

# Controllability of Open Workflow Nets

Karsten Schmidt

Humboldt-Universität zu Berlin, Institut für Informatik. Unter den Linden 6,  
D-10099 Berlin

**Abstract.** An open workflow net is basically a workflow net extended with a message passing interface. Open workflow nets are adequate models for services or parts of inter-organizational business processes. We investigate the problem of *controllability*, a natural counterpart of *soundness* in classical workflow nets (as studied by van der Aalst). We distinguish centralized, distributed, and local controllability and provide solutions to all problems.

## 1 Introduction

Throughout this paper, we study acyclic open workflow nets (*oWFN*), an extension of acyclic workflow Petri nets [Aal98]. The extension mainly consists of a message passing interface. While workflow nets [Aal98] proved useful for modelling and analyzing business processes, an oWFN is suitable for modelling a service, or a part of an inter-organizational workflow [Mar03]. oWFN can be automatically generated [Sta04,HSS05] from emerging service specification languages such as BPEL4WS [CGK<sup>+</sup>03].

Workflow nets have a distinguished initial as well as a final state. Possibility to reach the end state from every state reachable from the initial state has been established as an important quality criterion for workflow nets. A workflow net holding this property is called *weak sound* and is called *sound* if additionally there are no dead transitions in the net. In [Mar03], the concept of soundness was adapted to oWFN (called *workflow modules* there), coining the concept of *usability*: an oWFN is usable if there *exists* an environment such that the composition of the oWFN with the environment yields a weakly sound net (informally: it is *possible* to communicate properly with the oWFN, a condition also proposed in [AH01]). This way, usability turns out to actually be a controller synthesis problem where the environment is a supervisory controller whose goal is to preserve reachability of the final state. In this paper, we return to a control-theoretic vocabulary and talk about *controllability* rather than usability. This way, we would like to shift attention from a soundness property of an oWFN to the actual controllers. Synthesized controllers may turn out to be useful as a description for how potential communication partners are required to communicate properly with the given process. The can, in fact be extended to form *operating guidelines* for the given oWFN [MS05].

The classical approach to supervisory control of discrete event systems [RW87] models plant and controller as automata and uses a concept of transition sharing for modelling the interaction between plant and controller. In our setting,

interaction is fully asynchronous, by message passing. This seems to be appropriate for business process interaction. Furthermore, internal behavior of the plant is fully unobservable by the environment. Though it is in principle possible to map our setting into the terminology of [RW87], the resulting models become rather involved, and it is almost impossible to exploit the specific nature of the given control problem. Especially the asynchronous message passing mechanism requires a tedious construction while it can be modelled easily in our Petri net approach.

In this paper, we first introduce our concepts for modelling plant, controller, and their interplay. Then we study centralized control. We show the existence of a unique most permissive controller and present an algorithm to construct it. Next, we study decentralized control, showing that a most permissive controller does not exist, still presenting an algorithm to decide decentralized controllability. Finally, we introduce a novel problem called local controllability where the task is to construct a single part of a decentralized controller without knowledge about the other parts of the controller. We define a concept of *cooperative* controller and prove that *every* collection of cooperative controllers actually controls an oWFN. If there is a cooperative controller then there is a unique most permissive, too.

## 2 Plant and controller

We model plants as a special class of Petri nets.

**Definition 1 (Petri net).** *A Petri net  $N$  consists of two finite, disjoint sets  $P$  (places) and  $T$  (transitions), a flow relation  $F$  ( $F \subseteq (T \times P) \cup (P \times T)$ ), and an initial marking  $m_0$ . A marking is a mapping  $m : P \rightarrow \mathbf{N}$ .*

Places are depicted as circles, transitions as boxes, the flow relation as arrows, and markings as distributions of tokens on the places.

**Definition 2 (Behavior of Petri nets).** *Transition  $t$  is enabled in marking  $m$  if, for all places  $p$ ,  $[p, t] \in F$  implies  $m(p) > 0$ . Transition  $t$  may fire in marking  $m$  yielding a marking  $m'$  ( $m \xrightarrow{t} m'$ ) if  $t$  is enabled in  $m$ , and for all  $p$ ,  $m'(p) = m(p) + W([t, p]) - W([p, t])$  where  $W([x, y]) = 1$  if  $[x, y] \in F$ , and  $W([x, y]) = 0$ , else. With  $R_N(m)$ , we denote the set of markings that can be reached from  $m$  by firing any finite number of transitions.*

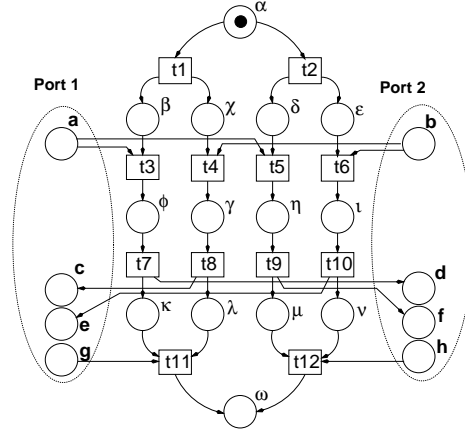
Open Workflow nets are a class of Petri nets. In this paper, we study only acyclic systems.

**Definition 3 (Open workflow net).** *A Petri net  $N$  is an open workflow net (oWFN)  $M$ , if*

- $P$  is the disjoint union of sets  $P_M$  (internal places),  $P_I$  (input channels), and  $P_O$  (output channels);
- $F \cap (P_O \times T) = \emptyset$ ,  $F \cap (T \times P_I) = \emptyset$ ;
- $F$  does not contain cycles (the transitive closure of  $F$  is irreflexive);

– there is a distinguished final marking  $m_f$ .

