Methods for the specification and verification of business processes

MPB (6 cfu, 295AA)

Roberto Bruni
http://www.di.unipi.it/~bruni

20 - Workflow modules
We study Workflow modules to model interaction between workflows
Problem

Not all tasks of a workflow net are automatic:
they can be triggered manually or by a message
they can be used to trigger other tasks

How do we represent this?
Implicit interaction

Separately developed workflow

Some activities can **input** messages

Some activities can **output** messages
Implicit interaction

Seller can receive (symbol ?) recommendations

Seller can send (symbol !) decisions
Interface

Seller has an interface for interaction

It consists of some **input** places and some **output** places
Interface

sending !

receiving ?
Problem

Assume the original workflow net has been validated:

it is a sound (and maybe safe) workflow net

When we add the (places in the) interface it is no longer a workflow net!
Workflow Modules

Definition: A workflow module consists of

a workflow net \((P, T, F)\)

plus a set \(P^I\) of incoming places
plus a set of incoming arcs \(F^I \subseteq (P^I \times T)\)

plus a set \(P^O\) of outgoing places
plus a set of outgoing arcs \(F^O \subseteq (T \times P^O)\)

such that each transition has
at most one connection to places in the interface
Problem

Workflow modules must be capable to interact

How do we check that their interfaces match?

How do we combine them together?
Strong structural compatibility

A set of workflow modules is called **strongly structural compatible** if for every message that can be sent there is a module who can receive it, and for every message that can be received there is a module who can send it

(formats of message data are assumed to match)
Weak structural compatibility

A set of workflow modules is called weakly structural compatible if all messages sent by modules can be received by other modules more likely than a complete structural match (workflow modules are developed separately)
Interaction
We have added places and arcs to single nets
We have joined places of different nets
We have paired their initial markings

How do we check that the system behaves well?

What has this check to do with WF net soundness?
Workflow systems
Workflow system
Workflow system

Definition: A workflow system consists of

a set of n structurally compatible workflow modules
(initial places $i_1, \ldots, i_n$, final places $o_1, \ldots, o_n$)

plus an initial place $i$
and a transition $t_i$ from $i$ to $i_1, \ldots, i_n$

plus a final place $o$
and a transition $t_o$ from $o_1, \ldots, o_n$ to $o$
Soundness of workflow systems

A workflow system is just an ordinary workflow net

We can check its soundness as usual
Exercise

Can the system deadlock?
Exercise

Can the system deadlock?
Exercise

Complete with missing arcs the following behavioural interfaces and check their compatibility.
Exercise

Check compatibility of WF modules below

(a)  

(b)  

(c)
Weak soundness
Problem

When checking behavioural compatibility the soundness of the overall net is a too restrictive requirement

Workflow modules are designed separately, possibly reused in several systems. It is unlikely that every functionality they offer is involved in each system.
Problem

Definition: A workflow net is \textit{weak sound} if it satisfies “option to complete” and “proper completion”

(dead tasks are allowed)

Weak soundness can be checked on the RG

It guarantees deadlock freedom and proper termination of all modules
Sound + Sound = ?
Sound + Sound = not sound
Sound + Sound = not sound
Sound + Sound = not sound
Sound + Sound = not sound
Sound + Sound = not sound
Sound + Sound = not sound
Sound + Sound = not sound
Sound + Sound = not sound
Sound + Sound = not sound

Dead tasks!
Sound + Sound = not sound
Exercise: Preliminaries

\[ N_0^{part} \]

contractor

\[ N_1^{part} \]

subcontractor

order

specification

cost_statement

product
Exercise: Check Weak Soundness of The Assembly
Exercise: Check Again After Refactoring Contractor

Diagram showing the process flow with nodes labeled as `order`, `specification`, `cost_statement`, and `product`.
Exercise: Check Again After Refactoring Both
(Contractor zoom-in)
(Subcontractor zoom-in)
Partner existence
(aka controllability)

Does My Service Have Partners?

Karsten Wolf

Universität Rostock, Institut für Informatik, 18051 Rostock, Germany
karsten.wolf@uni-rostock.de
Problem

Processes are designed in isolation (loose coupling)
We would like their composition to be well-behaved

Given a process:

Can we guarantee that at least one partner exists?

