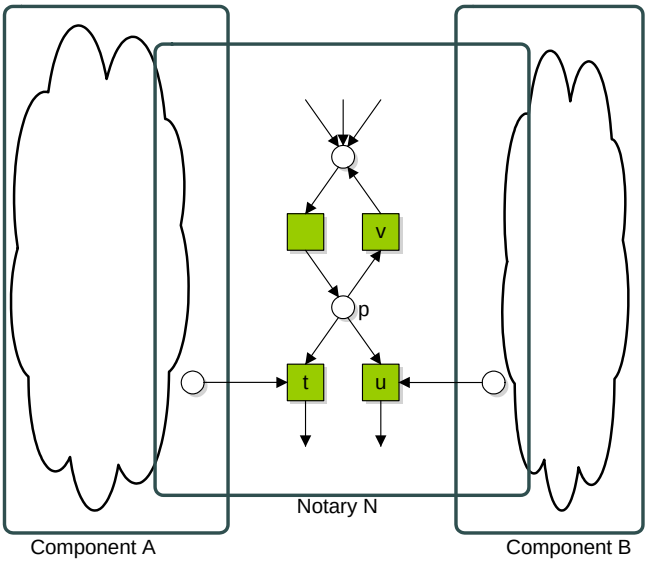


Fig. 4. Controlling conflicts in a simple-cyclic notary

cyclic nets, each loop can be replaced by a place, which results in an acyclic S-net (see Fig. 3).

Definition 12 (SESE Loop, simple-cyclic S-net). *Let (P,T,F) be an S-net. A set $L \subseteq P \cup T$ is a called a* single-entry-single-exit loop *(SESE loop) with* entry place $e \in L \cap P$ *and* exit place $o \in L \cap P$ *if all nodes $n \in L$ are on a path from e to o or on a path from o to e, ${}^\bullet e \setminus L \neq \emptyset$, $e^\bullet \subseteq L$, $o^\bullet \setminus L \neq \emptyset$, and ${}^\bullet o \subseteq L$, and for all places $p \in L \cap P$, if $|{}^\bullet p| > 1$ then $p = e$ and if $|p^\bullet| > 1$ then $p = o$.*

Let (P,T,F) be an S-net. It is called a simple-cyclic S-net *(SCS-net) if all loops are SESE loops and pairwise disjoint, i.e., for all loops $L_1, L_2 \subseteq P \cup T$ if $L_1 \cap L_2 \neq \emptyset$ then $L_1 = L_2$.*

Note that in the definition of SCS-nets, we require each node to be in at most one loop. By the definition of the SESE loop, we have that if a node contains multiple elements in its preset or postset, it is either the entry or the exit of the loop. As a consequence, all SESE loops are simple: there is one path from entry to exit and one path from exit to entry. The basis of an SCS-net is an acyclic S-net. Consequently, each loops will be entered at most once.

Whereas in an acyclic notary every conflict is controlled by a single component, this is not the case in the simple-cycled case, as shown in Fig. 4. Choices still need to be controlled by a single component, but silent loops, i.e., no transition in the loop is connected with an interface places, are allowed.

Similarly to the acyclic case, if the skeleton of a notary is a simple-cyclic S-net, soundness of the three parties is assured if the notary composed with each of the organizations individually is sound. Whereas in the acyclic case every

conflict is controlled by a single component, if the notary is cyclic, the moment of control can be postponed. As a consequence, we need to also consider the possibility of live-locks in which the notary is involved.

Theorem 13. *Let A and B be two disjoint OPNs and let N be an simple-cyclic notary such that $A \oplus N$ and $N \oplus B$ are sound. Then $A \oplus N \oplus B$ is sound.*

Proof. Define $K = A \oplus N$, $L = N \oplus B$ and $M = A \oplus N \oplus B$. Suppose M is not sound. Then there exists a marking $m \in \mathcal{R}(\mathcal{S}(M), i_M)$ such that $f_M \notin \mathcal{R}(\mathcal{S}(M), m)$.