Fig. 1 shows an oWFN. In all our examples, there is a start place  $\alpha$  which has initially one token, and a place  $\omega$  which is supposed to be the only place marked in  $m_f$ .



**Fig. 1.** An oWFN. Places with Greek names are internal, places  $a, b, g, h$  input channels, and  $c, d, e, f$  output channels. The boxes around the interface places represent an interface partition introduced for decentralized controllability.

One reason to use Petri nets as a modelling language is that channel places are a rather elegant way to model asynchronous message passing. Since the channel places are part of the oWFN, this reason is not valid for the controller. In this paper, we model controllers as classical automata. Of course, it would be possible to model controllers as Petri nets, too. However, the constructions proposed here are essentially based on the concept of state. It is in fact possible to construct a Petri net out of a state space, either brute force resulting in a Petri net called *state machine*, or more sophisticated, through applying *region theory* [BD98,NRT92]. Since these constructions are well-understood, we decided to concentrate on the core construction thus introducing controllers as automata.  $bags(Z)$  denotes the set of multisets over the set  $Z$ .

**Definition 4 (Controller).** Let  $M$  be an oWFN and  $I \subseteq (P_I \cup P_O)$ . A controller connected to  $I$  is an automaton with alphabet  $bags(I)$ , a set of states  $Q$ , a move relation  $\delta : Q \times bags(I) \rightarrow \wp(Q)$ , and an initial state  $q_0$ .

Fig. 2 shows two controllers. A move in the controller describes an interaction with the oWFN, coded in the alphabet. The multiset involved in the move stands for messages to be received or to be sent. Of course, only messages present on the output channels can be received. Definition 6 describes the interplay between an oWFN and a feasible set of controllers.

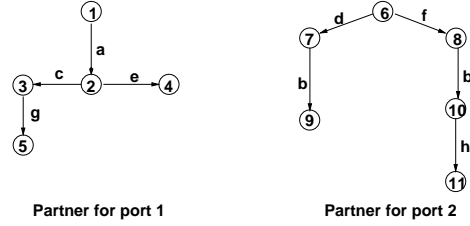


Fig. 2. A set of feasible controllers for the oWFN in Fig. 1.

**Definition 5 (Feasibility).** A set of controllers  $\{C_1, \dots, C_n\}$  is feasible for an oWFN  $M$  iff their alphabets are disjoint, and the union of their alphabets equals  $P_I \cup P_O$ .

The set of the two controllers in Fig. 2 is feasible for the oWFN in Fig. 1.

**Definition 6 (Composed system).** Let  $M$  be an oWFN and  $\{C_1, \dots, C_n\}$  a feasible set of controllers. Then the composed system is a transition system with  $R_N(m_0) \times Q_1 \times \dots \times Q_n$  as set of states, and edges

- from  $[m, q_1, \dots, q_n]$  to  $[m', q_1, \dots, q_n]$  if there is a transition  $t$  in  $M$  with  $m \xrightarrow{t} m'$ ;
- from  $[m, q_1, \dots, q_i, \dots, q_n]$  to  $[m', q_1, \dots, q'_i, \dots, q_n]$  if there is a multiset  $B$  such that in  $C_i$   $q'_i \in \delta_i(q_i, B)$ , for all  $p \in P_O$ ,  $m(p) \geq B(p)$  and  $m'(p) = m(p) - B(p)$ , for all  $p \in P_i$ ,  $m'(p) = m(p) + B(p)$ , and for all  $p \in P_M$ ,  $m'(p) = m(p)$ .

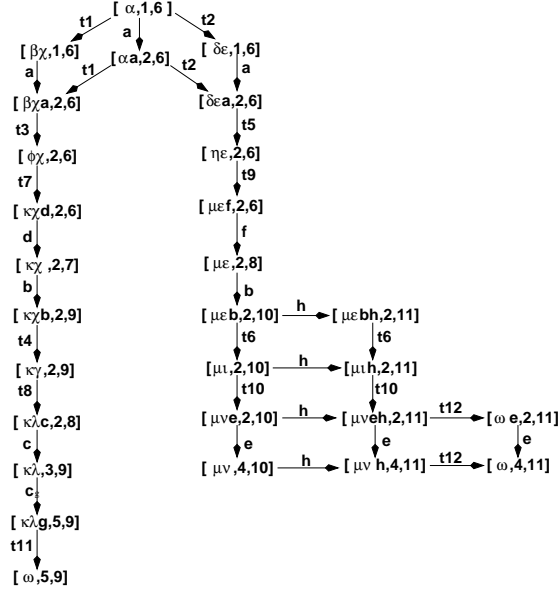
The initial state is  $[m_0, q_{0_1}, \dots, q_{0_n}]$ .

Fig. 3 depicts the system composed of the oWFN in Fig. 1 and the controllers in Fig. 2.

The goal of control is proper termination of the oWFN. This means that the oWFN is in marking  $m_f$ . In this marking, all channels places are empty.

**Definition 7 (Strategy).** Let  $M$  be an oWFN. A feasible set of controllers is a strategy for  $M$  if, in the composed system, from every state reachable from the initial state, a state is reachable whose first component is  $m_f$ .

