Fine-grained Compatibility and Replaceability Analysis of Timed Web Service Protocols

Julien Ponge[1,2], Boualem Benatallah[2], Fabio Casati[3] and Farouk Toumani[1]

(1) Université Blaise Pascal, Clermont-Ferrand, France

(2) UNSW, Sydney, Australia

(3) University of Trento, Italy

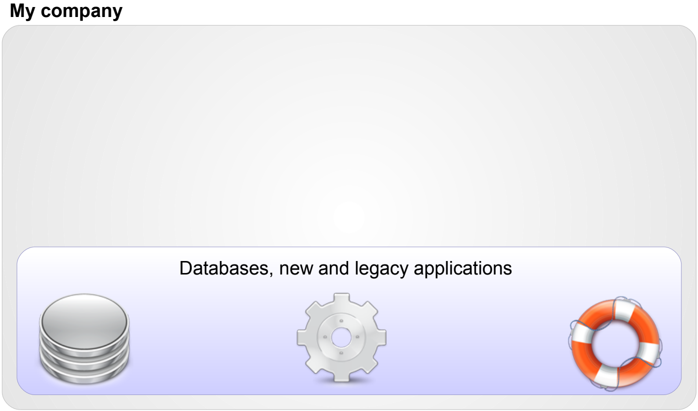
ER 2007, Auckland, New Zealand

Outline

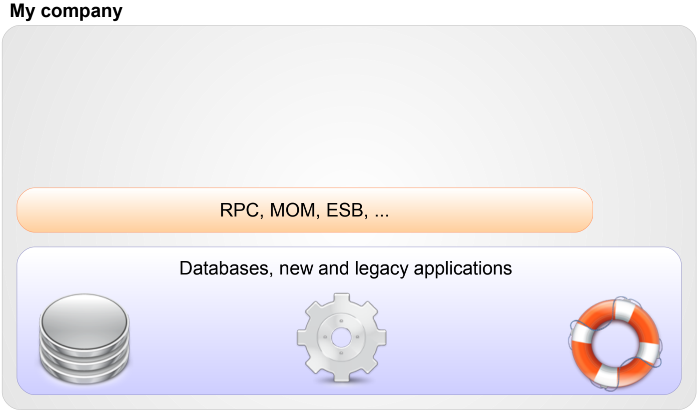
Outline

1. Introduction
2. Timed protocols
3. Formal framework
4. Implementation and conclusion

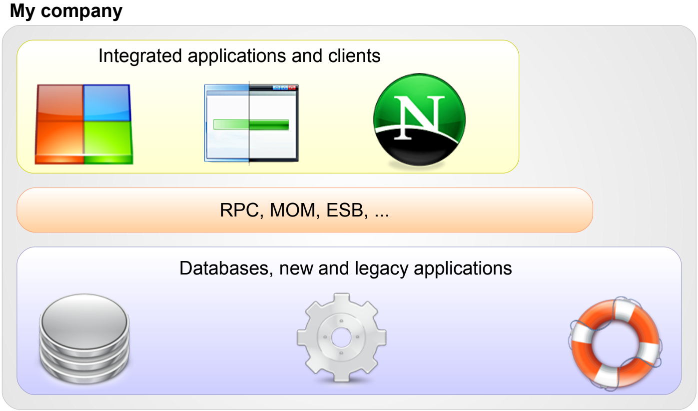
WS for application integration



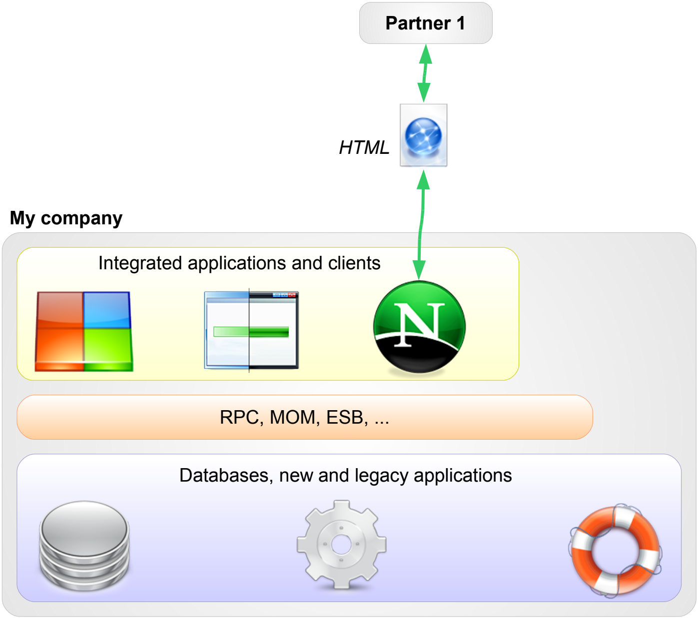
WS for application integration



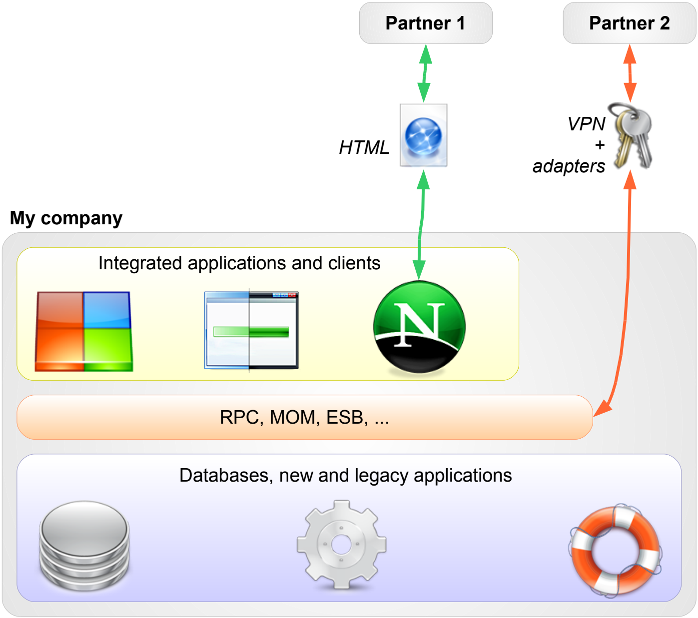
WS for application integration



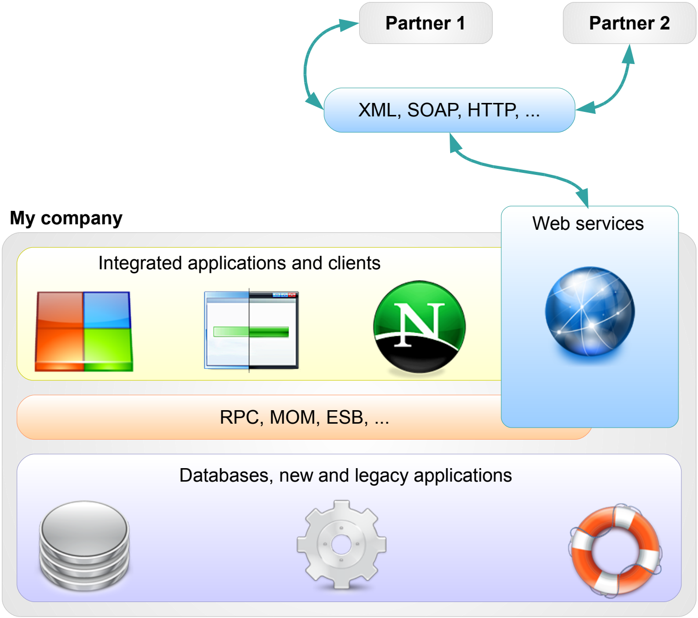