1. Notary N is in its final marking, but A or B cannot reach its final marking, or tokens are left in the interface places;
2. Notary N reaches a deadlock different from the final marking;
3. Notary N reaches a live lock with respect to the final marking;

The first case contradicts with Lm. 8.

Next, consider the second case. Suppose a marking $m \in \mathcal{R}(\mathcal{S}(M), i_M)$ not being the final marking exists that is a deadlock. Then, a place $p \in P_N$ exists with $p \neq f_N$, $m(p) = 1$ and ${}^\bullet t \not\leq m$ for all transitions $t \in p^\bullet$. Let $t \in p^\bullet$. Then $C_N(t) \neq \tau$. If $C_N(t) = C_N(u)$ for all $t, u \in p^\bullet$, then either $A \oplus N$ or $N \oplus B$ would not be sound. Hence, transitions $t, u \in p^\bullet$ exist such that $C_N(t) \neq C_N(u)$. Without loss of generality assume $C_N(t) = A$ and $C_N(u) = B$.

Since $\mathcal{S}(N)$ is an SCS-net, place p is either an entry or exit point of a loop, or outside a loop. Suppose place p is not the exit of a loop. Then in K, transition u is enabled in $m_{|P_{A \oplus N}}$. Since K is sound, it must be possible to fire transition t. Thus, a marking $m' \in \mathcal{R}(\mathcal{S}(K), i_L)$ exists such that $(\mathcal{S}(K) : m' \xrightarrow{t})$. As transition u is also enabled in m', u has to be in a loop, since otherwise K the token from the places $q \in {}^\bullet t \cap O_A$ would never be consumed. Similarly, transition t has to be in a loop, since otherwise the token from the places $q \in {}^\bullet u \cap O_B$ would never be consumed. Hence, a deadlock cannot occur.

Last, consider the case in which N reaches a live-lock, i.e, it entered a loop L with entry i and exit o such that it cannot leave the loop. Hence, $C_N(t) \neq \tau$ for all transitions $t \in o^\bullet \setminus L$. If $C_N(t) = C_N(u)$ for all $t, u \in p^\bullet$, then either $A \oplus N$ or $N \oplus B$ would not be sound. Hence, transitions $t, u \in p^\bullet$ exist such that $C_N(t) \neq C_N(u)$. Without loss of generality assume $C_N(t) = A$ and $C_N(u) = B$. Again due to liveness of K and L, this is not possible.

As all cases lead to a contradiction, $A \oplus N \oplus B$ is sound. □

5 Conclusions

We studied in this paper the problem of ensuring correctness of networks of co-operating organizations. By introducing a trusted third party, called a notary, organizations do not need to share their knowledge with the other organizations within the network. The notary needs to ensure that firstly it works correctly with each of the organizations individually, and secondly that all organizations

in the network, including the notary itself, work correctly together. In this paper, we showed for two organizations and a notary that if the notary is an acyclic state machine, or it contains only single-entry-single-exit (SESE) loops, then the notary ensures soundness if it is sound with each of the organizations individually.

In literature, different approaches exist. For example, in the approach of [4], the authors use contracts, implemented as public views. Organizations then need to implement their public views as a private view. If each of the private views agrees on the public view, the network is guaranteed to be correct. In [8], an interactive Petri net is designed, modeling the communication between different organizations.

The disadvantage of these approaches is that each of the organizations need to implement a private view, whereas often organizations already have existing components. In these approaches, the organizations have to re-engineer the existing components, and prove that these re-engineered components adhere to the views defined in the contract using e.g. accordance [12] or contract theory [6]. In the approach described in this paper, organizations can reuse existing components, as the approach requires an organization to cooperate correctly with the notary.