If so, can we synthesize the most permissive partner?
Controllability

Assume a notion of well-behaving is defined

We say that a process N is **controllable** if it has at least one partner N’ such that the composition of N with N’ is well-behaving.
Controllability: idea

Given a process $N$

we aim to construct an automaton that

over-approximates the behaviour of any partner,

then we iteratively remove states and arcs that

invalidate the behavioural property we are after:

if we end up with the empty automaton,

then the process $N$ is uncontrollable

otherwise, the automaton defines the most general

strategy to collaborate with $N$

(guaranteeing the behavioural property we are after)
Open nets

**Definition:** An open net consists of a net \((P,T,F,m_0)\) plus incoming places \(P^I \subseteq P\) plus outgoing places \(P^O \subseteq P\) plus a finite set \(M_f\) of final markings such that

- each transition has at most one connection to places in the interface
- any initial or final marking does not mark any place in the interface.
Example: open net

\[ m_0 = a \]
\[ P^i = \{ i \} \]
\[ P^o = \{ o \} \]
\[ M_f = \{ b \} \]
Definition: The label of a transition $t$ is the interface place connected to $t$, if any, or the special silent action $\tau$ otherwise.

$$\ell(t) = \begin{cases} x & \text{if } x \in (P^I \cup P^O) \text{ and } (t, x) \in F \lor (x, t) \in F \\ \tau & \text{otherwise} \end{cases}$$
Example: open net

\[ m_0 = a \]
\[ P^I = \{ i \} \]
\[ P^O = \{ o \} \]
\[ M_f = \{ b \} \]

\[ \ell(t_1) = i \]
\[ \ell(t_2) = o \]
Closed nets

An open net

\[ N = (P, T, F, m_0, P_I, P_O, M_f) \]

is called **closed** if

\[ P_I = P_O = \emptyset \]
Inner nets

Let $N = (P, T, F, m_0, P^I, P^O, M_f)$ be an open net and let $IO = (P^I \cup P^O)$ be its interface.

Its inner net $In(N)$ is the closed net obtained by removing the interface

$In(N) = (P \setminus IO, T, F \setminus ((IO \times T) \cup (T \times IO)), m_0, \emptyset, \emptyset, M_f)$
Example: inner net
Bounded and responsive nets

We focus on open nets $N$ such that their inner nets $\text{In}(N)$ are bounded (in the usual sense, they have finite occurrence graphs) responsive from any marking $m$ either a final marking is reachable or $m$ enables a transition $t$ connected to the interface.
Composition of open nets

Given two open nets
\[ N_1 = (P_1, T_1, F_1, m_{01}, P_{I_1}, P_{O_1}, M_{f1}) \]
\[ N_2 = (P_2, T_2, F_2, m_{02}, P_{I_2}, P_{O_2}, M_{f2}) \]

such that \( P_{I_1} = P_{O_2} \) and \( P_{O_1} = P_{I_2} \)

...
Composition of open nets

Given two open nets
\[ N_1 = (P_1, T_1, F_1, m_{01}, P^I, P^O, M_{f1}) \]
\[ N_2 = (P_2, T_2, F_2, m_{02}, P^O, P^I, M_{f2}) \]

their composition \( N = N_1 \oplus N_2 \) is the closed net
\[ N = ( P_1 \cup P_2 \ , \ T_1 \cup T_2 \ , \ F_1 \cup F_2 \ , \ m_{01} + m_{02} \ , \ \emptyset \ , \ \emptyset \ , \ M_{f1} \times M_{f2} ) \]

note that even if \( N_1 \) and \( N_2 \) are bounded and responsive,
their composition \( N_1 \oplus N_2 \) is not necessarily so
Behavioural properties: DF

A closed net $N = (P, T, F, m_0, \emptyset, \emptyset, M_f)$ is DF (deadlock-free) if any non-final reachable marking enables a transition

$$\forall m \in ([m_0] \setminus M_f). \exists t. M \xrightarrow{t}$$
DF,$k$-controllable nets