WS for application integration



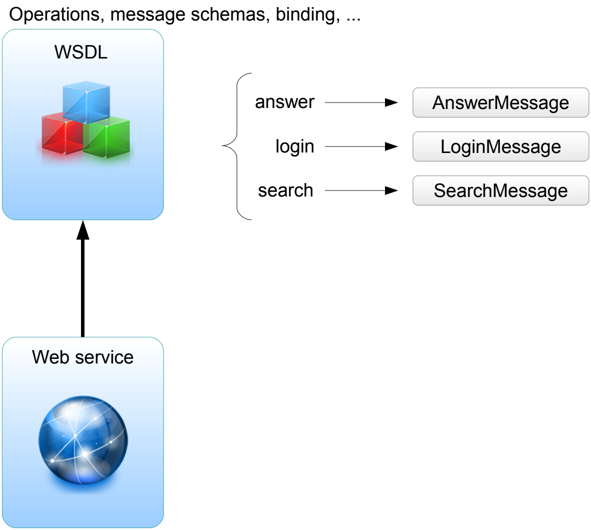
WS for application integration



WS for application integration

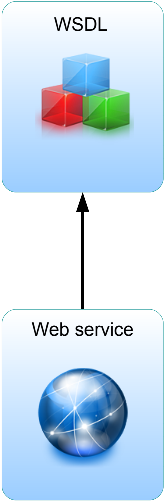


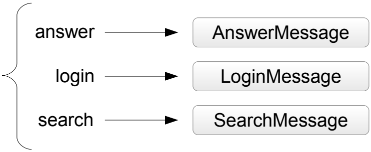
Static vs dynamic interface



Static vs dynamic interface

Operations, message schemas, binding, ...





Valid conversations:

```
login, search, answer
login, search, answer, search, answer
(...)
```

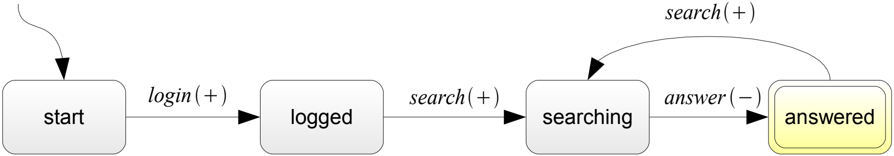
Invalid conversations:

```
search, login, answer
answer, search, login
(...)
```

Business Protocols

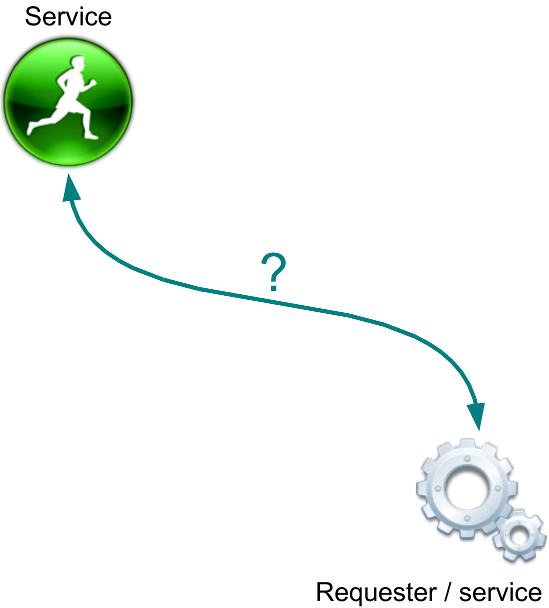
[ER 2004, DKE: Benatallah, Casati, Toumani]

- Conversations: message choreographies
- Finite deterministic automata
- Execution traces semantics