The setting in this paper is comparable with the more general setting of decentralized controllability [15], which is shown to be undecidable [16]. We limited ourselves to two organizations with a notary which is either acyclic or only contains SESE loops. Although these requirements are quite strong, they are needed to ensure soundness. Future work will be to search for more liberal notaries and to extend the results to service trees [3]. As shown in [14], soundness is not compositional, and additional requirements are needed.

References

1. W.M.P. van der Aalst. Verification of Workflow Nets. In *Application and Theory of Petri Nets 1997*, volume 1248 of *Lecture Notes in Computer Science*, pages 407 – 426. Springer-Verlag, Berlin, 1997.
2. W.M.P. van der Aalst, K.M. van Hee, A.H.M. ter Hofstede, N. Sidorova, H.M.W. Verbeek, M. Voorhoeve, and M.T. Wynn. Soundness of workflow nets: classification, decidability, and analysis. *Formal Aspects of Computing*, pages 1–31, 2010.
3. W.M.P. van der Aalst, K.M. van Hee, P. Massuthe, N. Sidorova, and J.M.E.M. van der Werf. Compositional Service Trees. In *Applications and Theory of Petri Nets 2009*, volume 5606 of *Lecture Notes in Computer Science*, pages 283 – 302. Springer-Verlag, Berlin, 2009.
4. W.M.P. van der Aalst, N. Lohmann, P. Massuthe, C. Stahl, and K. Wolf. Multiparty Contracts: Agreeing and Implementing Interorganizational Processes. *The Computer Journal*, 53(1):90–106, 2010.
5. G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web Services – Concepts, Architectures and Applications*. Springer-Verlag, Heidelberg, 2004.
6. S.S. Bauer, A. David, R. Hennicker, K.G. Larsen, A. Legay, U. Nyman, and A. Wasowski. Moving from specifications to contracts in component-based design. In *FASE 2012*, pages 43–58, 2012.

7. M. Beisiegel, K. Khand, A. Karmarkar, S. Patil, and M. Rowley. Service Component Architecture Assembly Model Specification Version 1.1, 2010.
8. G. Decker and M. Weske. Local Enforceability in Interaction Petri Nets. In *Business Process Management*, volume 4714 of *Lecture Notes in Computer Science*, pages 305–319. Springer-Verlag, Berlin, 2007.
9. K.M. van Hee, N. Sidorova, and M. Voorhoeve. Soundness and Separability of Workflow Nets in the Stepwise Refinement Approach. In *Application and Theory of Petri Nets 2003*, volume 2679 of *Lecture Notes in Computer Science*, pages 335 – 354. Springer-Verlag, Berlin, 2003.
10. P. Massuthe, W. Reisig, and K. Schmidt. An Operating Guideline Approach to the SOA. *Annals of Mathematics, Computing & Teleinformatics*, 1(3):35–43, 2005.
11. N. Mehandjiev and P.W.P.J. Grefen, editors. *Dynamic Business Process Formation for Instant Virtual Enterprises*. Springer-Verlag, Berlin, 2010.
12. A.j. Mooij, C. Stahl, and M. Voorhoeve. Relating fair testing and accordance for service replaceability. *J. Log. Algebr. Program.*, 79(3-5):233–244, 2010.
13. W. Reisig. *Petri Nets: An Introduction*, volume 4 of *Monographs in Theoretical Computer Science: An EATCS Series*. Springer-Verlag, Berlin, 1985.
14. J.M.E.M. van der Werf. *Compositional design and verification of component-based information systems*. PhD thesis, Technische Universiteit Eindhoven, 2011.
15. K. Wolf. Does My Service Have Partners? In *Transactions on Petri Nets and Other Models of Concurrency II*, Lecture Notes in Computer Science, pages 152 – 171. Springer-Verlag, Berlin, 2009.
16. K. Wolf. Decidability Issues for Decentralized Controllability of Open Nets. In *17th German Workshop on Algorithms and Tools for Petri Nets*, volume 643, pages 124–129. CEUR-WS, 2010.