We have defined controllers quite liberal (nondeterministic, with several messages processed in a step, with internal moves (empty multiset), possibly infinite). Actually, none of these features is necessary for controlling an oWFN. Furthermore, we may restrict to tree controllers where states uniquely identify the past behavior of the controller. Finally, we may observe that the number of messages to be reasonably exchanged between controller and oWFN (the number of tokens ever produced on channel places) is bounded, since the oWFN is acyclic. Such bound can actually be computed. However, we omit an algorithm and assume a bound  $l_M$  to be given. For an alphabet  $A$ , let  $A^*$  denote the set of finite words over  $A$ . Let  $\underline{a}$  denote the multiset containing  $a$  once, and no other element.



**Fig. 3.** System composed of the oWFN in Fig. 1 and the controllers in Fig. 2.

**Definition 8 (Tree controller).** A controller connected to  $I$  where  $Q \subseteq \{w \mid w \in I^*, \text{length}(w) \leq l_M\}$ ,  $Q$  contains with  $q$  all prefixes of  $q$ ,  $\delta(w, x) = \{wa\}$  if  $wa \in Q$  and there is an  $a$  with  $x = \underline{a}$ , and  $\delta(w, x) = \emptyset$ , else, and  $\lambda$  (the empty word) is the initial state, is called a tree controller.

Closure under prefixes is necessary and sufficient for reachability of all states from the initial state. The proposed restrictions simplify subsequent considerations but do not restrict generality:

**Proposition 1.** If  $\{C_1, \dots, C_n\}$  is a strategy for  $M$  and  $C_i$  is connected to  $I$  then there is a tree controller  $C'_i$  connected to  $I$  such that  $\{C_1, \dots, C'_i, \dots, C_n\}$  is a strategy for  $M$ .

This follows from the considerations about bounded communication and the fact that every automaton can be unrolled to a tree-like automaton that, in turn, is isomorphic to a tree controller.

An advantage of tree controllers is that they are completely determined by their set of states. In the sequel, we consider only tree controllers.

Tree controllers with unrestricted behavior are called *noise*.

**Definition 9 (Noise).** Let  $I \subseteq P_I \cup P_O$ . Then the tree controller with  $Q = \{w \mid w \in I^*, \text{length}(w) \leq l_M\}$  is called the noise for  $I$  and denoted  $\text{noise}(I)$ .

In a tree controller  $C$  connected to  $I$ , states encode the full record of past interaction with the oWFN  $M$ . We assume that the internal behavior of  $M$  is

fully unobservable for  $C$ , except the output channels in  $I \cap P_O$ . Nevertheless, knowing  $C$  and  $M$ , it is possible to deduce, in which markings  $M$  can possibly be while  $C$  is in a given state  $q$ . We formalize this knowledge that a state of  $C$  represents about the marking of  $M$  through a mapping  $K$ .

**Definition 10 (Knowledge of the controller).** *Let  $C$  be a controller connected to  $I$  and  $q$  a state of  $C$ . Let  $S$  be the set of states of the system composed of  $M$ ,  $C$  and  $\text{noise}((P_I \cup P_O) \setminus I)$ . Then the knowledge of  $C$  in state  $q$ ,  $K(q) := \{m \mid \exists q' : [m, q, q'] \in S\}$ .*

Complementing  $C$  with  $\text{noise}((P_I \cup P_O) \setminus I)$  makes this definition independent of other controllers. Fig. 5 shows values of  $K$  attached to a controller for the oWFN in Fig. 4.

As a last pre-requisite, we introduce the notion of *deadlock*.

**Definition 11.** *A marking of  $M$  where no transition of  $M$  is enabled, and which is not the end state of  $M$ , is called *deadlock* of  $M$ .*

Note, that deadlocks are considered only with respect to  $M$  in isolation. An environment may send tokens to  $M$  and this way resolve some of the deadlocks.

### 3 Centralized controllability

In this section, we study centralized control, i.e., we look for strategies that consist of a single controller, connected to  $P_I \cup P_O$ . For simpler notation, we identify controller  $C$  with the set  $\{C\}$ .

**Definition 12 (Centralized controllability).** *An oWFN  $M$  is centralized controllable if there exists a strategy connected to  $P_I \cup P_O$ .*

**Definition 13 (Most permissive strategy).** *A tree controller  $C$  with set of states  $Q$  is a most permissive strategy for  $M$  if it is a strategy and every tree controller that is strategy for  $M$  has a set of states included in  $Q$ .*

Inclusion of the sets of states means that every strategy can be seen as restriction of the behavior of the most permissive strategy.

Our approach to centralized controllability is based on the following characterization of centralized strategies.

**Theorem 1.** *A tree controller  $C$  connected to  $P_I \cup P_O$  is a strategy for oWFN  $M$  if and only if it has a nonempty set of states and, for all  $q \in Q$ , it holds: for every deadlock  $m \in K(q)$ , the system composed of  $M$  and  $C$  has a move starting in  $[m, q]$ .*

The move of the composed system required in the second item must actually be a move of  $C$  with some letter  $a$ , as  $m$  is a deadlock of  $M$ .

**Proof. Implication.** Let  $C$  be a strategy for  $M$  and  $q$  a state of  $C$ . If  $m \in K(q)$  then  $[m, q]$  is reachable in the composed system. If  $K(q)$  contains a deadlock  $m$  then, as  $m \neq m_f$ ,  $[m, q]$  must have a successor in the composed system.

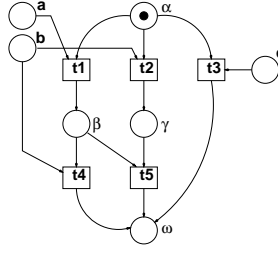


Fig. 4. A simple oWFN

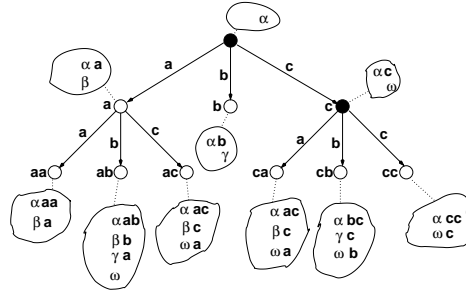


Fig. 5. Construction of a most permissive strategy for the oWFN in Fig. 4.