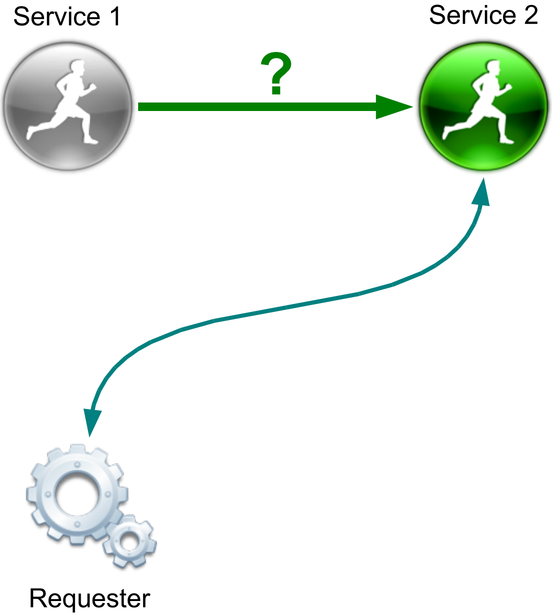
Extensions: transactions, timing constraints, policies, ...

Compatibility analysis

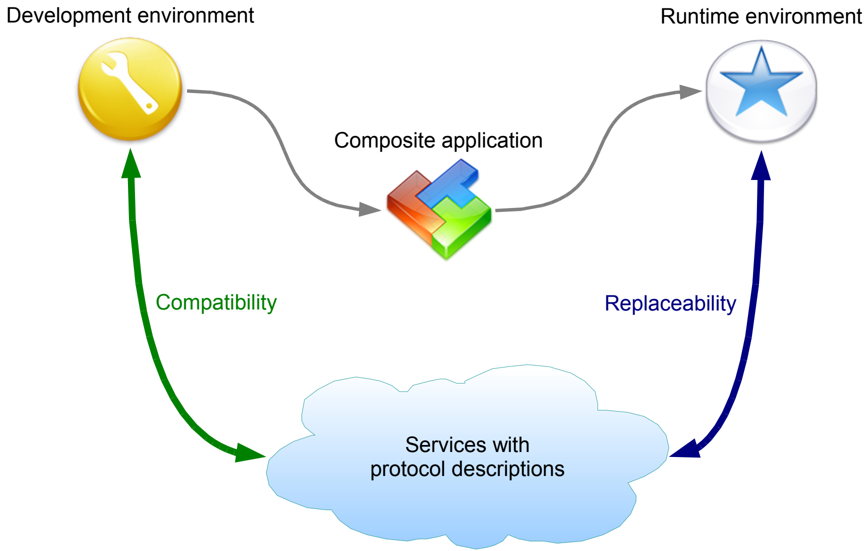


Replaceability analysis

Replaceability analysis



Use-case: agile composition runtimes



A need for timing constraints

Many examples:

- TCP/IP, watchdogs
- transaction locks
- business agreements
- BPEL (`wait` / `onAlarm`)
- RosettaNet
- ...

Outline of contributions

1. Extension of business protocols
2. Compatibility and replaceability analysis
3. A new class of timed automata
4. Implementation

Outline

Primitives

C-Invoke
: Temporal windows for a message exchange

M-Invoke
: Expiration for an implicit state change

C-Invoke($(T_1 < \text{12h:50m}) \wedge (T_2 > \text{1h})$)
M-Invoke($(T_1 = \text{6h}) \wedge (T_2 > \text{1h})$)
$(\cdots)$

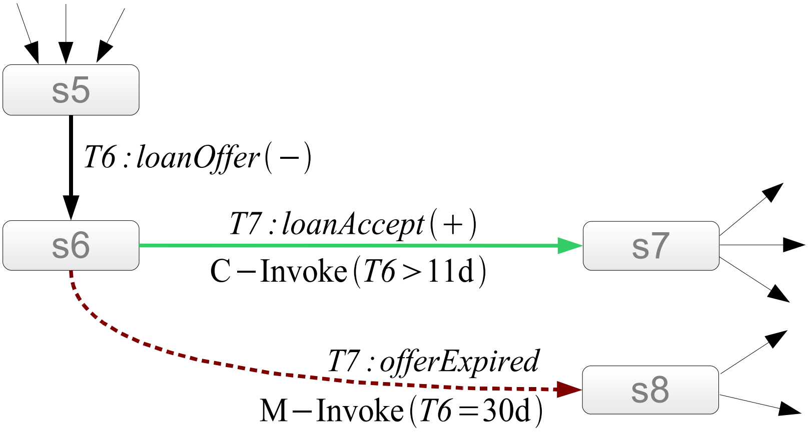
Primitives

C-Invoke
: Temporal windows for a message exchange

M-Invoke
: Expiration for an implicit state change

C-Invoke$((T_1 < 12\text{h:}50\text{m}) \wedge (T_2 > 1\text{h}))$
M-Invoke$((T_1 = 6\text{h}) \wedge (T_2 > 1\text{h}))$
$(\cdots)$

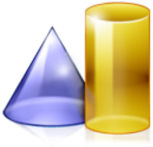
Extensions



s5
$T6 : loanOffer(-)$
s6
$T7 : loanAccept(+)$
$\mathrm{C-Invoke}(T6 > 11\mathrm{d})$
s7
$T7 : offerExpired$
$\mathrm{M-Invoke}(T6 = 30\mathrm{d})$
s8