A bounded and responsive open net

\[ N = (P, T, F, m_0, P^I, P^O, M_f) \]

is DF,$k$-controllable

if there is a (bounded and responsive) partner $N'$ such that

\[ N \oplus N' \text{ is DF} \]

and any place $p \in (P^I \cup P^O)$ is $k$-bounded in $N \oplus N'$ such that

such an $N'$ is called a **DF,$k$-strategy** of $N$
Approach

Start with a bounded and responsive open net
\[ N = (P, T, F, m_0, P^I, P^O, M_f) \]

1st step
Define a strategy \( TS_0 \) which is an automaton such that a state \( q \) of \( TS_0 \) represents the set of markings \( N \) can be in while \( TS_0 \) is in \( q \)

i.e. \( q \) is the view of \( N \) according to the interactions observed so far

Note that \( TS_0 \) can be infinite
Notation

A special symbol \( \# \) tags final states

Given a set of markings \( M \) of \( N \), we denote by \( \text{cl}(M)_N \) the closure of \( M \)

i.e., the set of markings reachable in \( N \) from any of the markings in \( M \)

\[
\text{cl}(M)_N = \{ m' \mid \exists m \in M. \ m' \in [M]_N \}
\]
$$\text{TS}_0 = (Q, E, q_0, Q_f)$$

$$q_0 = cl(\{m_0\})_N \quad q_0 \in Q$$

if \( q \in Q \)

if \# \not\in q \quad \text{any state q has a final counterpart} 

then \( q' = q \cup \# \in Q, q \xrightarrow{\tau} q \in E, q \xrightarrow{x} q' \in E \)

if \( x \in P^I \wedge \# \not\in q \) \quad \text{TS}_0 \text{ simulates the production of messages in input places} 

then \( q' = cl(\{m + x \mid m \in q\}) \in Q, q \xrightarrow{x} q' \in E \)

if \( x \in P^O \) \quad \text{TS}_0 \text{ simulates the consumption of messages from output places} 

then \( q' = \{m - x \mid m \in q, m(x) > 0\} \setminus \{\#\} \in Q, q \xrightarrow{x} q' \)

\( Q_f = \{q \in Q \mid \# \in q\} \)
Approach

Starting from the strategy $TS_0$

2nd step

Define a strategy $TS_1$
that removes from $TS_0$ all states $q$
that contains a marking $m$ that exceeds the capacity
bound $k$ for some place in the interface

(as a consequence remove all adjacent edges
and all states that become unreachable)

$TS_1$ is always a finite automaton
(actually it can be constructed directly from $N$)
\[ TS_1 \]

\[ TS_0 = (Q, E, q_0, Q_f) \]

\[ Q_1 = Q \setminus \{ q \in Q \mid \exists m \in q. \exists x \in (P^I \cup P^O). m(x) > k \} \]

\[ E_1 = \{ q \xrightarrow{\alpha} q' \in E \mid q, q' \in Q_1 \} \]

\[ Q_{f1} = Q_f \cap Q_1 \]

\[ TS_1 = (Q_1, E_1, q_0, Q_{f1}) \]
DF,1-controllability example: TS₁
DF,1-controllability example: $TS_1$

any state $q$ has a final counterpart
DF,1-controllability example: TS₁

simulates the production of messages in input places

\[ cl\{a + i, b + o + i\} \]
DF,1-controllability example: $TS_1$

any state $q$ has a final counterpart
$q_1$ has not outgoing arc with label $i$ because of 1-boundedness (this is $TS_1$, not $TS_0$)
DF,1-controllability
example: TS₁

cl(\{b\})ₜₙ

simulates the consumption of messages from output places
DF,1-controllability example: TS₁

any state q has a final counterpart
DF,1-controllability
example: TS₁

cl(\{b + i\})_N

simulates the consumption of messages from output places

simulates the production of messages in input places

N
DF,1-controllability example: TS$_1$

any state $q$ has a final counterpart
DF,1-controllability example: $TS_1$

(simulates the consumption of messages from output places)
DF, 1-controllability example: $TS_1$

\[ cl(\{\})_N \]

Simulates the consumption of messages from output places.
DF,1-controllability example: $TS_1$

any state $q$ has a final counterpart
DF,1-controllability
example: TS₁
Approach
Starting from the strategy $TS_i$

Iterative step
Define a strategy $TS_{i+1}$
that removes from $TS_i$ all states $q$
that can invalidate the property of interest