*Replication.* Let  $C$  be a tree controller satisfying the two stated condition. Since  $M$  is acyclic and  $C$  a tree controller, the composed system does not contain infinite paths. Thus, every sequence eventually leads to a state  $[m^*, q^*]$  without successors in the composed system.  $m^*$  cannot be a deadlock, as  $[m^*, q^*]$  does not have a successor. Thus  $m^*$  is the final marking  $m_f$ . q.e.d.

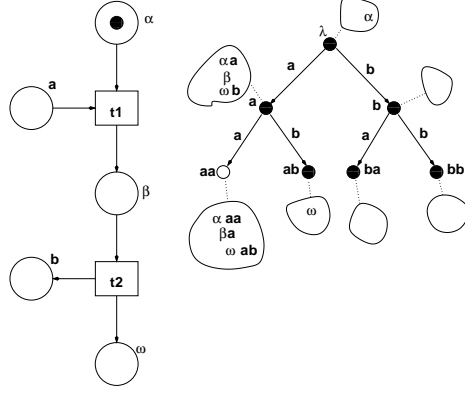
With this characterization, we can construct a most permissive centralized strategy for  $M$ . Starting with  $noise(P_I \cup P_O)$ , we repeatedly remove states where deadlocks have no successors. Removing  $q$  requires removing all  $qw$ , too, since those states would become unreachable from the initial state. In the algorithm, a controller move  $a$  is called *possible* in marking  $m$  of  $M$  if either  $a$  is an input channel, or  $a$  is an output channel and  $m(a) > 0$ .

### Algorithm 1

1.  $Q := \{w \mid w \in (P_I \cup P_O)^*, length(w) \leq l_M\}$ ;
2. WHILE  $\exists q : K(q)$  contains a deadlock  $m$  and, for all  $a$ ,  $a$  is not possible in  $m$  or  $qa \notin Q$  DO
  - (a)  $Q := Q \setminus \{qw \mid w \in (P_I \cup P_O)^*\}$ ;

Fig. 5 shows an example for the application of this algorithm. Starting with the noise automaton ( $K$ -values are attached), we first remove states  $aa$  (due to deadlock  $\beta a$ ),  $ab$  ( $\gamma a$ ),  $ac$  ( $\omega a$ ),  $b$  ( $\gamma$ ),  $ca$  ( $\omega a$ ),  $cb$  ( $\omega b$ ),  $cc$  ( $\omega c$ ). Since  $b$  is an unresolvable deadlock, we did not depict its successors. Then, in the second

iteration,  $a$  is removed since, after removal of state  $ab$ , there is no longer a move possible for the deadlock  $\beta$  contained in  $K(a)$ . It remains the set of states  $\{\lambda, c\}$  which is indeed the set of states of a strategy for the oWFN in Fig. 4.



**Fig. 6.** An oWFN with strange most permissive strategy (states represented by solid circles), shown as part of a noise automaton (all states)

Fig. 6 shows another example for the calculation of a most permissive strategy. There are several states  $q$  with  $K(q) = \emptyset$ . Since with  $K(q) = \emptyset$  the requirements of Thm. 1 are trivially satisfied, these states appear in the most permissive strategy though  $K(q) = \emptyset$  indicates that the controller can never enter these states when composed to the oWFN. Nevertheless it is important not to remove these states. The reason shall be revealed in Sec. 4.

**Theorem 2.** *Alg. 1 either returns the empty set of states, then  $M$  does not have strategies, or it returns a most permissive strategy for  $M$ .*

**Proof.** By Def. 8, every tree controller for  $M$  has a set of states included in  $\{w \mid w \in (P_I \cup P_O)^*, length(w) \leq l_M\}$ . None of the states removed by Alg. 1 can be contained in the set of states of any strategy, as it would violate Thm. 1. Thus, every strategy that is a tree controller has a set of states included in the one returned by Alg. 1. Consequently, if the empty set of states is returned, strategies for  $M$  do not exist. If a nonempty set is returned, it is the set of states of a strategy, as Thm. 1 is obviously satisfied, and hence a most permissive strategy. q.e.d.

**Corollary 1.** *If an oWFN is centralized controllable then there exists a most permissive strategy for  $M$ .*

## 4 Decentralized controllability

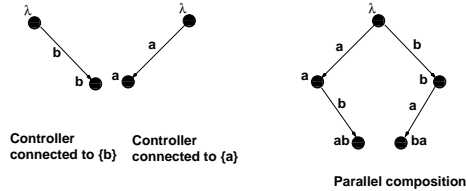
Throughout this and the next section, we consider a partition  $U = \{I_1, \dots, I_n\}$  of the set of channels  $P_I \cup P_O$  of the oWFN  $M$ . We study decentralized controllability always with respect to a given partition.

**Definition 14 (Decentralized strategy).** *A set of controllers  $\{C_1, \dots, C_n\}$  is a decentralized strategy w.r.t.  $U = \{I_1, \dots, I_n\}$  if, for all  $i$ ,  $C_i$  is connected to  $I_i$  and  $\{C_1, \dots, C_n\}$  is a strategy.*

Our definitions imply that controllers involved in a decentralized strategy communicate with the oWFN but not with each other. If we allowed controllers to communicate with each other, they would be able to implement every centralized strategy, so decentralized controllability would not be worth being studied. On the other hand, situations where different parties communicate with a process without communicating with each other appears to be typical in the world of business processes. For example, a travel agency communicates with customers, airline reservation systems, and hotel reservation systems without letting these parties communicate with each other.