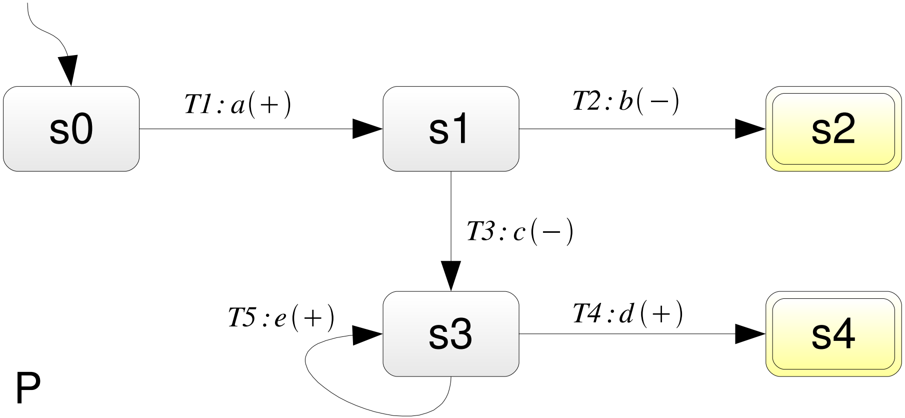
Analysis classes

- Compatibility:
  - full
  - partial
- Replaceability:
  - full
  - partial
  - subsumption, equivalence
  - w.r.t. client protocol
  - w.r.t. interaction role



A set of flexible classes because of a versatile environment

Illustration of replaceability w.r.t. client protocol



P

Illustration of replaceability w.r.t. client protocol

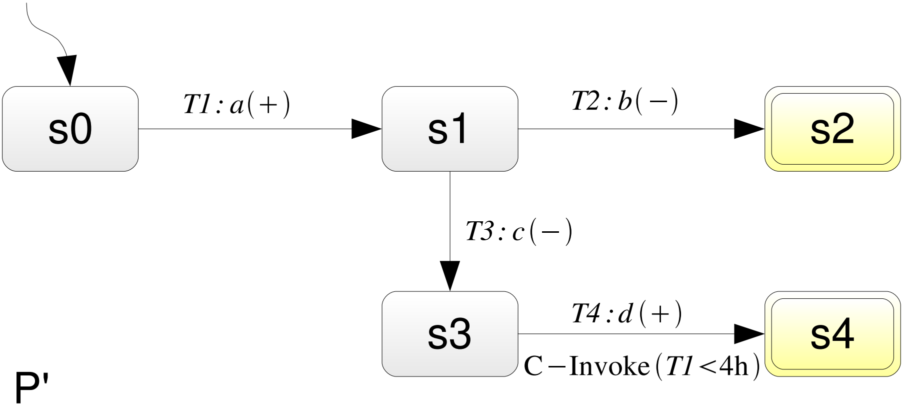


Illustration of replaceability w.r.t. client protocol

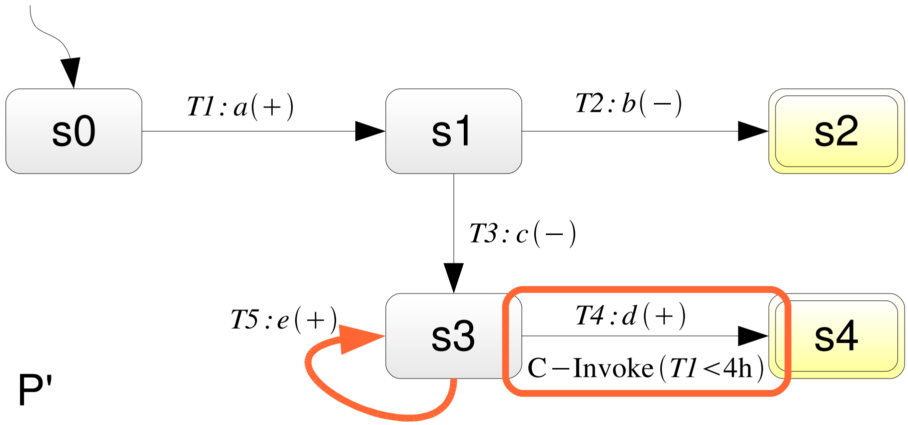


Illustration of replaceability w.r.t. client protocol

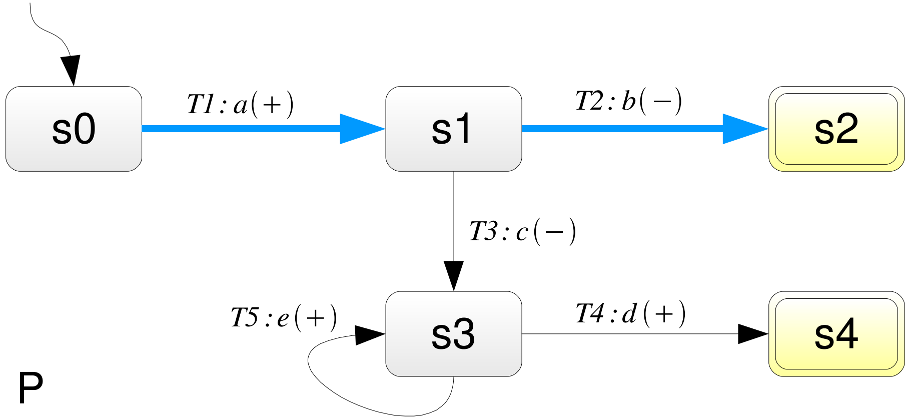
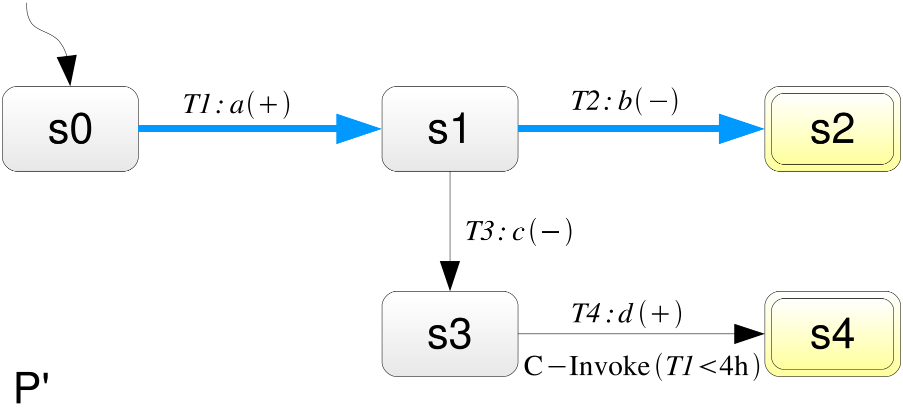