(as a consequence remove all adjacent edges
and all states that become unreachable)

At some point $TS_{j+1} = TS_j$
and we terminate
We can compose $T S_i$ with $N$, written $T S_i \oplus N$

States are pairs $[q, m]$ with $q \in Q_i$ and $m \in q$

**Notation**

if $m \xrightarrow{t} m'$ in $N$ then $[q, m] \xrightarrow{\ell(t)} [q, m']$

if $q \xrightarrow{\tau} q' \in E_i$ then $[q, m] \xrightarrow{\tau} [q', m]$

if $q \xrightarrow{x} q' \in E_i$ with $x \in P^I$ then $[q, m] \xrightarrow{x} [q', m + x]$

if $q \xrightarrow{x} q' \in E_i$ with $x \in P^O$ and $m(x) > 0$ then $[q, m] \xrightarrow{x} [q', m - x]$
DF,1-controllability
example: $TS_1 \oplus N$
$TS_{i+1}$

$TS_i = (Q_i, E_i, q_0, Q_{fi})$

remove the state $q$ if $q \xrightarrow{\tau} q$ is the only edge with source $q$

remove the state $q$ if $\exists m \in q$ such that in $TS_i \oplus N$

if $[q', m']$ is reachable from $[q, m]$ then $m' \notin M_f$

and only sequences of $\tau$-steps are possible from $[q, m]$
DF,1-controllability example: $TS_2$
DF,1-controllability example: $TS_2$
DF,1-controllability example: TS$_2$
DF,1-controllability example: $TŞ_2$
DF, 1-controllability example: TS$_2$
DF,1-controllability example: $TS_2$
DF,1-controllability example: TS₂ ⊕ N
DF,1-controllability
example: TS₂ ⊕ N
DF,1-controllability

example: $TS_2 \oplus N$
DF,1-controllability

example: $TS_2 \oplus N$
DF,1-controllability example: $TS_2 \oplus N$
DF,1-controllability

example: $TS_2 \oplus N$
DF,1-controllability
example: $TS_2 \oplus N$
DF,1-controllability example: $\text{TS}_3$
DF, 1-controllability
example: TS₃ ⊕ N
Note

Note the presence of a state associated with the empty set of markings.

It can appear in a strategy, but is actually unreachable in the composition with N.

For DF, k-controllability it makes no harm.
DF,k-controllability theorem

Let $TS = (Q,E,q_0,Q_f)$ be the automaton produced by applying the above procedure to the open net $N$

**Theorem**

$N$ is DF,k-controllable

iff

$Q \neq \emptyset$

(proof omitted)
A closed net $N = (P, T, F, m_0, \emptyset, \emptyset, M_f)$ is LF (**livelock-free**) if from any reachable marking a final marking is reachable

$$\forall m \in [m_0]. \ [m] \cap M_f \neq \emptyset$$
**LF,\(k\)-controllable nets**

A bounded and responsive open net

\[ N = (P,T,F,m_0,P^I,P^O,M_f) \]

is **LF,\(k\)-controllable**

if there is a (bounded and responsive) partner \(N'\) such that

\[ N \oplus N' \text{ is LF} \]

and any place \(p \in (P^I \cup P^O)\) is \(k\)-bounded in \(N \oplus N'\)

such an \(N'\) is called a **LF,\(k\)-strategy** of \(N\)
Approach

We construct $TS_0$ and $TS_1$ as before. We change only the iterative step.

Iterative step

Define a strategy $TS_{i+1}$ that removes from $TS_i$ all states $q$ that can invalidate the property of interest.

(as a consequence remove all adjacent edges and all states that become unreachable)

At some point $TS_{j+1} = TS_j$ and we terminate.
\( TS_{i+1} \)

\[ TS_i = (Q_i, E_i, q_0, Q_{fi}) \]

remove the state \( q \) if \( \exists m \in q \) such that in \( TS_i \oplus N \)

no \([q', m']\) with \( q' \in Q_{fi} \land m' \in M_f \) is reachable from \([q, m]\)
LF,k-controllability theorem

Let $TS = (Q, E, q_0, Q_f)$ be the automaton produced by applying the above procedure to the open net N

**Theorem**

N is LF,k-controllable

iff

$Q \neq \emptyset$

(proof omitted)