If a set of controllers forms a decentralized strategy then their parallel composition forms a central strategy. In the sequel let, for a word  $w$  over an alphabet  $A$ , and a subset  $B$  of  $A$ , the projection  $w_B$  of  $w$  to  $B$  be the word that is obtained from  $w$  by removing all letters not contained in  $B$ .

**Definition 15 (Parallel composition of tree controllers).** *Let  $C_1, \dots, C_n$  be tree controllers where  $C_i$  has  $Q_i$  as set of states and is connected to  $I_i$ . Then the controller  $C$  connected to  $\bigcup_{i=1}^n I_i$  that has  $q$  as its set of states where  $q \in Q$  if and only if, for all  $i$ ,  $q_{I_i} \in Q_i$ , is the parallel composition of  $C_1, \dots, C_n$ .*



**Fig. 7.** Parallel composition of controllers.

Fig. 7 shows an example for a parallel composition of controllers.

**Theorem 3.** *If  $\{C_1, \dots, C_n\}$  is a decentralized strategy for an oWFN  $M$  then their parallel composition  $C$  is a centralized strategy for  $M$ .*

**Proof.** By a simple induction on the structure of the transition system, the composed system of  $M$  and  $C_1, \dots, C_n$  is isomorphic to the one of  $M$  and  $C$ . q.e.d.

The two controllers depicted in the left part of Fig. 7 form a decentralized strategy for the oWFN in Fig. 6. Their parallel composition (depicted right in Fig. 7) is a central strategy and, as Thm. 2 states, included in the most permissive strategy (the automaton consisting of the states painted as solid circles in Fig. 6). This is only the case since we kept states  $q$  with  $K(q) = \emptyset$  in the most permissive strategy. In the case of distributed control, the controllers, though not communicating with each other, are synchronized via the oWFN. Hence, not all interleavings of states of the controllers are reachable. In the parallel composition, unreachable interleavings appear as states with empty  $K$ .

Thm. 3 suggests that a decentralized strategy can be found by looking for a central strategy that can be decomposed according to the given partition of the interface. Such a strategy must hold that moves that concern different classes of the interface partition are independent.

**Definition 16 (Independency).** *Let  $a, b \in P_I \cup P_O$ ,  $C$  a tree controller with set of states  $Q$ , and  $q \in Q$ .  $a$  enables  $b$  in  $q$  if  $qb \notin Q$  and  $qa, qab \in Q$ .  $a$  disables  $b$  in  $q$  if  $qb, qa \in Q$  and  $qab \notin Q$ . Two states  $q$  and  $q'$  are equivalent if, for all  $w$ ,  $qw \in Q$  if and only if  $q'w \in Q$ .  $a$  and  $b$  are independent if, for all states  $q$ ,  $a$  and  $b$  do not enable nor disable each other in  $q$  and, if  $qab, qba \in Q$  then they are equivalent.*

**Theorem 4.** *Let  $C$  be a controller connected to  $I$ , and  $I$  be partitioned into  $I_1, \dots, I_n$ . Then there exist Controllers  $C_1, \dots, C_n$  ( $C_i$  connected to  $I_i$ ) such that  $C$  is the parallel composition of  $C_1, \dots, C_n$  if and only if for all states  $q$  of  $C$ , all  $i$ , and all  $a, b \in I$ ,  $a \in I_i$  and  $b \notin I_i$  implies that  $a$  and  $b$  are independent.*

**Proof. Implication.** Let  $C$  be the parallel composition of the  $C_i$ . Let  $a \in I_j$  and  $b \in I_k$  ( $j \neq k$ ). Then  $qa, qab \in Q$  implies  $qb \in Q$  since  $qab_{I_k} = qb_{I_k}$ , so  $a$  does not enable  $b$ .  $qa, qb \in Q$  implies  $qab \in Q$  since  $qab_{I_j} = qa_{I_j}$ , so  $a$  does not disable  $b$ . Moreover, for all  $w$  and all  $l$ ,  $qabw_{I_l} = qbaw_{I_l}$ , so  $qab$  and  $qba$  are equivalent. Consequently,  $a$  and  $b$  are independent.

**Replication.** For  $i = 1, \dots, n$ , let  $Q_i = \{q_{I_i} \mid q \in Q\}$  and  $C_i$  the tree controller that has  $Q_i$  as set of states. We claim that  $C$  is the parallel composition of the  $C_i$ . First, if  $q_i \in Q_i$  then there is a  $q \in Q$  with  $q_{I_i} = q_i$ . Using independency, it can be shown that  $q_i q_{(P_I \cup P_O) \setminus I_i} \in Q$  and by prefix closure,  $q_i \in Q$ . By independency,  $q_1, \dots, q_n \in Q$  ( $q_i \in Q_i$ ) implies that every state  $q$  where, for all  $i$ ,  $q_{I_i} = q_i$ , is in  $Q$ . Thus,  $C$  is the parallel composition of the  $C_i$ . q.e.d.

Decentralized strategies can therefore be computed by extending Alg. 1. In the loop, not only states  $q$  that have deadlocks need to be removed but also states that are responsible for violating independency restrictions.