Illustration of replaceability w.r.t. client protocol



Characterization through operators

Comparison

subsumption ($\sqsubseteq$), equivalence ($\equiv$)

Manipulation

parallel composition ($\|^{\mathtt{TC}}$), intersection ($\|^{\mathtt{TI}}$), difference ($\|^{\mathtt{TD}}$)

Example: $\mathcal{P}_1$ can replace $\mathcal{P}_2$ w.r.t. a client protocol $\mathcal{P}_C$ iff:

- $\left[\mathcal{P}_C \|^{\mathtt{TC}} \mathcal{P}_2\right]_{\mathcal{P}_2} \sqsubseteq \mathcal{P}_1$, or
- $\mathcal{P}_C \|^{\mathtt{TC}} (\mathcal{P}_2 \|^{\mathtt{TD}} \mathcal{P}_1) = \emptyset$

Characterization through operators

Comparison

subsumption ($\sqsubseteq$), equivalence ($\equiv$)

Manipulation

parallel composition ($\|^{\mathtt{TC}}$), intersection ($\|^{\mathtt{TI}}$), difference ($\|^{\mathtt{TD}}$)

Example: $\mathcal{P}_1$ can replace $\mathcal{P}_2$ w.r.t. a client protocol $\mathcal{P}_C$ iff:

- $\left[\mathcal{P}_C \|^{\mathtt{TC}} \mathcal{P}_2\right]_{\mathcal{P}_2} \sqsubseteq \mathcal{P}_1$, or
- $\mathcal{P}_C \|^{\mathtt{TC}} (\mathcal{P}_2 \|^{\mathtt{TD}} \mathcal{P}_1) = \emptyset$

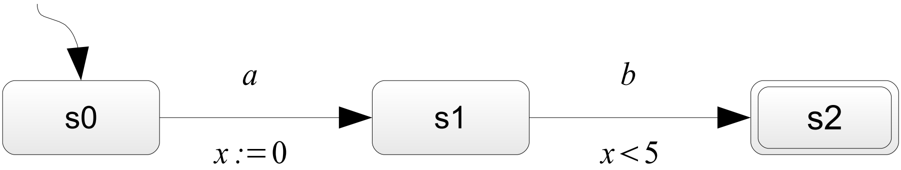
1. Algorithms and decidability?
2. Are timed protocols closed under our operators?

Outline

Timed automata

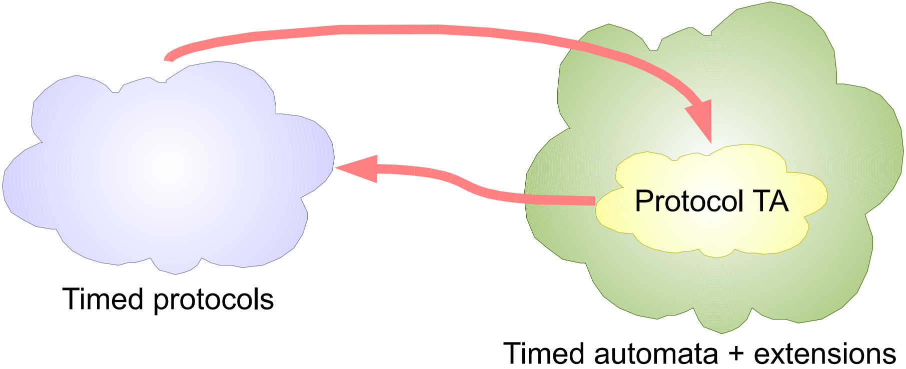
(Alur, Dill 1994)

- Clocks over dense time + constraints + resets
- Vibrant research
- Use-cases: {system, property} $\longrightarrow$ checker $\longrightarrow$ $\{\mathtt{yes}, \mathtt{no}\}$

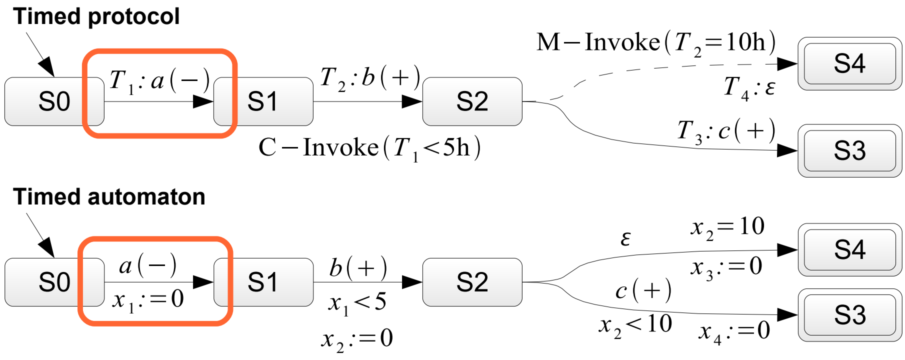


"Timed words such that a follows b by at most 5 units of time"

Mapping

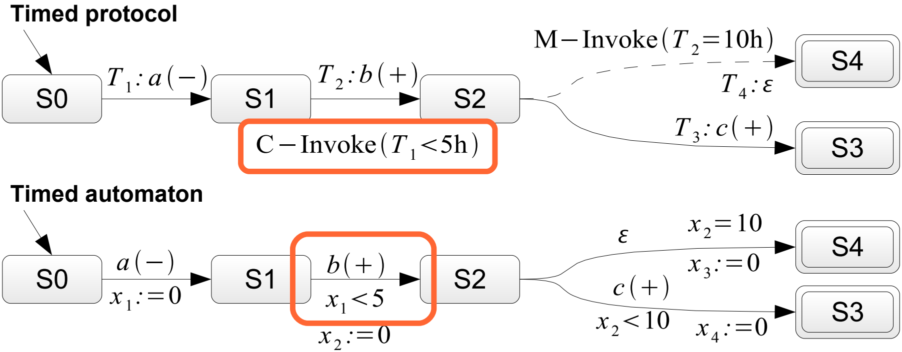


Mapping



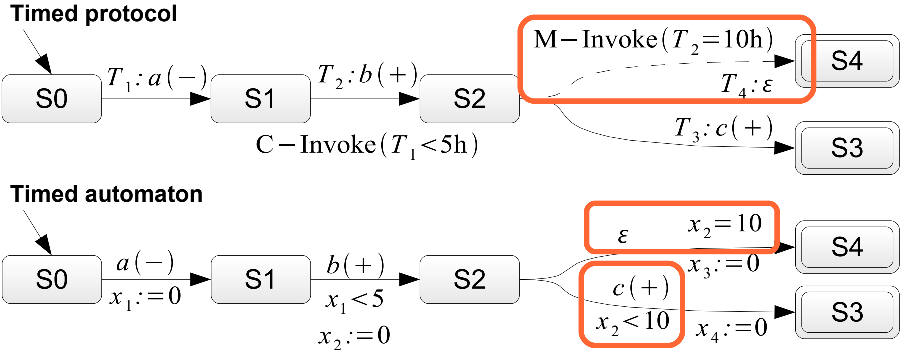
Timed protocol
S0
$T_1 : a(-)$
S1
$T_2 : b(+)$
S2
$\mathrm{M-Invoke}(T_2 = 10\mathrm{h})$
S4
$T_4 : \varepsilon$
$T_3 : c(+)$
S3
$\mathrm{C-Invoke}(T_1 < 5\mathrm{h})$
Timed automaton
S0
$a(-)$
$x_1 := 0$
S1
$b(+)$
$x_1 < 5$
$x_2 := 0$
S2
ε
$x_2 = 10$
$x_3 := 0$
S4
$c(+)$
$x_2 < 10$
$x_4 := 0$
S3

Mapping



Timed protocol
S0
$T_1 : a(-)$
S1
$T_2 : b(+)$
S2
$\mathrm{M-Invoke}(T_2 = 10\mathrm{h})$
$T_4 : \varepsilon$
S4
$T_3 : c(+)$
S3
$\mathrm{C-Invoke}(T_1 < 5\mathrm{h})$
Timed automaton
S0
$a(-)$
$x_1 := 0$
S1
$b(+)$
$x_1 < 5$
$x_2 := 0$
S2
ε
$x_2 = 10$
$x_3 := 0$
S4
$c(+)$
$x_2 < 10$
$x_4 := 0$
S3

Mapping



Timed protocol
S0
$T_1 : a(-)$
S1
$T_2 : b(+)$
S2
$\mathrm{M-Invoke}(T_2 = 10\mathrm{h})$
$T_4 : \varepsilon$
S4
$T_3 : c(+)$
S3
$\mathrm{C-Invoke}(T_1 < 5\mathrm{h})$
Timed automaton
S0
$a(-)$
$x_1 := 0$
S1
$b(+)$
$x_1 < 5$
$x_2 := 0$
S2
ε
$x_2 = 10$
$x_3 := 0$
S4
$c(+)$
$x_2 < 10$
$x_4 := 0$
S3

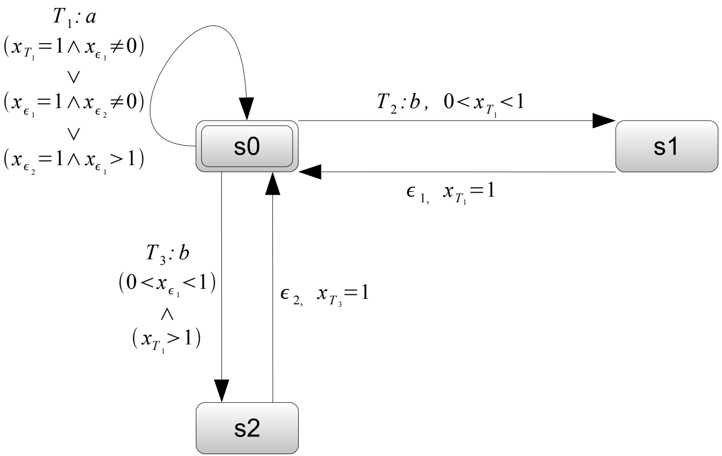
The case of ε-transitions

They have clock resets and they cannot be removed!