## Algorithm 2

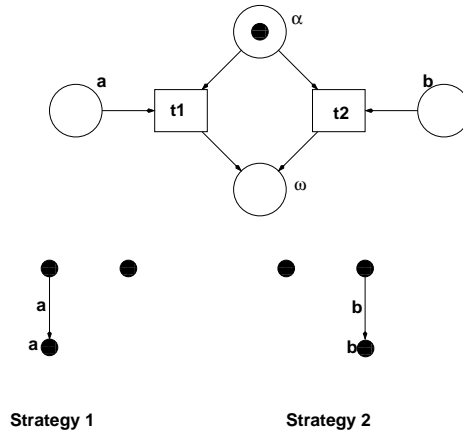
1.  $Q := \{w \mid w \in (P_I \cup P_O)^*, \text{length}(w) \leq l_M\}$ ;
2. REPEAT UNTIL  $Q$  does not change:
  - (a) IF exists  $q$  where  $K(q)$  contains a deadlock  $m$  and for all a possible in  $m$ ,  $qa \notin Q$  THEN remove  $q$ ;

- (b) IF exist  $q, a, b$ :  $a$  enables  $b$  in  $q$  THEN remove  $qab$ ;
  - (c) IF exist  $q, a, b$ :  $a$  disables  $b$  THEN remove  $qa$  or remove  $qb$ ;
  - (d) IF exist  $q, a, b$ :  $qab, qba \in Q$  and  $qab$  and  $qba$  are not equivalent THEN remove all  $qabw$  such that  $qbaw \notin Q$  and all  $qbaw$  where  $qabw \notin Q$ .
3. WHERE remove  $q = Q := Q \setminus \{qw \mid w \in (P_I \cup P_O)^*\}$ .

Observe that the step concerning disabling contains nondeterminism. As Fig. 8 illustrates, this nondeterminism seems to be necessary, as nondeterminism (or backtracking) is one of the few tools to break symmetry.

**Theorem 5.** An oWFN  $M$  is decentralized controllable w.r.t. a partition  $\{I_1, \dots, I_n\}$  of the interface if and only if Alg. 2 returns at least one nonempty set of states.

**Proof.** By Thm. 1 and Thm. 3, any set of states where none of the conditions inside the loop is applicable, is a central strategy and parallel composition of a set of controllers that consequently is a decentralized strategy. If a set of states contains any situation tested in the conditional statements, the removed states (in the case of disabling: not both of the removed states) can occur in any subset of  $Q$  that is both strategy and decomposable. q.e.d.



**Fig. 8.** For the depicted oWFN and the partition  $\{\{a\}, \{b\}\}$ , Alg. 2 would start with  $Q = \{\lambda, a, b, ab, ba\}$ .  $ab$  and  $ba$  are removed since the deadlocks  $\omega a$  and  $\omega b$  appear in  $K(ab)$  and  $K(ba)$ . After this removal  $a$  and  $b$  disable each other in  $\lambda$ . This yields the two depicted strategies. Though the oWFN is symmetric, the strategies are not symmetric in themselves.

The strategies depicted in Fig. 8 are the only strategies for the given partition. None of them can be more permissive than the other. Hence, the example shows that there are cases where there is no single most permissive decentralized strategy. However, our algorithm computes a set of strategies such that for every strategy, one of the computed strategies is at least as permissive.

## 5 Local controllability

Consider the following scenario: A company publishes a business process (or a public view thereof), we are one of the parties that wants to connect to that process via a given subset of the interface but do not know who else is connecting, nor how. Our task is to build a controller that, ideally, works properly with every set of controllers connected to the remaining parts of the interface. In this generality, our task cannot be accomplished, since we cannot avoid that a defecting controller denies to send a required message or sends too many messages of some kind. So it is at most possible to build a controller that works properly with arbitrary "non-defecting" controllers. In this section, we define *cooperative* controllers as a formalization for non-defection, and show that every feasible set of cooperative controllers forms a decentralized strategy. Unfortunately, cooperative controllers do not necessarily exist. For instance, in Fig. 8, assuming the interface partition  $\{\{a\}, \{b\}\}$ , cooperative controllers do not exist since it is impossible to choose among the two options depicted there without requiring that the other party picks the corresponding counterpart. We therefore define *local controllability* as the existence of cooperative strategies for all classes in a given interface partition. Locally controllable oWFN are therefore those whose publication is sufficient for enabling all communication partners to design their communication with the oWFN independently of each other.

Consider an oWFN  $M$  and a class  $I_i$  of a given partition  $\{I_1, \dots, I_n\}$  of  $P_I \cup P_O$ . We establish two quite plausible requirements for calling a controller  $C_i$  connected to  $I_i$  cooperative: First,  $C_i$  should behave like a central controller assuming that the channels  $(P_I \cup P_O) \setminus I_i$  were not present, and second,  $C_i$  should help in making progress, if it can. We formalize the first requirement by introducing the  $I_i$ -view of  $M$ .

**Definition 17 (*X*-view of  $M$ ).** *Let  $M$  be an oWFN and  $X \subseteq P_I \cup P_O$ . Then the  $X$ -view of  $M$ ,  $M_X$  is the oWFN obtained from  $M$  by removing all places in  $(P_I \cup P_O) \setminus X$  and all arcs connected to them.*

**Definition 18 (Cooperative controller).** *Let  $M$  be an oWFN and  $\{I_1, \dots, I_n\}$  a partition of  $P_I \cup P_O$ . Then controller  $C_i$  connected to  $I_i$  is cooperative if it is a central strategy for  $M_{I_i}$  and for all  $q \in C_i$ , all  $m \in K(q)$  such that*

- No transition outside  $P_{IM\bullet}$  is enabled in  $m$ ;
- there is a transition  $t \in P_{IM_i}$  such that
  - All  $p \in \bullet p \setminus P_{IM_i}$  are marked in  $m$ ;
  - there is an unmarked  $p \in \bullet p \cap P_{IM_i}$

*there is a  $b \in I_i$  possible in  $[m, q]$  such that  $qb \in Q_i$ .*

Informally: whenever there is a potential deadlock, that is, only externally controlled transitions (those in  $P_{IM\bullet}$ ) are enabled, and  $C_i$  can contribute to making progress, namely by helping enabling the mentioned transition  $t$ , then  $C_i$  does contribute to making progress.