Proof

Based on *precise actions* (Bérard, Diekert, Gastin, Petit 99)

The case of ε-transitions



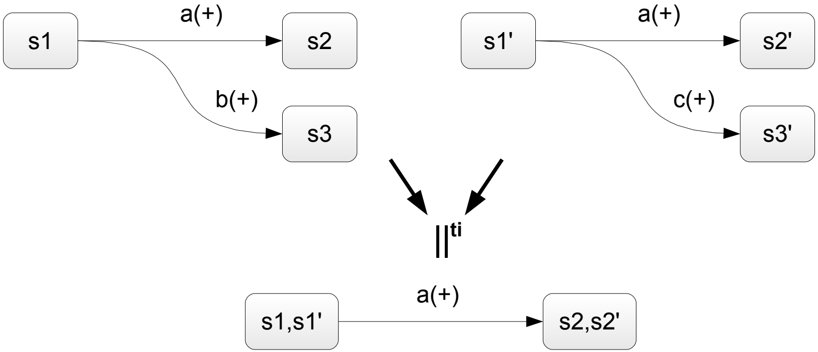
$(b, \delta_1) \cdot (b, \delta_2) \cdots (b, \delta_{d-1}) \cdot (a, d) \cdot (a, d+1) \cdots$

$\longrightarrow$ the occurrences of a-events should be precise

The case of intersection / composition

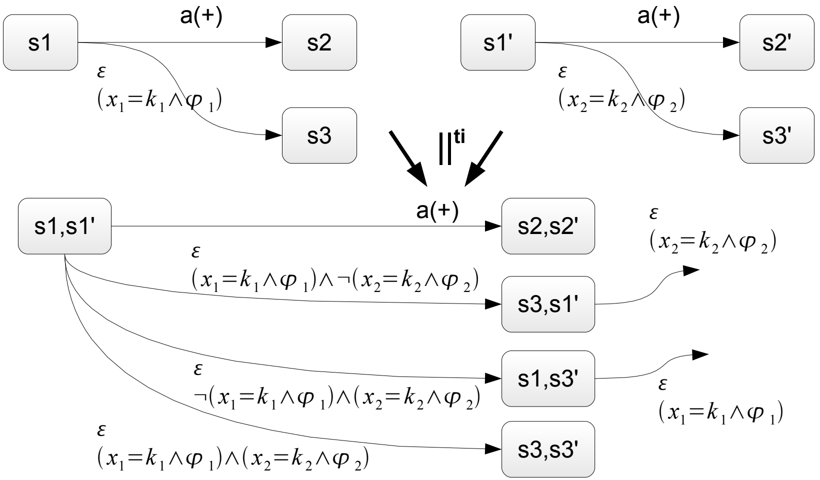
Usual technique

Product with label synchronization



Determinism problem: ε-transitions are never synchronized!

The case of intersection / composition

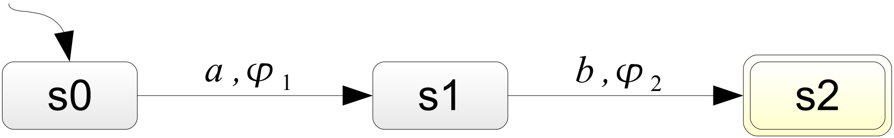


Keeps semantics and determinism!

(mandatory $(x_i = k_i)$ clauses in M-Invoke)

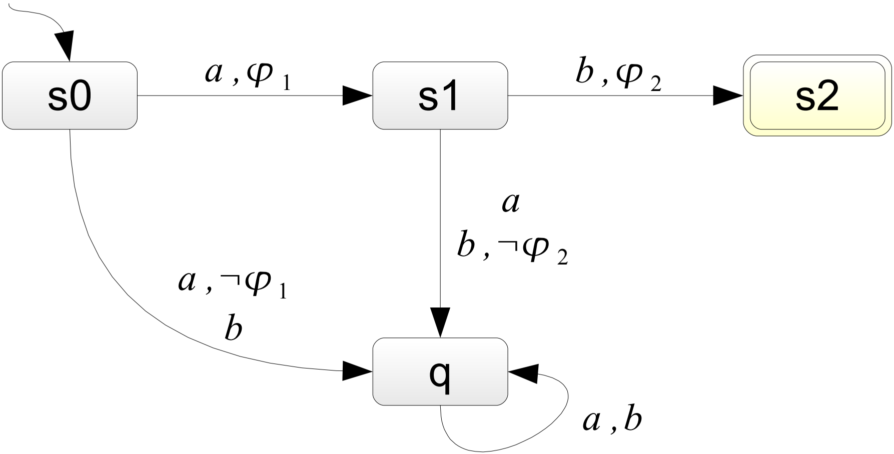
The case of difference / complementation

Extension of the procedure on deterministic TA



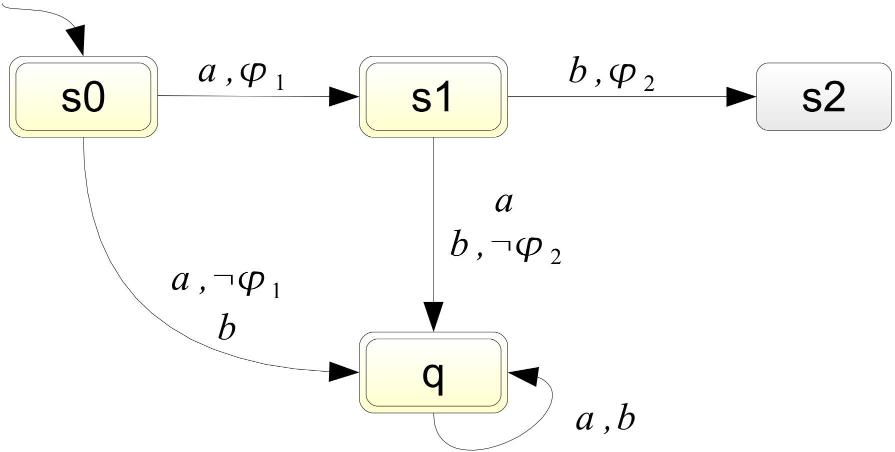
The case of difference / complementation

Extension of the procedure on deterministic TA



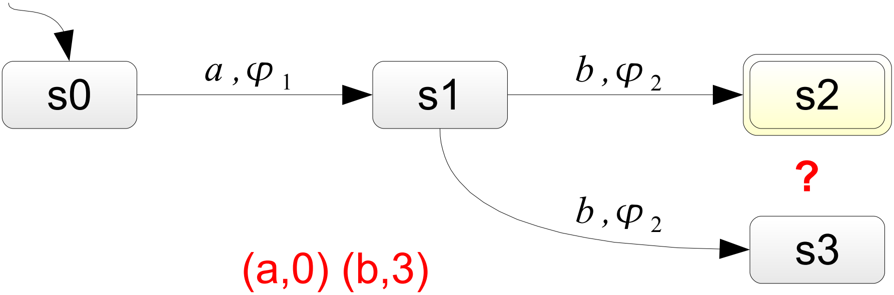
The case of difference / complementation

Extension of the procedure on deterministic TA



The case of difference / complementation

Needs exactly 1 run per recognized timed word!



The extended complementation procedure keeps this property

The case of subsumption / equivalence

The test $\mathcal{P}_1 \sqsubseteq \mathcal{P}_2$ is equivalent to $\mathcal{P}_1 \cap \overline{\mathcal{P}_2} = \emptyset$ (timed language inclusion problem)

Complementation

PTA are closed under complementation

Emptiness checking (Alur, Dill 94)

The problem is PSPACE-complete

$\longrightarrow \sqsubseteq$ and $\equiv$ need a model-checker (UPPAAL, Kronos, ...)

The case of subsumption / equivalence

The test $\mathcal{P}_1 \sqsubseteq \mathcal{P}_2$ is equivalent to $\mathcal{P}_1 \cap \overline{\mathcal{P}_2} = \emptyset$ (timed language inclusion problem)

Complementation

PTA are closed under complementation

Emptiness checking (Alur, Dill 94)

The problem is PSPACE-COMPLETE

$\longrightarrow$ $\sqsubseteq$ and $\equiv$ need a model-checker (UPPAAL, Kronos, ...)

Results

1. Timed protocols are closed under manipulation operators
2. Timed automata based algorithms for manipulation and comparison operators
3. Every compatibility and replaceability class can be implemented
4. Protocol timed automata form a new class of timed automata

Outline

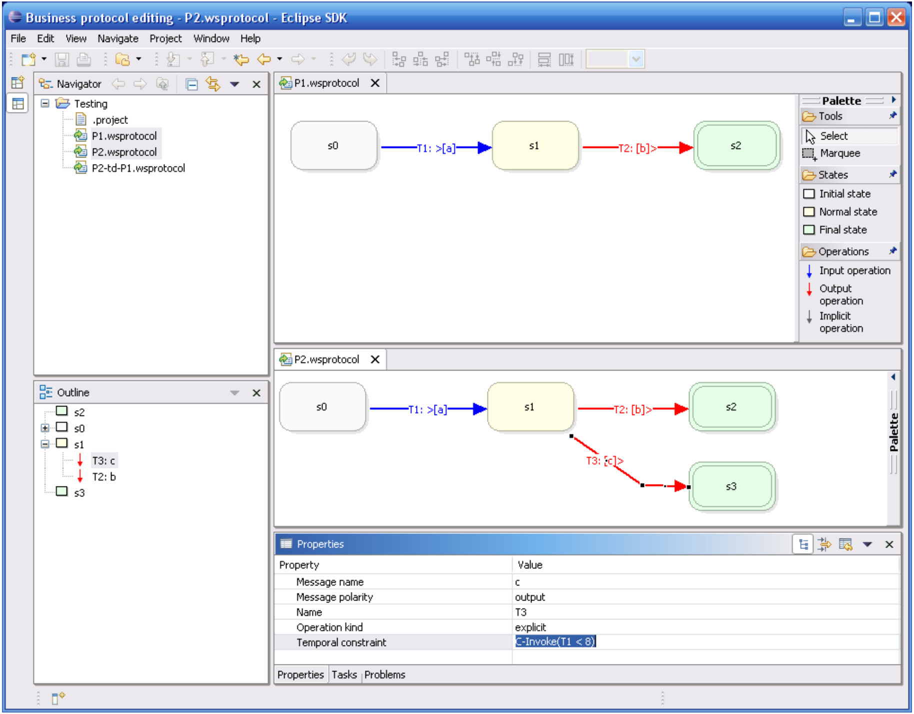
Prototype

ServiceMosaic

- Eclipse-based
- Protocol editor
- Protocol operators

Complementary modules:

- Protocol extraction from BPEL
- Protocol mining from execution logs (lead by Hamid Motahari)


Business protocol editing - P2.wsprotocol - Eclipse SDK
File Edit View Navigate Project Window Help
Navigator
Testing
.project
P1.wsprotocol
P2.wsprotocol
P2-td-P1.wsprotocol
Outline
s2
s0
s1
T3: c
T2: b
s3
P1.wsprotocol
s0
T1: >[a]
s1
T2: [b]>
s2
Palette
Tools
Select
Marquee
States
Initial state
Normal state
Final state
Operations
Input operation
Output operation
Implicit operation
P2.wsprotocol
s0
T1: >[a]
s1
T2: [b]>
s2
T3: [c]>
s3
Palette
Properties
Property
Value
Message name
c
Message polarity
output
Name
T3
Operation kind
explicit
Temporal constraint
C-Invoke(T1 < 8)
Properties
Tasks
Problems

Perspectives

- Refined expressiveness (in progress)
- Agile composition development and execution runtimes
- Analyse at the composition level
- Help BPEL engines scalability (with O. Coupelon)

Questions?

http://www.isima.fr/~ponge/

http://servicemosaic.isima.fr/

Université Blaise Pascal

UNSW THE UNIVERSITY OF NEW SOUTH WALES