**Theorem 6.** *Every feasible set of cooperative controllers forms a decentralized strategy for  $M$ .*

Note, that this statement is only relevant if cooperative controllers exist for all classes of an interface partition.

**Proof.** We show that every terminal state in the composed system has  $m_f$  as first component. Consider an arbitrary terminal state  $[m^*, q_1^*, \dots, q_n^*]$  of the composed system. Let  $m_{i^*}$  be the projection of  $m^*$  to  $P_M \cup I_i$ . By definition of the views on  $M$ , for all  $i$ ,  $[m_{i^*}, q_i]$  is reachable in the system composed of  $M_{I_i}$  and  $C_i$ . Assume  $m^*$  is a deadlock. If every transition of  $M$  has an unmarked pre-place in  $P_M$  then all  $m_{i^*}$  are deadlocks, too, contradicting the assumption that the  $C_i$  are central strategies. Otherwise, there is at least one transition such that  $t$  has an unmarked pre-place  $p^*$  in  $P_I$ . Since  $m^*$  is a deadlock, transitions outside  $P_{IM^\bullet}$  are not enabled in  $m_{i^*}$ . Thus, Def. 18 states that at least one  $C_i$  (the one with  $p^* \in I_i$ ) has a move in  $m_{i^*}$ . This move is possible in  $m^*$ , too, contradicting the assumption that the considered state is terminal. q.e.d.

This result justifies the definition of local controllability.

**Definition 19 (Local controllability).** *An oWFN  $M$  is locally controllable w.r.t. a partition  $\{I_1, \dots, I_n\}$  of  $P_I \cup P_O$  if, for all  $i$ , there exists a cooperative strategy for  $I_i$ .*

Given  $M$  and  $I_i$ , existence of a cooperative strategy can be decided using a slight modification of Alg. 1. The modification consists of working on  $M_{I_i}$  instead of  $M$ , but identifying deadlocks as if we worked on  $M$ .

**Corollary 2.** *If there is a cooperative controller for  $I_i$  then there is also a most permissive one.*

For the oWFN in Fig. 1, the controller depicted left in Fig.2 is cooperative while the one depicted right is not. In the latter one, move  $b$  is possible in marking  $\beta\chi \in K(6)$  but not executed in 6. If both controllers behaved like the one depicted right (observe that the oWFN is symmetric w.r.t. the interface partition) then the composed system would deadlock in  $\beta\chi$ . Since cooperative controllers exist (like the one depicted left), the oWFN in Fig. 1 is locally controllable. The oWFN in Fig. 8 is not locally controllable w.r.t.  $\{\{a\}, \{b\}\}$ : Consider part  $\{a\}$ . State  $a$  cannot be present since  $K(a) = \{\alpha a, \omega, \omega a\}$  contains the deadlock  $\omega a$  reached by firing  $b$  in  $M_{\{a\}}$ . Without state  $a$ , the remaining controller does not help in leaving the deadlock  $\alpha \in K(\lambda)$  though  $a$  is possible in  $\alpha$ .

## 6 Comparison

By the established conditions, it is clear that every locally controllable oWFN is decentralized controllable w.r.t. the same partition of the interface, and every decentralized controllable oWFN is centralized controllable (proven via parallel composition). All inclusions are strict. Fig. 8 shows an oWFN that is not locally

controllable but decentralized controllable. An oWFN that expects a  $c$  after having sent an  $a$ , and expects a  $d$  after having sent a  $b$  is centralized controllable, but not decentralized controllable for the partition  $\{\{a, b\}, \{c, d\}\}$ . For an interface partition consisting of exactly one class (the central case), the requirements for centralized, decentralized, and local controllability coincide, and the most permissive cooperative local controller is just the most permissive central controller. We may therefore conclude that the constituents of the theory presented in this paper fit together quite nicely.

## 7 Related work

Central controllability for oWFN based on message passing was studied in [Mar03]. MARTENS uses, however a different structure for modelling controllers. His structure is a bipartite graph where send and receive phases alternate. Furthermore he is more restrictive concerning possible moves: in particular he removes states  $q$  with  $K(q) = \emptyset$ . With these differences, he could not prove the existence of most permissive controllers - though he is able to compute some controller whenever one exists. In his setting, the parallel composition of a decentralized strategy is not necessarily a centralized strategy, so his structure is not suitable for studying decentralized controllability. MARTENS proposes an algorithm similar to Alg. 1 and proves it using observations similar to the ones formalized in Thm. 1. However, his approach is more complicated, proofs are significantly longer. On the other hand, his controllers are more condensed than our ones, an advantage that can, however, be compensated by applying reduction techniques to our construction [Wei04].

Our control problem can be formulated [MSS05] as a model checking problem for alternating-time temporal logic. The (more general) model checking problem for that logic with incomplete information (our case) is, however, undecidable.

Supervisory control using Petri net models as plants were all based on control by selecting controllable transitions rather than message passing. They assume fully observable Petri net states [LW94,HGZ96], considered classes of Petri nets that are incomparable to our one [ARX04,HK90,BBS98,GRX03,DX03], or propose different objectives for control [YMLA96,HK90,GRX03,GS02].

Control problems for general discrete event systems or general Petri nets studied in [RW87,LW94,MA99,Str00], as far as they cover our setting, consider much more general settings and thus provide weaker results (e.g., concerning existence of most permissive strategies).

To our best knowledge, there is no counterpart to our concept of local controllability in the literature.

## 8 Conclusion

We studied control problems for a class of Petri nets. Both the considered class and the studied control problems are motivated by possible applicability to distributed business processes. Since, at this time, our results are not yet imple-

mented, we cannot validate whether our approach can be implemented efficiently enough for solving practically relevant problem instances. We showed that there is a most permissive central strategy, if any, solved the decentralized control problem by looking for special central strategies and showed, that a most permissive strategy does not always exist. Finally, we defined the concept of local controllability as a special case of decentralized controllability. This concept relies on cooperative controllers. We stated that there is a most permissive cooperative controller, if any.

Ongoing research includes improving the reduction techniques, aiming at algorithms that are applicable to large models, extending the approach to oWFN that contain cycles, and considering advanced control objectives such as transactional correctness (for transitions modelling database operations).

## References

- [Aal98] W.M.P. van der Aalst. The application of Petri nets to workflow management. *Journal of circuits, systems, and computers* 8(1) pp. 21–66, 1998.
- [AH01] L. de Alfaro and T.A. Henzinger. Interface automata. *Proc. FSE*. pp. 109–120, 2001.
- [AHK02] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal ACM* 49, pp. 672–713, 2002.
- [ARX04] Z. Achour, N. Rezg, and X. Xie. Supervisory control of partially observable marked graphs. *IEEE Transactions on Automatic Control* 49 (11), pp. 2007–2011, 2004.
- [BBS98] R.K. Boel, B. Bordbar, and G. Stremersch. A min-plus polynomial approach to forbidden state control for general PNs. *Proc. WODES*, pp. 79–84, 1998.
- [BD98] E. Badouel and P. Darondeau. Theory of regions. *Lectures on Petri nets 1: basic models*, pp. 529–258. LNCS 1491, 1998.
- [CGK<sup>+</sup>03] Curbera, Golland, Klein, Leymann, Roller, Thatte, and Weerawarana. Business Process Execution Language for Web Services, Version 1.1. Technical report, BEA Systems, Interantional Business Machines Corporation, Microsoft Corporation, May 2003.
- [DX03] P. Darondeau and X. Xie. Linear control of live marked graphs. *Automatica* 39 (3), pp. 429/440, 2003.
- [GRX03] A. Ghaffari, N. Rezg, and X. Xie. Feedback control logic for forbidden state problem of marked graphs. *IEEE Transactions on Automatic Control* 48, pp. 18–29, 2003.
- [GS02] A. Giua and C. Seatzu. Observability of place/transition nets. *Transactions on Automatic Control* 47, pp. 1424–1437, 2002.
- [HGZ96] L. Holloway, X. Guan, and L. Zhang. A generalization of state avoidance policies for controlled PN. *IEEE Transactions on Automatic Control* 39, pp. 512–531, 1996.
- [HK90] L. Holloway and B. Krogh. Synthesis of feedback control logic for a class of controlled PNs. *IEEE Transactions in Automatic Control* 35, pp. 514–523, 1990.
- [HSS05] S. Hinz, K. Schmidt and C. Stahl. Transforming BPEL to Petri nets. Accepted for *Int. Conf. Business Process Management (BPM)* 2005

- [LW94] Y. Li and W.M.Wonham. Control of vector discrete event systems II – controller synthesis. *IEEE transactions on Automatic Control* 39, pp. 512-531, 1994.
- [MA99] J.O. Moody and P.J. Antsaklis. PN supervisors for DES with uncontrollable and unobservable transitions. Fevrier, Tech. Report ISIS-99-04, 1999.
- [Mar03] A. Martens. Verteilte Geschäftsprozesse – Modellierung und Verifikation mit Hilfe von Web Services. Dissertation, Humboldt-Universität zu Berlin, 2003.
- [MS05] P. Massuthe and K. Schmidt. Operating guidelines - an automata-theoretic foundation for the service-oriented architecture. Accepted for 1st Int. Workshop on Services Engineering, Melbourne 2005.
- [MSS05] A. Martens, H. Schlingloff, and K. Schmidt. Modeling and Model Checking Web Services. submitted to *Electronic Notes in Theoretical Computer Science*, spec. Sec. on LCMAS 04.
- [NRT92] M. Nielsen, G. Rozenberg, and P.S. Thiagarajan. Elementary transition systems. *Theoretical Computer Science* 96, 1992, pp. 3-33.
- [RW87] P.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete event processes. *SIAM J.Control and Optimization* 25(1), pp. 206–230, 1987.
- [Sta04] C. Stahl. Transformation von BPEL4WS in Petrinetze. Diplomarbeit, Humboldt-Universität zu Berlin, April 2004.
- [Str00] G. Stremersch. Linear algebraic design of supervisors for partially observed PN. . *Proc. CSD Bratislava*, pp. 281-286, 2000.
- [Wei04] D. Weinberg. Implementation der Bedienbarkeit. Diploma Thesis, Humboldt-Universität zu Berlin, 2004.
- [YMLA96] K. Yamalidou, J. Moody, M. Lemmon, and P. Antsaklis. Feedback control of PNs based on place invariants. *Automatica* 32 (1), pp. 15-28, 1996.